WEARABLES-BASED USER IDENTITY RECOGNITION THROUGH IMAGE REPRESENTATION OF MOTION SENSOR DATA SEQUENCES AND PRETRAINED VISION MODELS

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

Rabia Ela Ünlü

July 2025

Wearables-Based User Identity Recognition Through Image Representation of Motion Sensor Data Sequences and Pretrained Vision Models
By Rabia Ela Ünlü
July 2025

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)

Süleyman Serdar Kozat

Bahaeddin Eravcı

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

WEARABLES-BASED USER IDENTITY RECOGNITION THROUGH IMAGE REPRESENTATION OF MOTION SENSOR DATA SEQUENCES AND PRETRAINED VISION MODELS

Rabia Ela Ünlü
M.S. in Electrical and Electronics Engineering
Advisor: Billur Barshan Özaktaş
July 2025

The common methods employed in User Identity Recognition (UIR) and verification are often vulnerable to cyber attacks, requiring more robust solutions. Motion sensor data and biometric data are used in tackling both the UIR and Human Activity Recognition (HAR) tasks. These tasks are mostly accomplished by using Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM), and CNN-LSTM hybrid models. We propose a method that employs pretrained CNN and vision transformer-based models to achieve the UIR task by classifying image representations of sensor data. We conduct a comparative study by evaluating the performance of various pretrained networks in the image classification task by processing four activity datasets comprising raw data sequences. We construct a new hybrid architecture which combines DeiT-B and DenseNet201 models in a parallel configuration. This study also compares two kinds of preprocessing methods which are spectrogram and wavelet spectrogram and introduces a novel approach that is fundamentally distinct from these methods. This technique fuses raw data, spectrogram, and wavelet spectrogram information. The DeiT-B model obtains the highest accuracy as 99.76% on the DSA Dataset; however, our new hybrid architecture that combines DeiT-B and DenseNet201 performs superior.

Keywords: Wearable sensors, motion sensors, accelerometer, gyroscope, magnetometer, user identity recognition, human activity recognition, deep learning, transfer learning, pretrained models, preprocessing methods.

ÖZET

HAREKET SENSÖRÜ VERİ DİZİLERİNİN GÖRÜNTÜ TEMSİLLERİ VE ÖNCEDEN EĞİTİLMİŞ GÖRME MODELLERİ İLE GİYİLEBİLİR CİHAZ TABANLI KULLANICI KİMLİK TANIMA

Rabia Ela Ünlü
Elektrik ve Elektronik Mühendisliği, Yüksek Lisans
Tez Danışmanı: Billur Barshan Özaktaş
Temmuz 2025

Kullanıcı kimliği tanıma ve doğrulama süreçlerinde yaygın olarak kullanılan yöntemler, siber saldırılara karşı sıklıkla savunmasız kalmakta ve bu süreçler daha sağlam çözümler gerektirmektedir. Bu bağlamda, hareket sensörü verileri ve biyometrik veriler alternatif çözümler olarak kullanılmakta ve bu durum insan hareketi tanıma görevini gündeme getirmektedir. Bu görev, genellikle evrişimli sinir ağları, uzun kısa süreli bellek ve söz konusu iki tip modelin birleştirilmesiyle oluşturulan hibrit (melez) modelleri kullanılarak gerçekleştirilmektedir. Bu çalışmada, sensör verilerinin görüntü temsillerini sınıflandırarak kullanıcı kimliği tanıma görevini gerçekleştirmeyi amaçlayan, önceden eğitilmiş evrişimli sinir ağları ve dönüştürücü tabanlı modelleri kullanan bir yöntem önerilmiştir. Zaman serisi ham verilerinden oluşan dört farklı aktivite veri kümesi kullanılarak önceden eğitilmiş çeşitli ağların görüntü sınıflandırma görevindeki başarımları değerlendirilmiş ve karşılaştırmalı bir çalışma gerçekleştirilmiştir. Ayrıca, DeiT-B ve DenseNet201 mimarilerini birleştiren yeni bir hibrit mimari geliştirilmiştir. Bunlara ek olarak, bu çalışma spektrogram ve dalgacık spektrogram olmak üzere iki farklı önişleme yöntemini karşılaştırmakta ve bu yöntemlerden yapısal olarak farklı olan özgün bir yaklaşım sunmaktadır. Söz konusu yöntem, ham veri, spektrogram ve wavelet spektrogram bilgilerini birleştiren bir füzyon (tümleşim) tekniği olarak öne çıkmaktadır. DeiT-B modeli, DSA veri kümesinde %99.76 doğruluk elde etse de, DeiT-B ve DenseNet201 tabanlı hibrit mimarimiz daha yüksek performans (başarım) göstermiştir.

Anahtar sözcükler: Giyilebilir sensörler, hareket sensörleri, ivmeölçer, dönüölçer, manyetometre, kullanıcı kimliği tanıma, insan hareketi tanıma, derin öğrenme, öğrenme aktarması, önceden eğitilmiş modeller, önişleme yöntemleri.

Acknowledgement

I would like to express my sincere gratitude to my supervisor Prof. Dr. Billur Barshan Özaktaş for her support and guidance during the preparation of this thesis.

Furthermore, I would also like thank the members of my jury, Prof. Dr. Süleyman Serdar Kozat and Asst. Prof. Dr. Bahaeddin Eravcı for reviewing this thesis and providing their supportive comments, feedback, and criticism.

I would like to thank Turkcell and Information and Communication Technologies Authority (BTK) for their support of my master's degree through the 5G and Beyond Scholarship.

Finally, I want to express my sincere gratitude to my mother for her endless love, support, and motivation throughout my master's journey. Without her, I would not have been able to complete this thesis. My sincere appreciation is reserved for my father and brother for their help and support during this challenging period.

Contents

List of Figures

List of Tables

List of Abbreviations

UIR : User Identity Recognition

HAR : Human Activity Recognition

MEMS : Micro-Electromechanical System

DSA Dataset : Daily and Sports Activities Dataset

USC-HAD Dataset : University of Southern California Human Activity Dataset

UCI-HAR Dataset : University of California Irvine Human Activity Recognition Dataset

ML : Machine Learning

DL : Deep Learning

TL : Transfer Learning

NN : Neural Networks

CNN : Convolutional Neural Network

SNN : Sequential Neural Network

RNN : Recurrent Neural Network

MLP : Multi-Layer Perceptron

SVM : Support Vector Machine

LSTM : Long Short-Term Memory

Bi-LSTM : Bidirectional LSTM

ConvLSTM : Convolutional LSTM

ResNet18 : Residual Network-18

DenseNet201 : Densely Convolution Network-201

ReLU : Rectified Linear Unit

ViT : Vision Transformer

DeiT : Data-efficient Image Transformer

GAF : Gramian Angular Field

MTF : Markov Transition Field

SI : Signal Image

RP : Recurrence Plot

CCF : Canonical Correlation-based Fusion

FT : Fourier Transform

DFT : Discrete Fourier Transform

STFT : Short-Time Fourier Transform

DWT : Discrete Wavelet Transform

CLS : Classification token

RBF : Radial Basis Function

RGB : Red, Green, Blue

1D : one-dimensional

2D : two-dimensional

Chapter 1

Introduction

1.1 Problem Definition

The increase in the usage of smart devices has enhanced the importance of cybersecurity. Under cybersecurity, User Identity Recognition (UIR) and verification are important tasks that have drawn considerable attention recently. Traditional methods have been employed to handle these tasks which are the use of passwords and fingerprints. However, these methods have weaknesses as they can become inefficient as a result of processed and exploited attacks. To overcome these issues, motion sensor data and biometric data are utilized [1]. Various human activities can be differentiated by processing motion sensor data for the purpose of Human Activity Recognition (HAR). Accelerometers, gyroscopes, and magnetometers serve as the primary types of motion sensors. Fig. 1.1 presents typical raw sensor signals of each axis of each three sensor type from the DSA Dataset, collected from User 1.

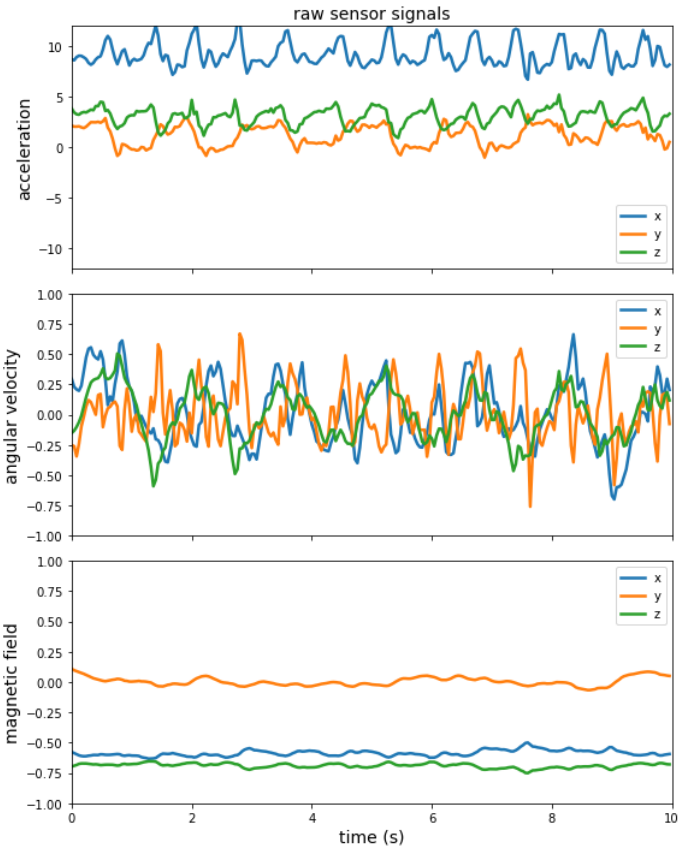


Figure 1.1: Typical raw sensor signals available in the DSA Dataset.

Many works based upon Machine Learning (ML) and Deep Learning (DL) methods process sensory data to classify users [2]. Commonly used approaches are Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM), and CNN-LSTM hybrid models [3]. These methods focus on either HAR or UIR, both of which are classification tasks (Fig. 1.2). Some of the works feed sensory data to models in the form of images [4], [5], [6] so that the classification of raw data sequences turns into an image classification task. The most commonly employed models utilized in image classification are CNN architectures and transformer-based models. Data augmentation is an important process since the amount of images is often not sufficient for the model to work accurately. Another important concept to improve the performance of DL models is the incorporation of Transfer Learning (TL). With the help of TL, pretrained models are leveraged, resulting in more accurate results. The method

proposed in this work aims to apply pretrained CNN and transformer-based models to tackle the UIR task by classifying image representations of motion sensor data (Fig. 1.3).

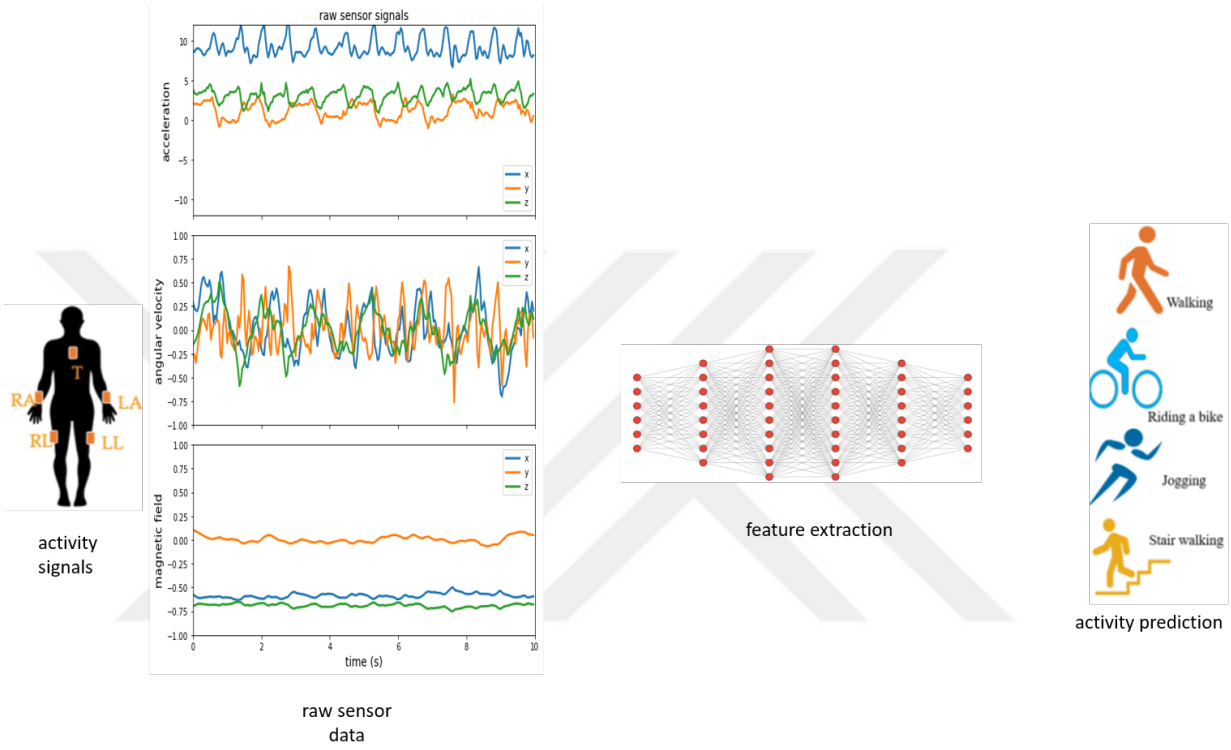


Figure 1.2: DL-based HAR.

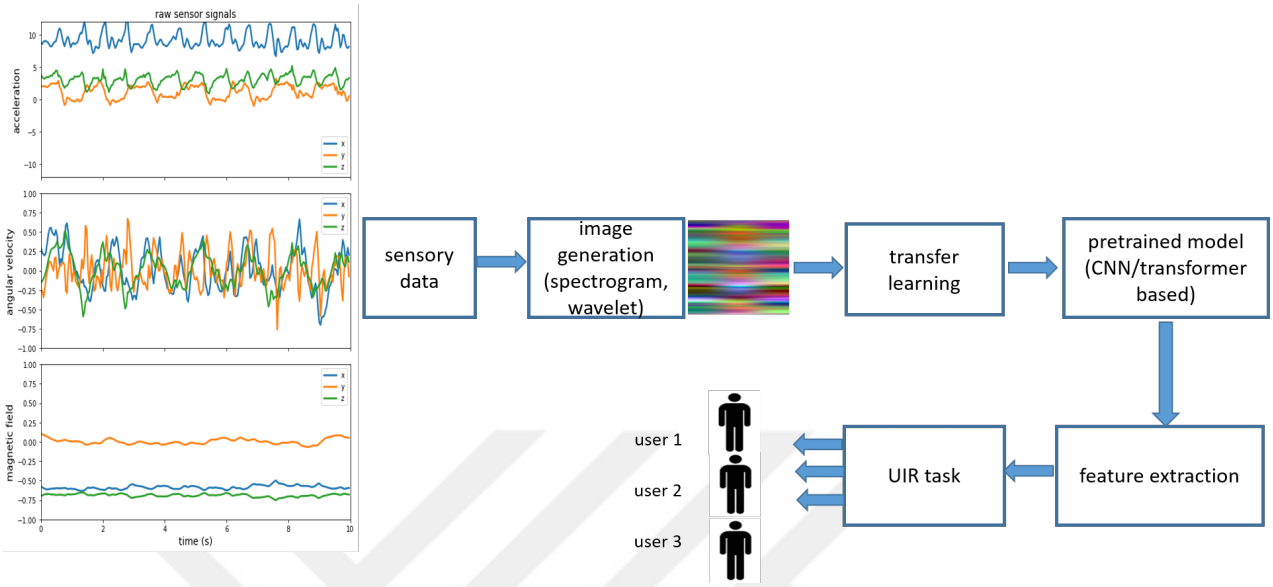


Figure 1.3: The UIR method proposed in this work.

1.2 Related Work

1.2.1 User Identification with Motion Sensor Data

Raval and Anwar [7] achieve passive UIR by processing data acquired by smart phone sensors. They compare five models which are Multi-Layer Perceptron (MLP), Sequential Neural Network (SNN), CNN, Support Vector Machines (SVMs), and Bidirectional LSTM (Bi-LSTM). Shi et al. [8] set up a UIR framework called SenGuard. The authors of that work employ four sensor modalities which are voice, multitouch, location, and locomotion. They use a tri-axial Micro-Electromechanical System (MEMS) motion sensor. Kim et al. [9] propose a LSTM-based UIR method using accelerometer data from smart shoes. They train the model with walking data of random sizes acquired from different locations. Lopez et al. [10] manage inertial gate UIR by using data collected by smart phone sensors and feed these data into a Recurrent Neural Network (RNN). Qin et al. [11] propose a framework for smart phone sensor UIR

(SSUI) which is of hybrid nature since it includes both CNN and RNN. Mekruksavanich and Jitpattanakul [12] present a novel framework for multi-class wearables-based UIR. In this two-stage framework, first they classify the activity type, followed by user identification based on the classified activity. They train the framework with data collected from motion sensors and attain the highest accuracies by using CNN and LSTM. In addition, they implement hybrid structures which are CNN-LSTM and Convolutional LSTM (ConvLSTM). Ehatistam-ul-Haq et al. [13] present a novel multi-class UIR scheme for the authentication of smart phone users where the users are identified again based on their activities. In that work, the Bayes Net classifier outperforms other classifiers in accurate classification of the selected activities.

1.2.2 Image Representation of Motion Sensor Data

Jafari et al. [14] generate images from time-series data collected from different types of sensors. Then, the authors extract the features to realize the classification task. They feed the images into a CNN to handle different tasks such as HAR and stress detection. The main difference between this work and ours is that in Jafari's work, the authors implement a CNN architecture whereas in our work, we employ pretrained CNN-based and transformer-based image classifier models. Wang and Oates [15] generate images based on time-series data through the use of Gramian Angular Fields (GAF) and Markov Transition Fields (MTF). They feed the images into a CNN architecture to accomplish the classification task. Jiang and Yin [16] generate images by first converting sensory data into a Signal Image (SI), followed by applying the two-dimensional (2D) Discrete Fourier Transform (DFT) for creating activity images. For the classification task, they implement a CNN with two layers, which is adopted by filters in order as 5×5, 4×4, and 2×2 average pooling layers. Xu et al. [17] convert one-dimensional (1D) sensor data into 2D images through the use of GAF; then, they adopt a data fusion network called as Fusion-MdkResNet. Ahmad and Khan [18] generate activity images as SI, GAF, MTF, and Recurrence Plot (RP) images. The similarity between this work and ours is the implementation of ResNet-18 for classification but it differs from our work in the use of Canonical Correlation based

Fusion (CCF). Bakshi [19] employs ViT with attention layer for fall and activity detection. The author utilizes accelerometer data collected from a waist-worn sensor, applies the Short-Time Fourier Transform (STFT) to the data, and subsequently generates STFT spectrogram images. The experiments differ according to various types of patch sizes and attention heads. Benegui and Ionescu [20] convert tri-axial signals acquired from motion sensors into gray-scale images. They apply a CNN for feature extraction and employ binary SVMs for classification.

1.2.3 Image Classifier Models

A network that has already been trained on a large dataset produces an already pretrained model, which is typically distinguished by a high level of generalization. The two primary methods for TL are feature extraction and fine-tuning. In feature extraction, some existing layers are used as they are and a layer for classification is added to the model. In fine-tuning, some layers are not frozen and are available for training; these adjust the learned parameters by using custom data.

Some pretrained models are given data-augmented images as input. These models differ according to whether they are based on CNN or transformer. Their approaches of processing visual information and architectural design are where they diverge the most.

CNNs employ convolutional layers to extract information in a hierarchical fashion, while transformers directly capture global dependencies and relations across picture patches through the application of self-attention processes. The UIR task is considered as a multi-class classification task where each class corresponds to a different user. The CNN models that we have used are Residual Network-18 (ResNet18) and Densely Convolution Network-201 (DenseNet201). He et al. [21] introduce the ResNet model to overcome the degradation problem caused as a result of increase in depth. This model solves this problem by making residual connections instead of full transformations. ResNet passes the output to the following layer or block by performing an element-wise addition. First, they consider ResNet18 (meaning that this architecture

is 18 layers deep) and ResNet34. After that, they employ ResNet with 50, 101, and 152 layers. Huang et al. [22] propose DenseNet201 to solve the problem caused as a result of vanishing gradients. DenseNet establishes direct connections between each layer and all the preceding layers, which makes the gradient flow better. They try DenseNet121 (which is a CNN that is 121 layers deep), DenseNet161, DenseNet169, and DenseNet201. Dosovitskiy et al. [23] introduce ViT to show transformers can be implemented in image classification instead of CNNs. In this architecture, the authors divide the images into patches, following this by linear embedding, and lastly giving these to a transformer encoder. Touvron et al. [24] present DeiT to manage the image classification task in situations when there is not sufficient amount of data. DeiT is also a variation of ViT which has the same architecture as ViT. There is a small difference between them as the input token of DeiT has an additional distillation token. This token takes information from another network, that enables the model to learn both from the dataset and the other network. Liu et al. [25] handle high resolution images by using a transformer with shifted windows, named as Swin transformer. The difference between ViT and Swin comes from these windows: ViT uses multi-head self attention mechanism while Swin transformer uses multi-layer shifted windows. In our work, we have implemented ResNet18 and ResNet152 to observe the effect of depth on the performance of image classification. The transformer-based models that we have used are ViT, Data-efficient Image Transformer (DeiT-B), and Swin-B. We employ TL through these pretrained models and freeze the pretrained weights except those of the classification layer. Then, we update the classification layer according to our task and add a linear layer with 512 neurons, a Rectified Linear Unit (ReLU) activation layer, a dropout layer (with 0.3 rate), another linear layer (number of neurons differ according to number of class in the dataset), and a LogSoftmax activation layer.

1.2.4 Transfer Learning for User Identification

Transfer learning (TL) has become a potent method in user identification tasks, especially when labeled data are hard to acquire or prohibitively expensive. By utilizing knowledge from pretrained models, referred to as *source domains*, on large-scale datasets, TL allows for effective adaptation to specific domains, referred to as *target domains*, like biometric recognition, activity-based identification, or sensor-based user profiling. CNNs and transformer-based architectures pretrained on benchmark datasets (e.g., ImageNet) have been fine-tuned to classify users based on image representations of motion or behavioral patterns in the context of vision-based and inertial sensor-based user identification. This method enhances generalization and reduces the training time, particularly when working with imbalanced or small-scale datasets.

We can observe TL methods under three perspectives, which are (i) solution-oriented, (ii) data-oriented, and (iii) model-oriented solutions. Since we have used a model-oriented approach in this work, we transfer model architectures and parameters. The first type of model-oriented TL is parameter sharing. This method includes transferring parameters, mostly weights. Certain layers of the source model are kept fixed. The second type is model initilization. This comprises starting training by using the weights derived from a pretrained source model. The third type of model-oriented TL is modular transfer, which means transferring only particular modules rather than the entire model.

Several studies have revealed the effectiveness of TL in HAR, which is closely related to user identification. Researchers have fine-tuned models pretrained on general image classification tasks using spectrograms or wavelet-transformed images derived from motion sensor data, achieving improved performance compared to training from scratch. Similarly, transformer-based models such as Data-efficient Image Transformers (DeiT) and Vision Transformers (ViT) have exhibited outstanding performance in identifying temporal dependencies and fine-grained patterns that are essential for differentiating across users with comparable motion profiles.

Apart from model-level transfer, researchers have investigated domain adaptation

methods including aligning feature spaces or employing adversarial training strategies to close the gap between source and target distributions. These methods further improve robustness when implementing UIR systems across devices, sessions, or environmental conditions, as also demonstrated in recent work by Barshan and Koşar [26] where bidirectional transfer learning between HAR and UIR tasks was successfully implemented.

Overall, TL continues to play an essential role in advancing UIR systems, enabling scalable, precise, and data-efficient solutions across a range of input modalities and deployment circumstances.

1.3 Contributions and Outline of the Thesis

Two main components of this work are tackling the UIR task by processing motion sensor data and using time-series motion sensor data in the form of images. The main contribution of this thesis is to present a comparative study on the performance of different kinds of pretrained networks in classifying time-series motion sensor data based on four different datasets represented in image form. We construct a new hybrid architecture which combines two pretrained networks which are DeiT-B and DenseNet201 in parallel form. In addition to these, we compare two kinds of preprocessing methods which are spectrogram and wavelet spectrogram.

The remainder of this thesis is organized as follows. The present chapter describes the related work based on application of image representations of the motion sensor data and handling the UIR task with Neural Networks (NN). Chapter 2 gives an overview of the proposed approach based on image generation. The chapter begins with a description of the four datasets employed in this study, followed by explaining the method adopted in image generation. We detail our method by presenting the preprocessing stages and the implemented cross-validation techniques. Furthermore, we emphasize the procedure for hyper-parameter tuning performed for each dataset in relation to the corresponding models and provide details on implementation and

resources. Lastly, we share our performance analysis with accuracy scores. Chapter 3 describes the data augmentation methods applied to the datasets and the utilized image classifier models. We present our novel hybrid structure and provide the results of our experiments in terms of four performance metrics: accuracy, precision, recall, and f_1 score. Additionally, we provide the training times of the CNN-based and transformer-based models for each dataset. Chapter 4 summarizes the work presented in this thesis, draws conclusions, and suggests some future research directions.

Chapter 2

Generating Image Representations for Enhanced User Identity Recognition: A Novel Image Generation Method

2.1 Datasets

In this subsection, we describe the four datasets used in this study. The first dataset was acquired by our research group and the remaining three are benchmarking datasets, all publicly available. We summarize the properties of the four datasets in Table 2.1. The given number of images is without augmentation.

1) Daily and Sports Activities (DSA) Dataset: Nineteen different daily and sports activities are performed by eight subjects (four male, four female) [27], [28], [29]. Each subject has five sensor units placed on their chest (torso), right/left arm, and right/left leg. Each sensor unit collects data by using three tri-axial sensors that are accelerometer, gyroscope, and magnetometer at a sampling rate of 25 Hz. Data

acquisition is performed through 45 signal channels (5 sensor units×3 sensor types×3 axes). The activities are performed for five minutes and data are segmented into non-overlapping five-second segments and stored in segmented form. This dataset is balanced since data recorded from each user for each activity type is represented with an equal number of image representations.

2) USC-HAD Dataset: This dataset comprises 12 activities performed by 14 subjects (seven male and seven female) [30]. The activities are: walking forward, walking left, walking right, walking upstairs, walking downstairs, running forward, jumping, sitting, standing, sleeping, elevator up, and elevator down. Each user is represented with a different number of recordings corresponding to his/her activities which makes this dataset imbalanced. A MotionNode is used as the sensor unit which is extremely small in size and connected to a miniature laptop. The right hip is selected as the location to wear the sensor unit. Data are collected by using accelerometer and gyroscope, both tri-axial, resulting in six readings from each sensor. The sampling frequency is selected as 100 Hz.

3) MotionSense Dataset: An iPhone 6s in the subject's front pocket with SensingKit is employed, which collects information from Core Motion framework on iOS devices [31]. Six activities in 15 trials are performed by 24 participants of different genders (14 men and 10 women), ages, heights, and weights. The six activities are, walking, sitting, standing, jogging, going upstairs, and downstairs. Accelerometer and gyroscope are utilized to collect the data, again with six readings taken from each tri-axial sensor. The data are collected with a sampling rate of 50 Hz and the sensor readings include altitude, gravity, user acceleration, and rotation rate.

4) UCI-HAR Dataset: The dataset contains activity recordings of 30 participants between the ages 19 and 48 as they perform six activities [32]. A mobile phone (Samsung Galaxy S2), carried on the subject's waist, is used. The recorded activities are: walking on a flat surface, walking upstairs, walking downstairs, sitting, standing, and lying. The number of samples for each user differs, making this dataset also imbalanced. Consequently, the number of images associated with each user for identifying his/her activity is different. Data from an accelerometer and gyroscope

are collected at a sampling rate set of 50 Hz. Both of these sensors have tri-axial structure so that six readings are taken from each sensor unit at the same time. The dataset contains 7,352 train and 2,947 test samples. Accelerometer data which are provided in this dataset comprise total acceleration and the body component of the acceleration. The body component of the acceleration is obtained by subtracting the gravitational component of acceleration from the total acceleration.

Table 2.1: Properties of the four datasets used in this thesis. The given number of generated images is without augmentation.

dataset	**no. of subjects**	**sampling frequency (Hz)**	**dimensions of segments**	**no. of images**
DSA [**?**]	8	25	125×9	480
USC-HAD [30]	14	100	500×6	342
MotionSense [31]	24	50	250×6	1366
UCI-HAR [32]	30	50	128×6	1692

2.2 The Proposed Methods

In the first image generation method (Method 1), we map acceleration time-series data to the red channel, gyroscope data to the green channel, and magnetometer data to the blue channel. In the second image generation method (Method 2), grouping is done based on the axes of each sensor modality. We encode the x-axes of the accelerometer, gyroscope, and magnetometer to the red channel, the y-axes to the green channel, and the z-axes to the blue channel. We introduce a novel image generation approach (Method 3) in which acceleration time-series data are assigned to the red channel, spectrogram data to the green channel, and wavelet-transformed data to the blue channel (Fig. 2.1). Unlike the other image generation techniques that we have used, the RGB images produced with this method encapsulate signal characteristics both in the time-domain and the frequency-domain.

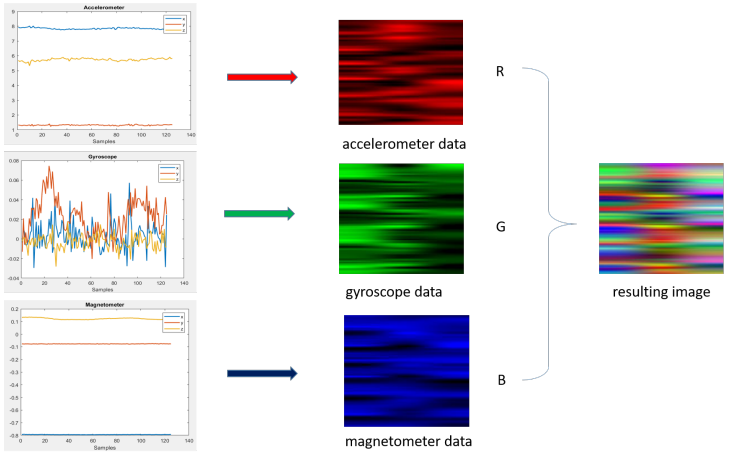
2.2.1 Generating the Image Sequences

To manage the UIR task, we have selected a common activity included in all four datasets which is walking. In our experiments, we have focused on the walking activity across all four datasets. For the DSA Dataset, we generate the images by using data from activity A9, which corresponds to walking, recorded from the sensor unit on the torso. In the USC-HAD and UCI-HAR Datasets, walking corresponds to activity A1. For the MotionSense Dataset, we chose and then combined the files named as wlk7, wlk8, and wlk15. These files are all related to the walking activity where the only difference between them is their duration. The number of constructed images corresponding to each user differs, depending on the sampling frequency, duration of each recording, and the number of users involved in acquiring these datasets.

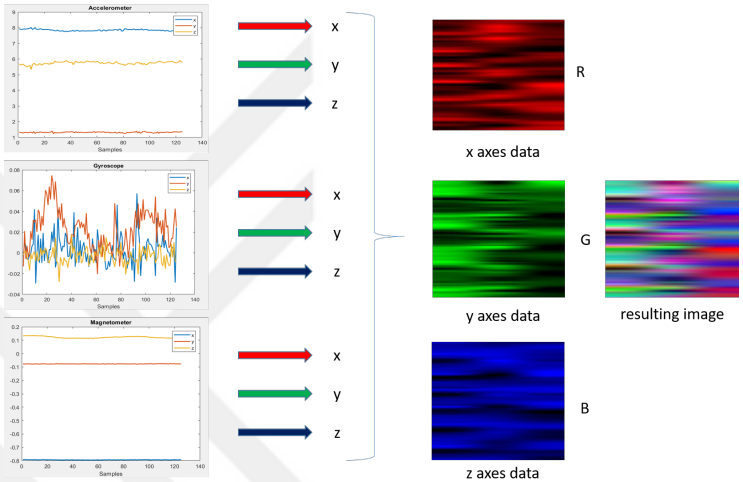
In generating the image sequences, we have taken the following steps: We divided the raw data sequences in each dataset into three channels. Each channel is made up of three groups as there are three axes. Fig. 2.1(a) illustrates the image generation process in Method 1. The first group comprises the accelerometer data, the second one contains the gyroscope data, and the last one includes the magnetometer data. Fig. 2.1(b) depicts the image generation process in Method 2. In this method, we define the x-axes data of the accelerometer, gyroscope, and magnetometer as the first group, their y-axes data as the second group, and their z-axes data as the third group. We used the raw data sequences in segmented form, meaning the sequences are divided according to the sampling frequency of the dataset and the duration of the measurement. For instance, in the DSA Dataset, there is no need to segment the dataset since the dataset is already available in segmented form. We generate images based on the segments.

A difference arises if spectrogram data are used instead of raw data. Each 1D sequence is represented as a 2D sequence as a result of frequency data generated based on the Fourier Transform (FT). The process for generating images by using spectrogram data is similar to the one based on raw data. We have scaled the data into the [0−255] interval. Most of the computer vision models are designed to process raw pixel values in this range without requiring additional normalization. Therefore,

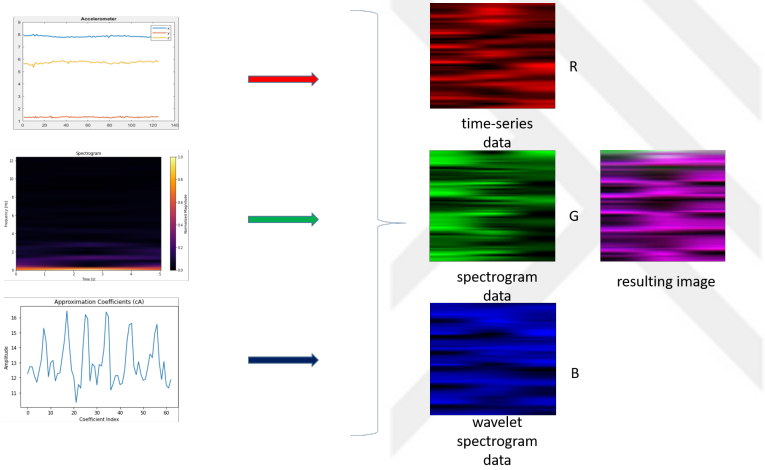
we did not apply further normalization to the data. After scaling the data, we have turned these scaled data into Red, Green, Blue (RGB) images. By using bicubic interpolation, we have resized the images to 224×224, which we illustrate in Fig. 2.1.



(a) Method 1



(b) Method 2



(c) Method 3

Figure 2.1: Image generation frameworks: (a) Method 1, (b) Method 2, and (c) Method 3.

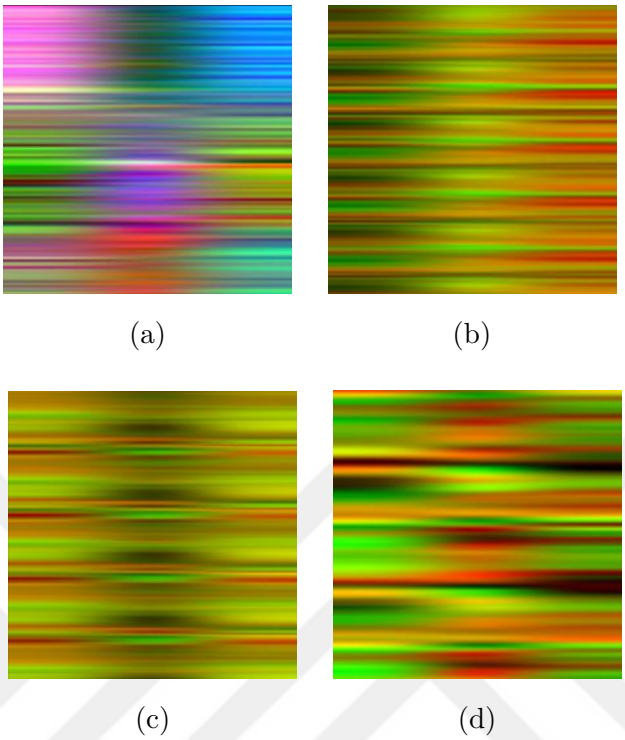


Figure 2.2: Sample images generated based on the (a) DSA, (b) USC-HAD, (b) MotionSense, and (d) UCI-HAR Datasets.

We observe in Fig. 2.2 that the generated images have different patterns, which provide us with insights regarding the dataset. For instance, analyzing the color composition of these images enables us to deduce the types of the sensors involved in the data collection process. The image in Fig. 2.2(a), consists of red, green, and blue colors and their combinations, since all three types of sensors in this dataset (accelerometer, gyroscope, and magnetometer) provide data, whereas in Fig. 2.2(b), the image comprises only red and green colors, and their combinations. This is because accelerometer and gyroscope are the two sensor types used in the USC-HAD Dataset (no magnetometer).

2.2.2 Preprocessing

2.2.2.1 Preprocessing the Data for Spectrogram Image Generation

In the experiments that use spectrogram images, we have transformed each time segment of the data by taking the magnitude of its STFT [33]. The segment length of the DSA Dataset was 125 samples. The resulting spectrograms have the dimensionalities of 63×9×3. The STFT operation can be mathematically formulated as:

$$\mathrm{STFT}\{x\}(m,\omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n}, \tag{2.1}$$

where $w[n-m]$ denotes the window function centered around time m.

2.2.2.2 Preprocessing the Data for Wavelet Spectrogram Image Generation

In some of the experiments, we need to produce wavelet spectrogram images. To do this, we have employed the Discrete Wavelet Transform (DWT) [34]. When we take the DWT of a given time sequence, two coefficients which are called the *approximation* and *detail* coefficients are calculated. For our task, it is proper to use the approximation coefficient since it is more focused on major structure and retains the low-frequency components of the frequency spectrum. We chose the mother wavelet as Daubechies 1 (db1), since this works well in capturing edges, textures, and constant variations in images. The DWT operation can be mathematically formulated as:

$$\text{approximation coefficients: } a_j[k] = \sum_{n=-\infty}^{\infty} x[n]\,\phi_{j,k}[n], \tag{2.2}$$

$$\text{detail coefficients: } d_j[k] = \sum_{n=-\infty}^{\infty} x[n]\,\psi_{j,k}[n]. \tag{2.3}$$

2.2.3 Cross Validation

Cross-validation techniques are applied when the amount of data is limited. It is important to investigate the performance of cross validation with the use of unseen data. We have used a robust type of this technique, which is called k-fold cross validation to test the performance of our image generation technique where we chose the k parameter as five (Fig. 2.3). Thus, we divided the images into five folds. We trained the model also five times. In each training, we used four folds as the training set and the remaining one as the validation set. In addition, we found the values of the parameters which are *batch size* and *learning rate* that give the highest accuracy with the help of cross validation. Because three of the four datasets that we have employed are imbalanced, we have applied *stratified cross-validation* considered in our work [35]. In some parts of the work, we have applied *hold-out cross-validation*, a widely used method in which the dataset is split into train and test sets using a fixed ratio. Bergstra and Bengio [36] also employed this method for hyper-parameter optimization.

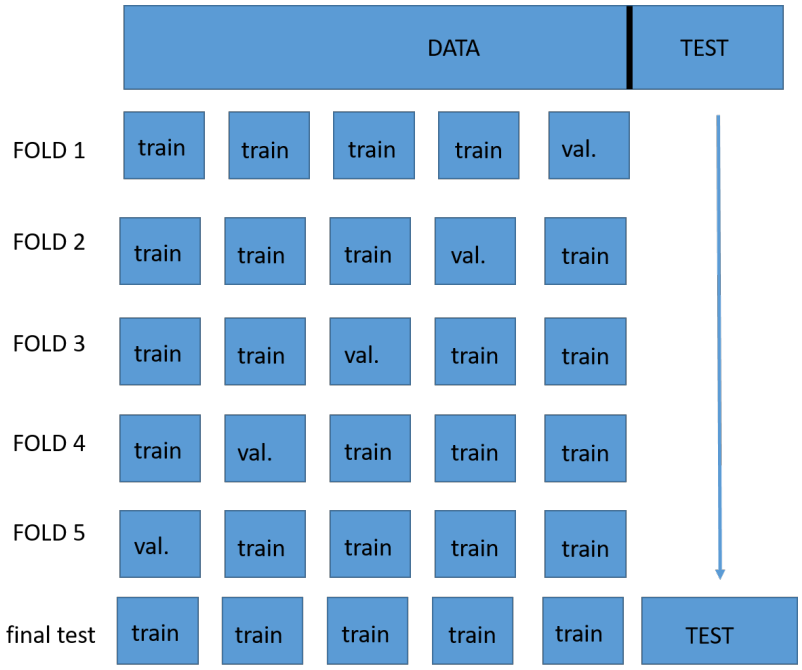


Figure 2.3: Five-fold cross validation (val.: validation).

Fig. 2.4 displays the training and validation accuracy and Fig. 2.5 depicts the train loss values both across the five folds for the DSA Dataset. Each fold gives results similar to the others. We have used the ResNet18 model. Table 2.2 presents the accuracy and loss values of each fold at the last epoch. The results for the five folds are observed to be close to each other. The average training accuracy is 98.06 ± 0.25% and the average loss is 1.17. The average value of the test accuracy is 98.13 ± 0.32%.

Table 2.2: Training and test accuracy, train loss across folds in processing the DSA Dataset.

folds	**training accuracy (%)**	**test accuracy (%)**	**train loss**
1	98.26	97.76	1.17
2	97.70	98.40	1.18
3	98.14	98.16	1.17
4	97.90	97.84	1.18
5	98.28	98.48	1.17
average accuracy:	98.06 ± 0.25	98.13 ± 0.32	1.17

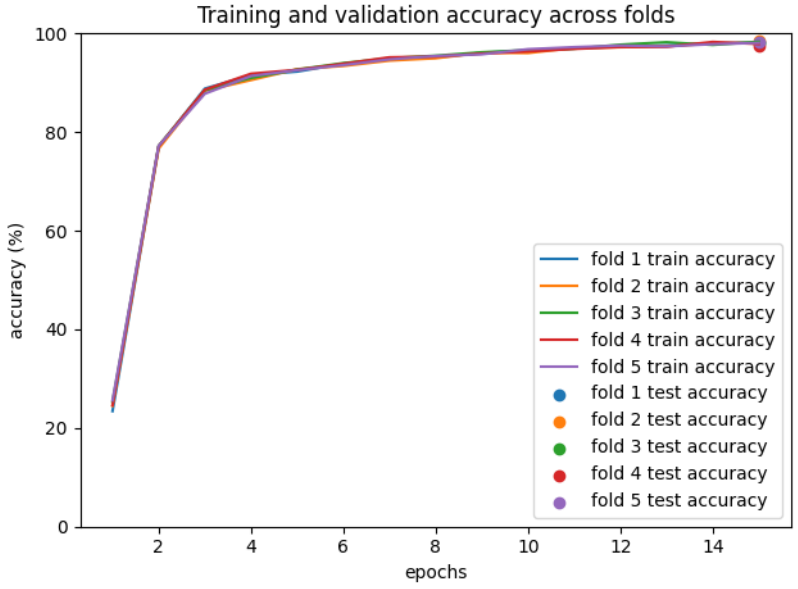


Figure 2.4: Training and test accuracy across the five folds in processing the DSA Dataset.

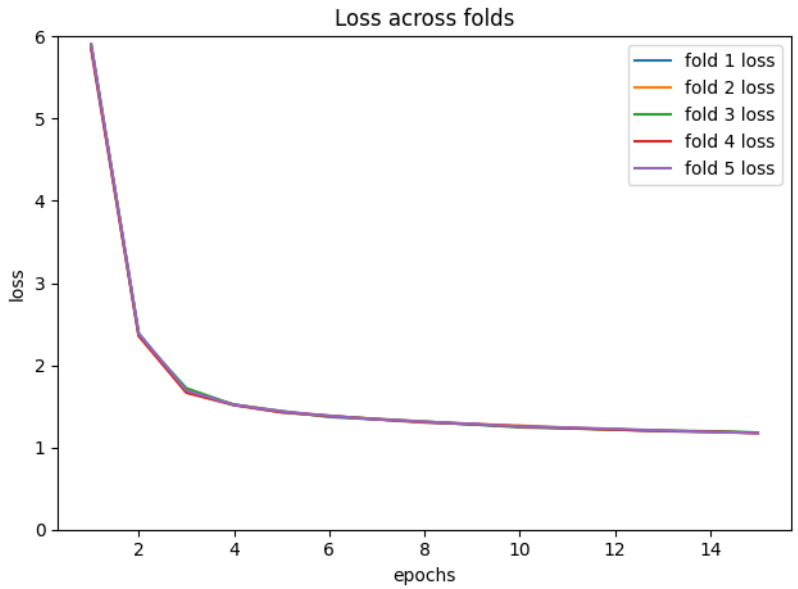


Figure 2.5: Train loss across the five folds in processing the DSA Dataset.

2.2.4 Hyper-Parameter Selection

The commonly used hyper-parameter methods [36] are grid search, random search, and Bayesian optimization.

In grid search [37], the hyper-parameter space is discretized into a predefined set of values, and all possible combinations are systematically evaluated. Since this method involves evaluating every possible combination, its major drawback is its heavy computational load.

Bayesian optimization [38] is a probabilistic approach, which estimates an objective function and selects the next set of hyper-parameters.

The third method is random search, which selects the parameters randomly, trains and scores the parameters. This method is adopted by Nugroho and Suhartanto [39] for tuning hyper-parameters of DenseNet. We have implemented this method in our work. We defined sets for the hyper-parameters to tune and considered 10 trials for

each combination.

In TL, the optimizer and the loss function are regarded as hyper-parameters; however, we have not tuned them. We have implemented the Adam optimizer and the Cross Entropy loss function.

In Table 2.3, we tabulate the test accuracy and train loss results in the hyper-parameter tuning phase for DenseNet201 in processing the DSA Dataset. We present the optimized hyper-parameters of each model based on the DSA Dataset in Table 2.4.

Table 2.3: Hyper-parameter tuning results of DenseNet201 model in processing the DSA Dataset.

batch size	**learning rate**	**test accuracy (%)**	**train loss**
32	0.001	94.19 ± 0.57	0.21
16	0.1	12.32 ± 0.43	2.09
64	0.1	11.84 ± 0.91	2.08
16	0.001	91.65 ± 1.07	0.26
64	0.5	12.24 ± 0.71	2.10
16	0.3	12.24 ± 0.82	2.13
256	0.01	93.27 ± 1.01	0.28
256	0.5	12.26 ± 0.77	2.09
64	0.001	94.95 ± 0.92	0.17

Table 2.4: Optimized hyper-parameters of the six models in processing the DSA Dataset.

model	**batch size**	**learning rate**
ResNet18	64	0.001
ResNet152	256	0.001
DenseNet201	256	0.001
ViT-B	16	0.01
DeiT-B	128	0.00001
Swin-B	32	0.0001

Table 2.5: Optimized hyper-parameters of the six models in processing the USC-HAD Dataset.

model	**batch size**	**learning rate**
ResNet18	256	0.001
ResNet152	128	0.001
DenseNet201	64	0.001
ViT-B	16	0.5
DeiT-B	32	0.00001
Swin-B	32	0.00001

Table 2.6: Optimized hyper-parameters of the six models in processing the MotionSense Dataset.

model	**batch size**	**learning rate**
ResNet18	128	0.001
ResNet152	16	0.01
DenseNet201	64	0.001
ViT-B	16	0.01
DeiT-B	64	0.00001
Swin-B	32	0.00001

Table 2.7: Optimized hyper-parameters of the six models in processing the UCI-HAR Dataset.

model	**batch size**	**learning rate**
ResNet18	256	0.001
ResNet152	128	0.001
DenseNet201	16	0.001
ViT-B	128	0.001
DeiT-B	128	0.0001
Swin-B	64	0.00001

We have tuned the hyper-parameters using training and validation sets. To ensure a reliable tuning process, we adopt k-fold cross validation. After selecting the optimal hyper-parameters, we evaluated the model performance on the test set. In our hyper-parameter tuning process, we implemented random search over a predefined set of candidate values for each parameter. We applied 10-repeated 5-fold cross validation since the DSA Dataset has a limited amount of data. We split the training data into five folds. We utilized one fold for validation and the remaining folds as the training set and identified the optimal values of the hyper-parameters based on the average performance across the folds. The balanced nature of the DSA Dataset allows the use of accuracy as the main evaluation metric.

2.3 Implementation and Resources

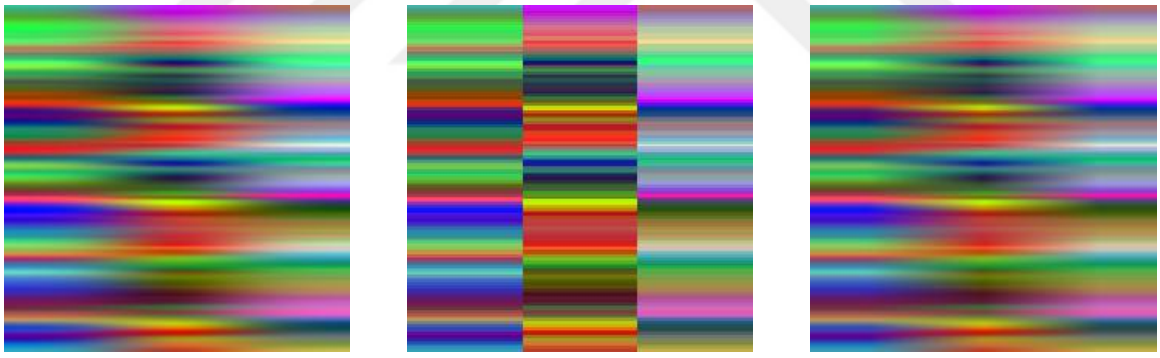
We have conducted the image representation of motion sensor data in Spyder [40]. We have implemented the pretrained models by using PyTorch library [41] in Python. Since the UIR task by using images is computationally intensive, to overcome the computational burden, we have used Google Colab platform [42] which provides NVIDIA Tesla K80 GPU.

2.4 Experimental Results and Discussion

We first implement two experiments for the image generation procedure. These are observing the effect of resizing techniques and domain of data. In addition, we compare Method 1 and Method 2. After that, we propose a new approach for image generation.

2.4.1 Effect of Resizing Techniques

In this part, we analyze the effect of resizing techniques. These techniques are nearest neighbour, bilinear interpolation, and bicubic interpolation. We have employed the standard resizing technique, which is bicubic interpolation, in the remaining experiments.



(a) (b) (c)

Figure 2.6: Images generated by using (a) bicubic, (b) nearest neighbour, and (c) bilinear interpolation.

Fig. 2.6 displays the images that we have generated by using these techniques. All of them are based on data recorded from User 1 in the DSA Dataset.

In this chapter, as we employ only the DSA Dataset which is balanced, we provide the accuracy results to measure the performance of the image generation techniques. In contrast, in the following chapter, the study goes beyond accuracy and evaluates

three other performance metrics (precision, recall, and f_1 score) to achieve a more comprehensive performance analysis.

2.4.1.1 Accuracy Scores of Different Image Resizing Methods

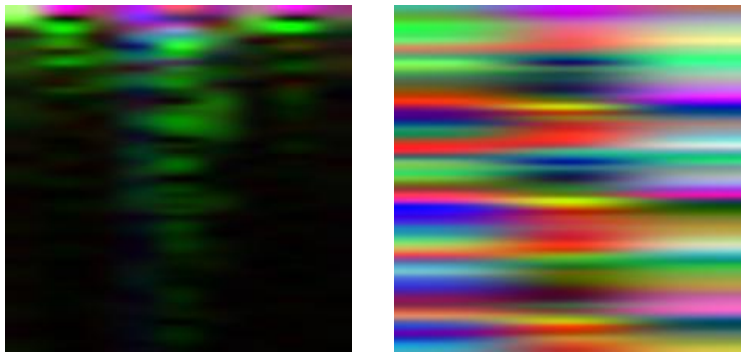
Table 2.8: Accuracy scores (%) of different image resizing methods.

Model	**Bicubic**	**Nearest Neighbour**	**Bilinear**
ResNet18	97.98 ± 0.46	97.40 ± 0.19	98.06 ± 0.32
ResNet152	97.90 ± 0.31	97.47 ± 0.50	98.37 ± 0.28
DenseNet201	98.16 ± 0.35	97.21 ± 0.52	98.35 ± 0.24
ViT-B	98.30 ± 1.29	97.87 ± 0.60	98.03 ± 0.70
DeiT-B	99.34 ± 0.14	99.25 ± 0.19	99.18 ± 0.21
Swin-B	99.12 ± 0.41	98.81 ± 0.27	99.17 ± 0.24

The images generated using bicubic and bilinear interpolation result in the highest accuracy values in every model as displayed in Table 2.8. The two techniques give quite similar results. The images generated using the nearest neighbour image resizing method gives lower accuracy, since this technique causes discontinuities in images.

2.4.2 Effect of Domain of Data

Secondly, we observe the effect of domain of data in three different forms as raw data, spectrogram data, and wavelet spectrogram data. We have used the magnitude resulting from taking the STFT of the sensor data while creating spectrogram images based on motion sensor data. To create the wavelet spectrogram images, we apply DWT to sensory data.



(a) (b)

Figure 2.7: Images generated by using the (a) spectrogram data and the (b) wavelet spectrogram data.

While generating the image in Fig. 2.7(a), we use the spectrogram data which we calculate based on the DSA Dataset. We create the second image given in Fig. 2.7(b) with the wavelet spectrogram data. The sensor readings employed in both parts of Fig. 2.7 belong to User 1 of the DSA Dataset.

2.4.2.1 Accuracy Scores of Images Generated from Different Domains

Table 2.9: Accuracy scores (%) of different models fed by images generated from spectrogram and wavelet spectrogram data.

Type of data	**ResNet18**	**ResNet152**	**DenseNet201**	**ViT-B**	**DeiT-B**	**Swin-B**
spectrogram	97.72 ± 0.27	97.31 ± 0.36	97.39 ± 0.24	96.22 ± 1.04	98.19 ± 0.26	94.76 ± 1.09
wavelet spectrogram	97.50 ± 0.31	97.63 ± 0.27	97.24 ± 0.44	98.17 ± 0.77	98.85 ± 0.46	95.53 ± 1.75

The model DeiT-B gives the highest accuracy when the spectrogram data are used, as displayed in Table 2.9. We intend to see the difference of the accuracy and computation time of the model, caused as a result of using images produced by raw data and spectrogram data by comparing the values in Table 2.9 with the values in Table 2.10 which are based on raw data. Overall, a negligible amount of difference occurs between the performance of the models as a result of giving images based on raw data and spectrogram data as input. The raw data images yield better results in all models employed in this study.

The model DeiT-B also results in the highest accuracy by using the wavelet spectrogram data as we can observe in Table 2.9. The main purpose of creating wavelet spectrogram images is to investigate the performance of the pretrained model when images produced by spectrogram data and wavelet spectrogram data are given as input. The results achieved with all of the models are quite similar when compared with the ones that use spectrogram images. Out of the six models that we have considered and included in our experiments, four of them (ResNet152, ViT-B, DeiT-B, and Swin-B) result in higher accuracy when we provide wavelet spectrogram images as input. The wavelet spectrogram images perform better than spectrogram images, since DWT is usually more effective for our task as it can capture features at different scales and resolutions when it is compared with STFT.

2.4.2.2 Accuracy Scores of Images Generated with Different Methods

Table 2.10: Accuracy scores (%) of different models for the DSA Dataset (Method 1).

Model	**ResNet18**	**ResNet152**	**DenseNet201**	**ViT-B**	**DeiT-B**	**Swin-B**
	98.25 ± 0.31	98.25 ± 0.23	97.95 ± 0.44	98.30 ± 1.29	99.76 ± 0.37	96.20 ± 1.67

Table 2.11: Accuracy scores (%) of different models for the DSA Dataset (Method 2).

Model	**ResNet18**	**ResNet152**	**DenseNet201**	**ViT-B**	**DeiT-B**	**Swin-B**
	98.48 ± 0.22	98.80 ± 0.26	98.69 ± 0.59	98.67 ± 0.50	99.29 ± 0.32	95.88 ± 1.92

To investigate the effect of the image generation method on the accuracy results, we compare the results in Table 2.10 with those in Table 2.11. For each model, DeiT-B with Method 1 gives better results.

2.4.3 Results of the Novel Image Generation Method

We have developed a new image generation method (Method 3) in which, we have given acceleration raw time-series data to the red channel, acceleration spectrogram data to the green channel, and acceleration wavelet data to the blue channel. The resulting RGB images differ from those of the other two techniques (Method 1 and Method 2) that we have implemented in image generation, since these images carry information related to both raw data and frequency. Table 2.12 presents the results of this model.

Table 2.12: Accuracy scores (%) of different models for the DSA Dataset (Method 3).

Model	**ResNet18**	**ResNet152**	**DenseNet201**	**ViT-B**	**DeiT-B**	**Swin-B**
	96.63 ± 0.49	96.31 ± 0.31	96.46 ± 0.49	96.09 ± 0.98	97.55 ± 0.54	93.85 ± 1.85

To make a comparison between our feature fusion image generation technique and the other two methods that we have implemented, Tables 2.10, 2.11, and 2.12 should be compared. Our new technique demonstrates slightly lower performance compared to the other two techniques, with a maximum accuracy difference of 2.58% observed across the models.

In this chapter, we have presented our image generation approach and preprocessing techniques. We examined the impact of resizing techniques and domain of data on model performance and introduced a novel image generation method that integrates both time-domain and frequency-domain information.

Chapter 3

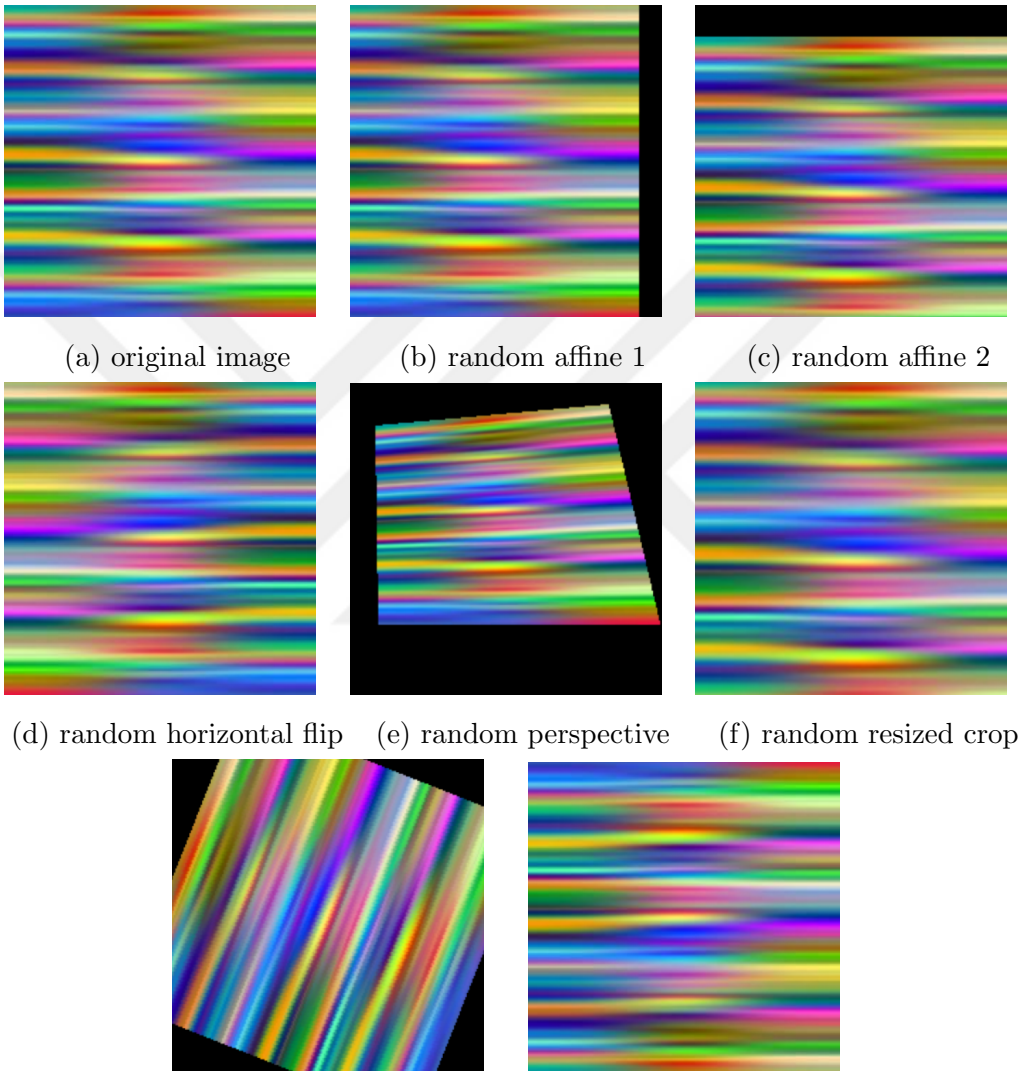
User Identification Based on Image Representations: A Novel Hybrid Architecture

In the previous chapter, we have investigated different image generation methods and compared them. In the present chapter, we introduce a novel hybrid network architecture that combines DeiT-B and DenseNet201 models in parallel form and investigate its performance.

3.1 Data Augmentation

DL models need huge amount of data to acquire satisfactory results. Augmenting the data is necessary to enhance the performance of the model as it enriches datasets by creating variations of the existing data [43]. This is done by applying diversified transformations to the original images. Since models work with different types of data as a result of data augmentation, they can generalize better. This technique reduces overfitting and improves the performance of the model. We have applied the following data augmentation techniques on original images (see Fig. 3.1): random

affine (1 and 2), random horizontal flip, random perspective, random resized crop, random rotation, and random vertical flip. In addition to these techniques, we have applied different variations of color jitter and Gaussian blur on the datasets.



(a) original image (b) random affine 1 (c) random affine 2

(d) random horizontal flip (e) random perspective (f) random resized crop

(g) random rotation (h) random vertical flip

Figure 3.1: The original image and its data augmented variations.

3.2 Experimental Results and Discussion

3.2.1 Effect of TL and Data Augmentation

Our aim is to observe the effect of data augmentation and TL. We have implemented four cases on the DSA Dataset as (i) augmentation and pretrained weights are used, (ii) only augmentation is employed, (iii) only pretrained weights are used, and (iv) neither augmentation nor pretrained weights are employed.

3.2.1.1 Accuracy Scores of TL and Data Augmentation

Table 3.1: Effect of data augmentation and TL in processing the DSA Dataset on the accuracy (%).

use of augmentation and pretrained weights	**number of images**	**DenseNet201**
augmentation and pretrained weights	6240	97.95 ± 0.44
augmentation	6240	97.80 ± 0.30
pretrained weights	480	80.62 ± 2.99
neither	480	79.37 ± 2.98

In Table 3.1, we identify that, the highest accuracy is achieved when both augmentation and pretrained weights are employed. Implementing pretrained weights is more important than applying augmentation, since the case which includes employing data augmentation [case (ii)] gives better accuracy than the case which includes utilizing pretrained weights [case (iii)]. By adopting pretrained weights, the model starts with the features which it has already learned. These weights provide transfer of useful knowledge. However, in the absence of data augmentation as a result of insufficient data, the model may potentially result in lower performance. On the other hand, data augmentation increases size and variability of data, which helps prevent overfitting. However, the augmentation techniques may produce distortions which can cause the model not to learn the task. In the case when neither of them are used [case (iv)], the accuracy is the lowest as expected.

3.2.2 Accuracy versus Epoch and Loss versus Epoch Graphs, and Training Time Tables

In this experiment, we have observed and compared the accuracy scores of the pretrained models for different HAR datasets. Figs. 3.2–3.9 indicate the performances of the DL models we have considered by showing the accuracy versus epoch and the loss versus epoch graphs for each of the four datasets. In addition to that, we have measured the training times of the models for the four datasets that we have considered in Tables 3.2–3.5. These tables display the (i) total training time and the (ii) training time per image.

3.2.2.1 DSA Dataset

Figs. 3.2 and 3.3 show the accuracy versus epoch and loss versus epoch plots for the pretrained models in processing the DSA Dataset. For the DSA Dataset, we have acquired the highest accuracy by using the model DeiT-B and the lowest accuracy by using the model Swin-B. Referring to Table 3.2, ResNet18 takes the least amount of time to train, whereas we observe that Swin-B takes the longest time to train.

Table 3.2: Total training time of CNN-based and transformer-based models in processing the DSA Dataset (4445 images).

Model Type	**Models**	**total training time (s)**	**training time per image (ms)**
CNN-based	ResNet18	164.36	36.97
	ResNet152	209.85	47.20
	DenseNet201	366.11	83.34
transformer-based	ViT-B	250.37	56.31
	DeiT-B	230.81	51.91
	Swin-B	400.19	90.01

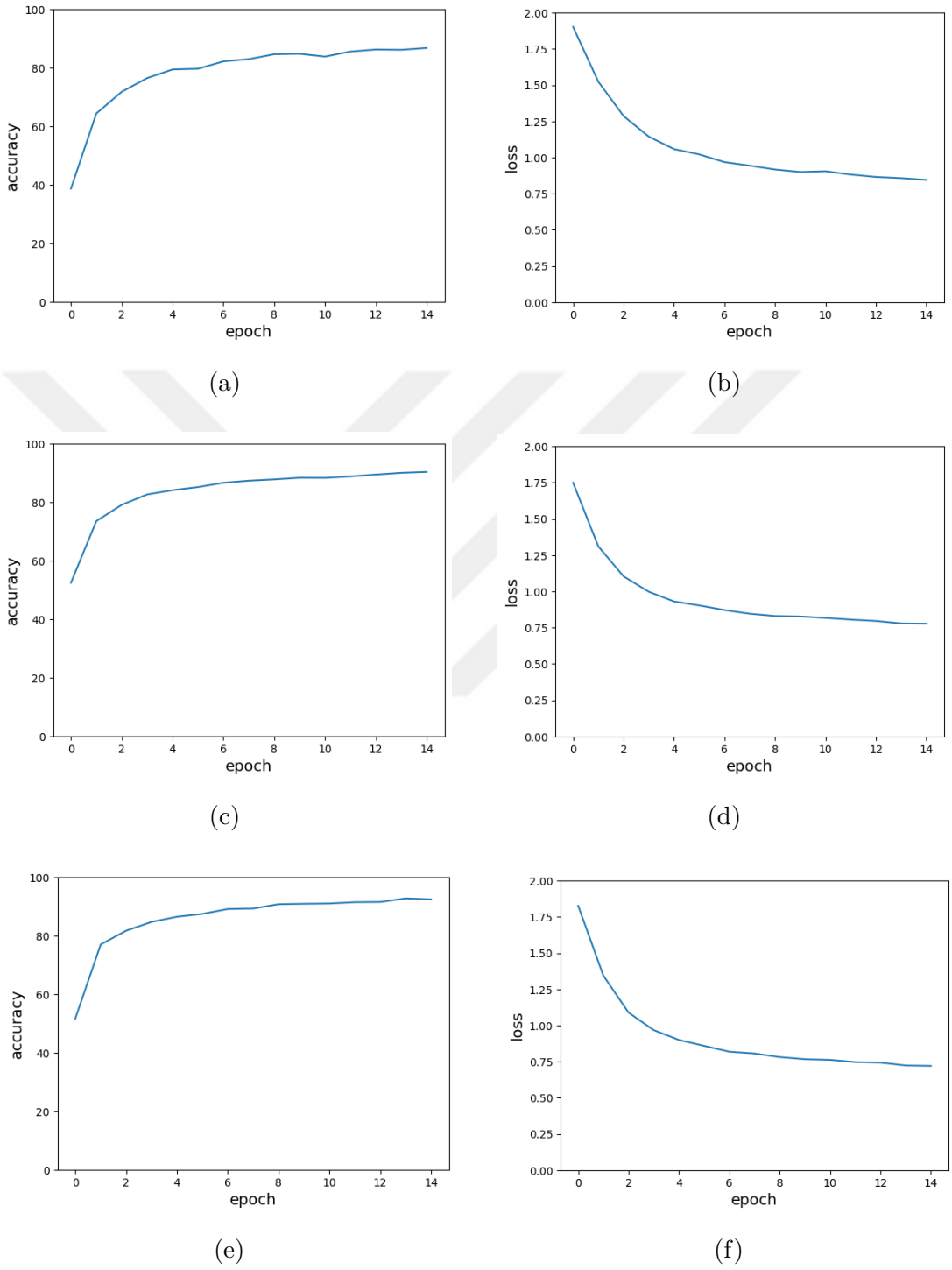


Figure 3.2: Accuracy and train loss of the CNN-based models for the DSA Dataset: (a)-(b) ResNet18, (c)-(d) ResNet152, and (e)-(f) DenseNet201.

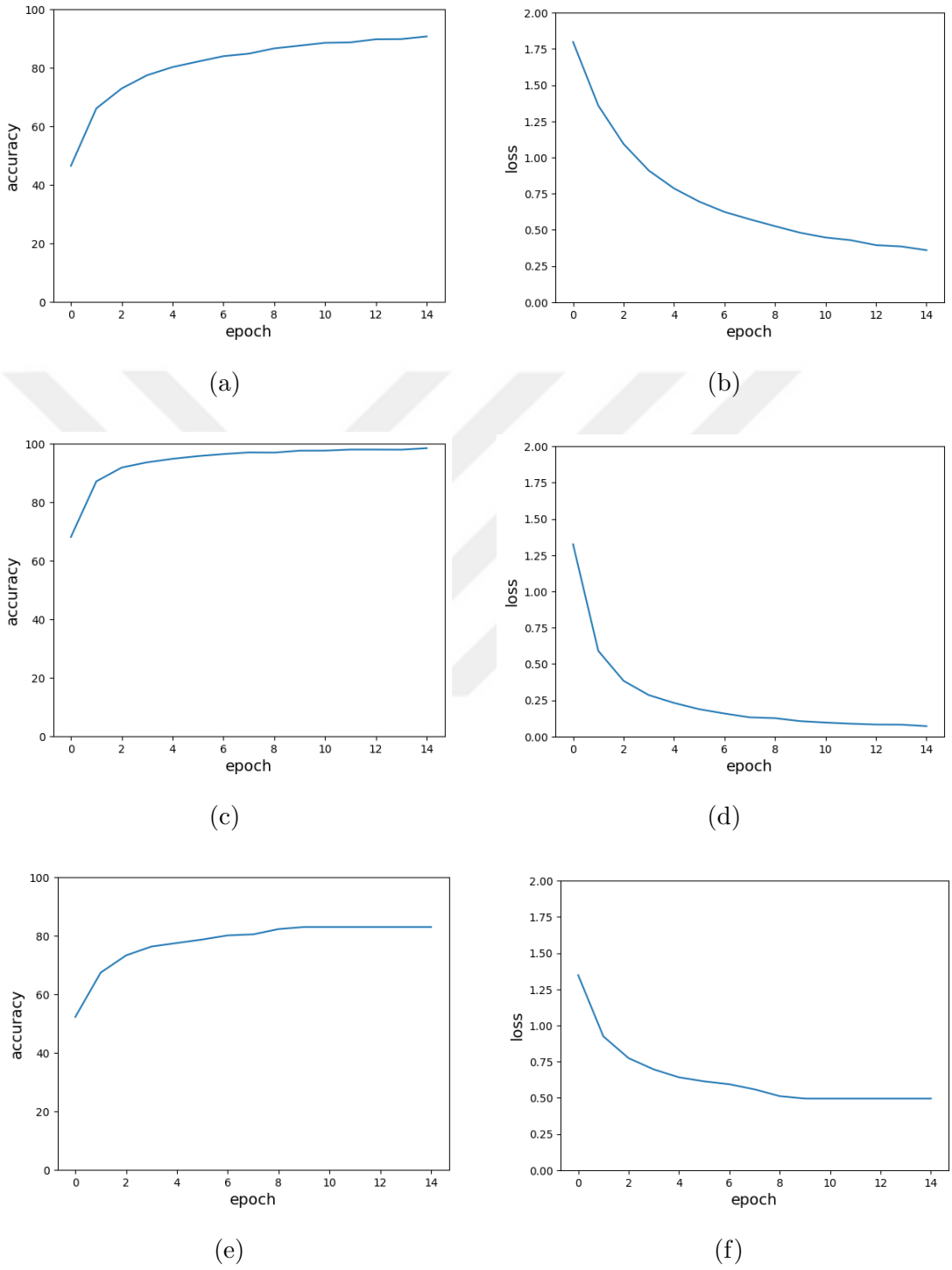


Figure 3.3: Accuracy and train loss of the transformer-based models for the DSA Dataset: (a)-(b) ViT-B, (c)-(d) DeiT-B, and (e)-(f) Swin-B.

3.2.2.2 USC-HAD Dataset

We illustrate the performance metrics of the models attained by using USC-HAD Dataset in Figs. 3.4 and 3.5. For this dataset, the model DeiT-B achieves better accuracy and the model ViT-B results in lower accuracy. We observe in Table 3.3 that ResNet18 takes the least and Swin-B takes the longest amount of time to train.

Table 3.3: Total training time of CNN-based and transformer-based models in processing the USC-HAD Dataset (3556 images).

Model Type	**Models**	**total training time (s)**	**training time per image (ms)**
CNN-based	ResNet18	111.92	31.47
	ResNet152	158.97	44.70
	DenseNet201	152.29	42.82
transformer-based	ViT-B	179.27	50.41
	DeiT-B	165.24	46.47
	Swin-B	272.12	76.52

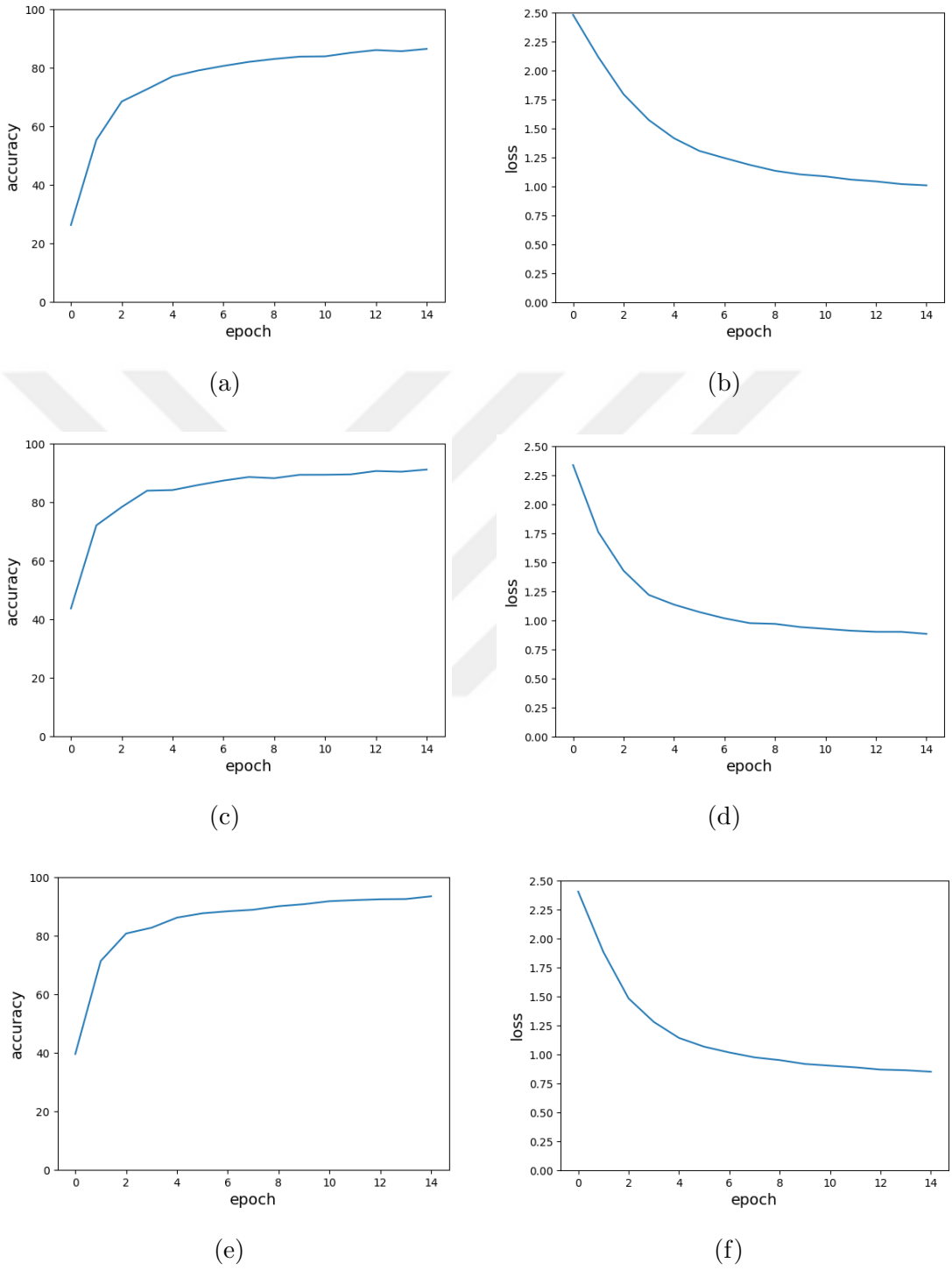


Figure 3.4: Accuracy and train loss of the CNN-based models for the USC-HAD Dataset: (a)-(b) ResNet18, (c)-(d) ResNet152, and (e)-(f) DenseNet201.

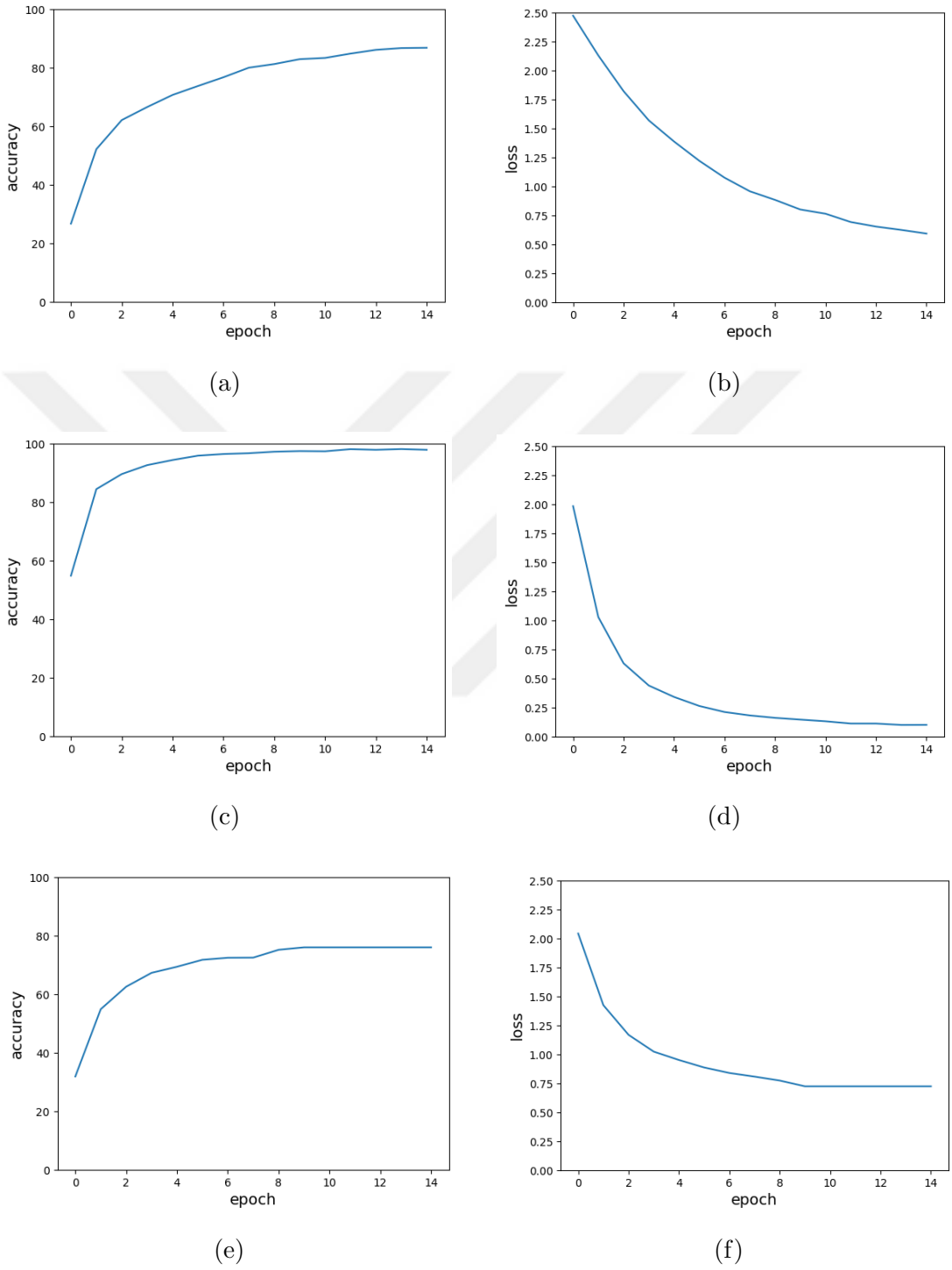


Figure 3.5: Accuracy and train loss of the transformer-based models for the USC-HAD Dataset: (a)-(b) ViT-B, (c)-(d) DeiT-B, and (e)-(f) Swin-B.

3.2.2.3 MotionSense Dataset

For the MotionSense Dataset, the DenseNet201 model obtains the highest accuracy, whereas the Swin-B model obtains the lowest, as depicted in Figs. 3.6 and 3.7. In addition to its least favorable performance, once again Swin-B takes the longest time to train, ResNet18 takes the shortest, as displayed in Table 3.4.

Table 3.4: Total training time of CNN-based and transformer-based models in processing the MotionSense Dataset (14206 images).

Model Type	**Models**	**total training time (s)**	**training time per image (ms)**
CNN-based	ResNet18	452.17	31.83
	ResNet152	603.38	42.47
	DenseNet201	637.26	44.86
transformer-based	ViT-B	714.57	50.30
	DeiT-B	643.32	45.28
	Swin-B	740.95	52.16

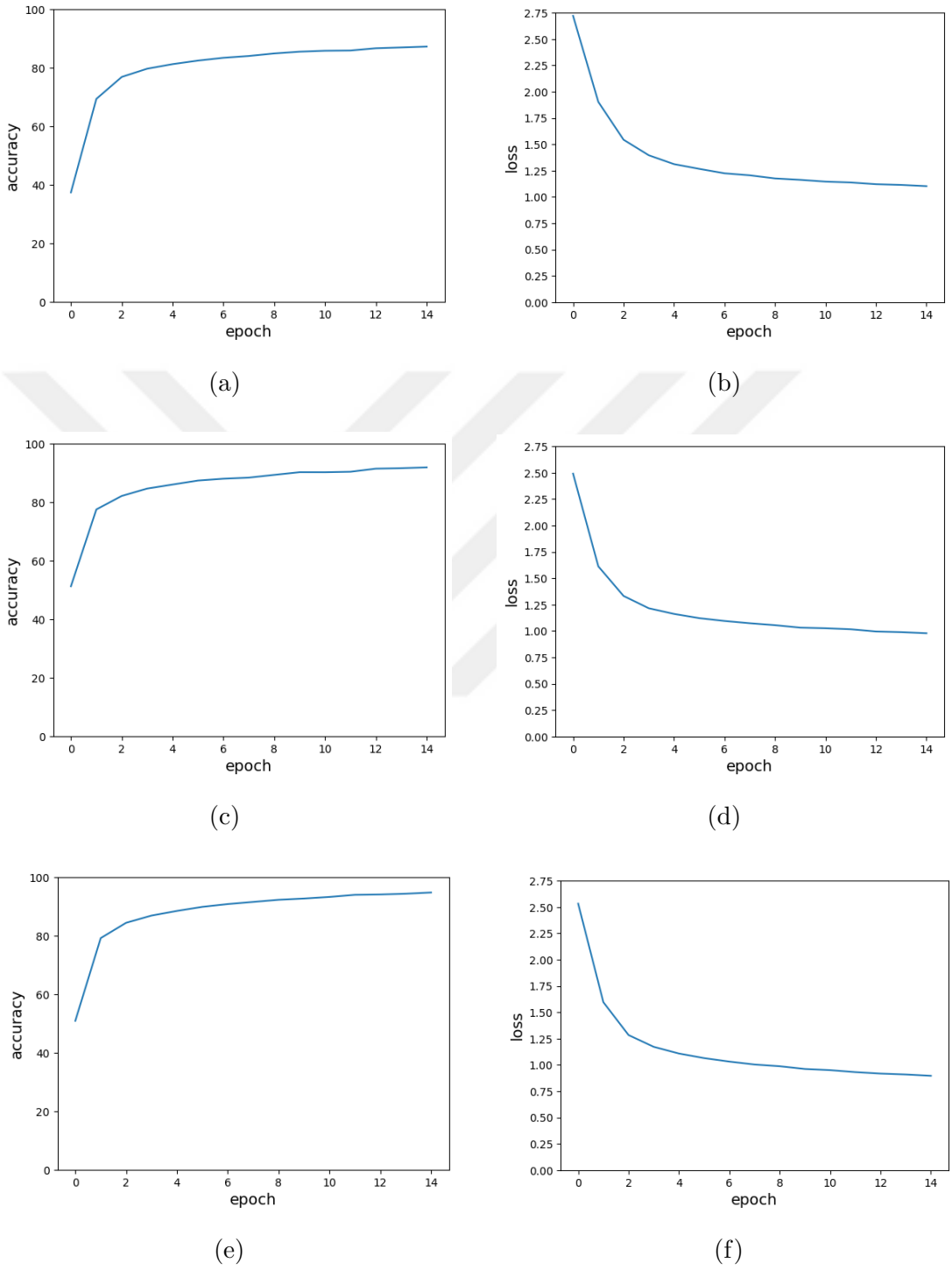


Figure 3.6: Accuracy and train loss of the CNN-based models for the MotionSense Dataset: (a)-(b) ResNet18, (c)-(d) ResNet152, and (e)-(f) DenseNet201.

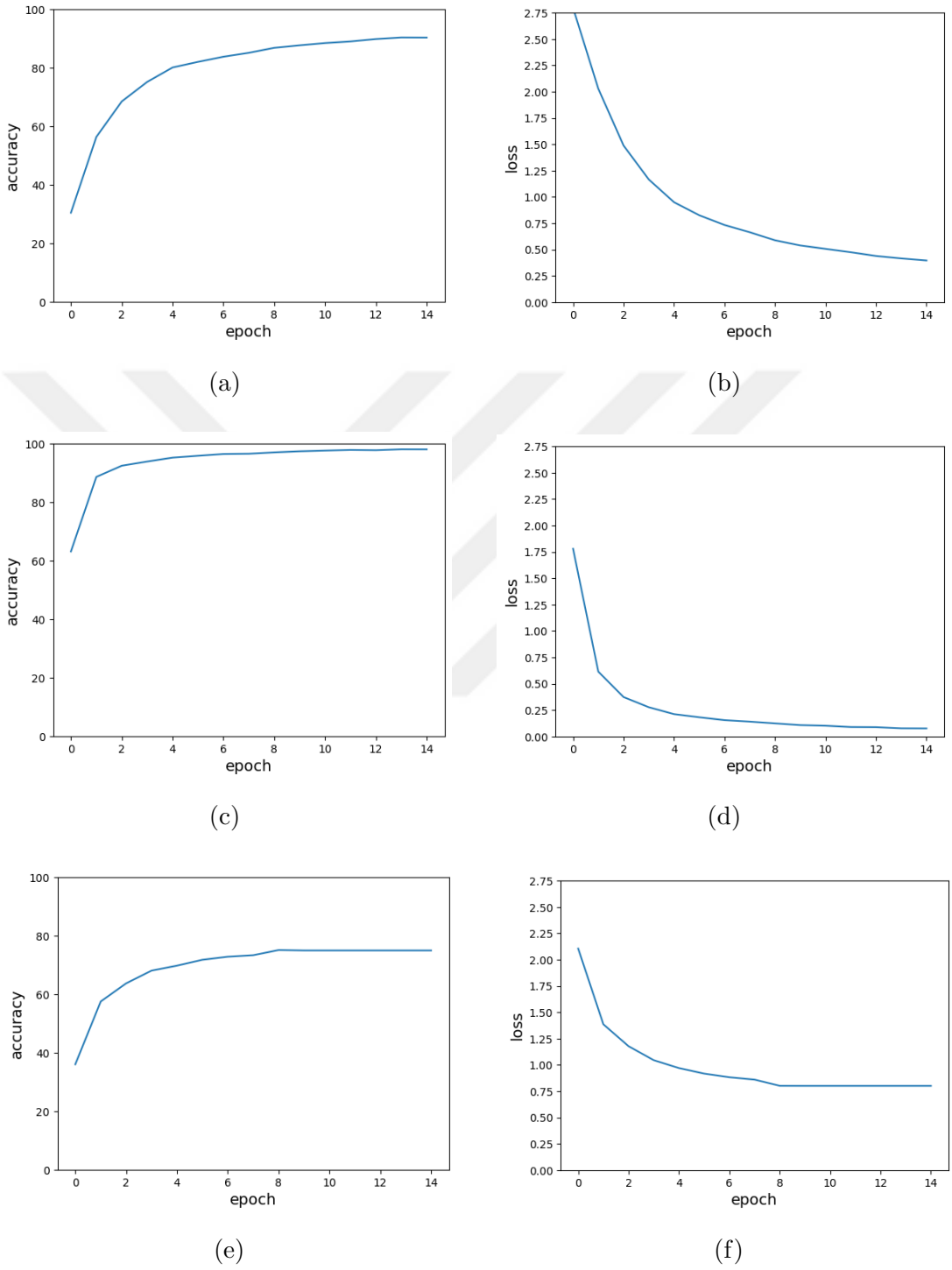


Figure 3.7: Accuracy and train loss of the transformer-based models for the MotionSense Dataset: (a)-(b) ViT-B, (c)-(d) DeiT-B, and (e)-(f) Swin-B.

3.2.2.4 UCI-HAR Dataset

Figs. 3.8 and 3.9 show that DeiT-B attains the peak accuracy among all of the models in procesing the UCI-HAR Dataset. The Swin-B model attains the lowest accuracy. When we inspect the results in Table 3.5, the Swin-B model requires the longest training time, while ResNet18 requires the shortest.

Table 3.5: Total training time of CNN-based and transformer-based models in processing the UCI-HAR Dataset (17955 images).

Model Type	**Models**	**total training time (s)**	**training time per image (ms)**
CNN-based	ResNet18	546.01	30.41
	ResNet152	737.41	41.07
	DenseNet201	779.60	43.42
transformer-based	ViT-B	876.22	48.80
	DeiT-B	791.22	44.06
	Swin-B	907.75	50.55

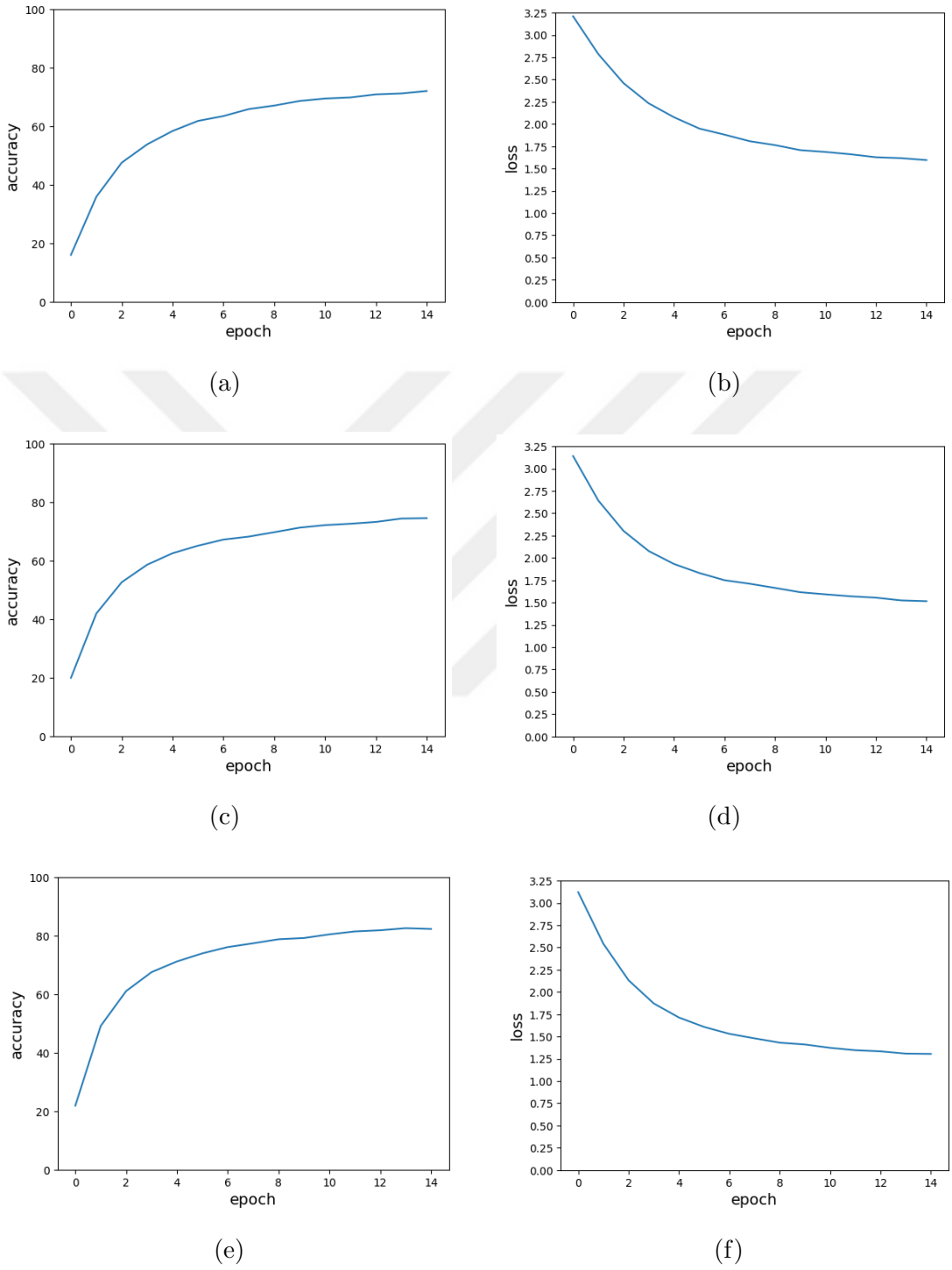


Figure 3.8: Accuracy and train loss of the CNN-based models for the UCI-HAR Dataset: (a)-(b) ResNet18, (c)-(d) ResNet152, and (e)-(f) DenseNet201.

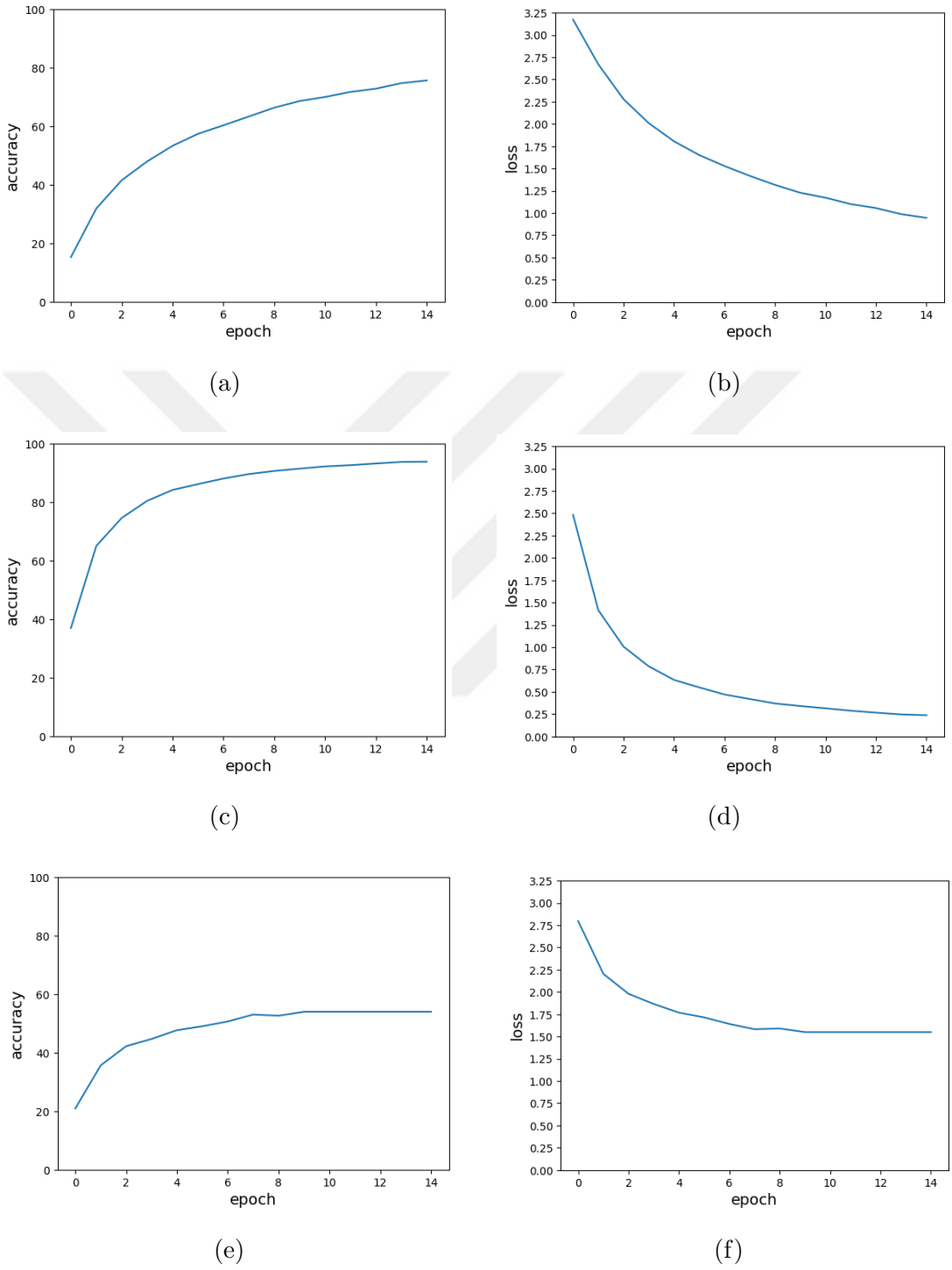
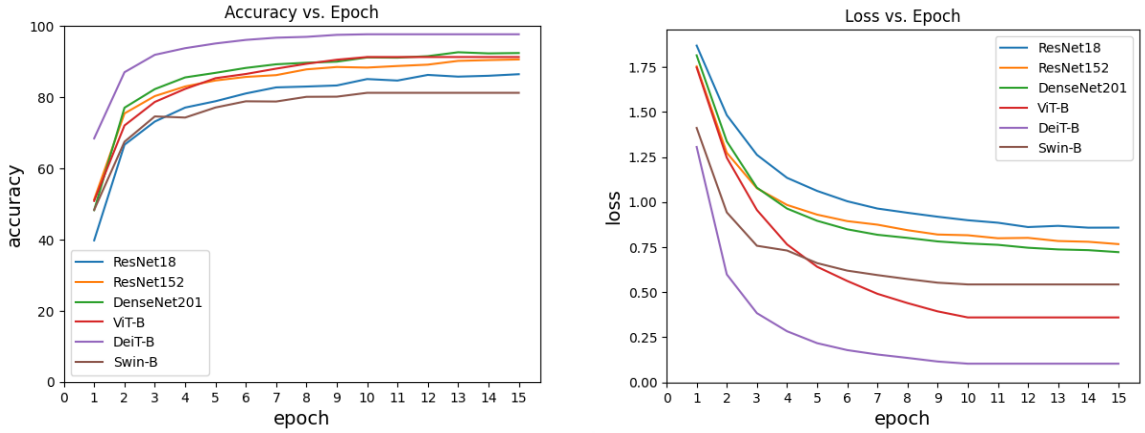


Figure 3.9: Accuracy and train loss of the transformer-based models for the UCI-HAR Dataset: (a)-(b) ViT-B, (c)-(d) DeiT-B, and (e)-(f) Swin-B.



(a) Accuracy of all six models.

(b) Loss of all six models.

Figure 3.10: Accuracy and train loss of all the models in processing the DSA Dataset.

Overall, for all four datasets, DenseNet201 and DeiT-B models yield the highest accuracies and the Swin-B model results in the lowest. We can also observe this fact in Fig. 3.10 which summarizes the results of the six models over the DSA Dataset. Among all models, Swin-B requires the longest training time, whereas ResNet18 is the shortest to train.

The effect of depth of the model to the accuracy results can be observed by comparing the results of the models ResNet18 and ResNet152. In all of the four datasets that we have considered and processed, ResNet152 gives higher accuracy compared to ResNet18. This result is expected since deeper neural networks have better capacity to learn complex patterns.

Table 3.6: Training time per image (ms) of the different models for the four datasets.

Model	**DSA**	**USC-HAD**	**MotionSense**	**UCI-HAR**
no. of images (augmented)	4445	3556	14206	17955
ResNet18	36.97	31.47	31.83	30.41
ResNet152	47.20	44.70	42.47	41.07
DenseNet201	83.34	42.82	44.86	43.42
ViT-B	56.31	50.41	50.30	48.80
DeiT-B	51.91	46.47	45.28	44.06
Swin-B	90.01	76.52	52.16	50.55

As we can see in Table 3.6 which summarizes the training time per image results, for all four datasets, ResNet18 model requires the least, and Swin-B takes the longest training time. Given that the accuracy of Swin-B is also the lowest, we can conclude that it is not a preferred model.

We acquire the accuracy values presented in Figs. 3.2–3.10 after applying hold-out cross-validation, whereas we obtain the accuracy values in Tables 3.7–3.10 through k-fold cross validation to better measure the model's performance. The difference between two evaluation results is due to the difference in the validation methods. Based on the accuracy values in Tables 3.7–3.10, by using the images generated from each dataset in our study, the success of the metrics of DeiT-B in processing all of the datasets identifies it as the most favorable, while the model Swin-B results in the lowest accuracy. The DeiT-B in processing the DSA Dataset, obtains the highest accuracy of 99.76%. The performance on the MotionSense Dataset is quite similar. The testing times of the models vary between 30–90 ms per image. In terms of accuracy, precision, recall, and f_1 score (Tables 3.7–3.10) all six models perform better in processing the MotionSense Dataset.

Table 3.7: Accuracy, precision, recall, and f_1 score (%) metrics of different models for the DSA Dataset (Method 1).

Model	**Accuracy**	**Precision**	**Recall**	f_1 **Score**
ResNet18	98.25 ± 0.31	98.26 ± 0.31	98.25 ± 0.31	98.25 ± 0.31
ResNet152	98.25 ± 0.23	98.27 ± 0.23	98.25 ± 0.23	98.25 ± 0.23
DenseNet201	97.95 ± 0.44	97.96 ± 0.44	97.95 ± 0.44	97.95 ± 0.44
ViT-B	98.30 ± 1.29	98.38 ± 1.10	98.27 ± 1.36	98.29 ± 1.30
DeiT-B	99.76 ± 0.37	99.78 ± 0.38	99.78 ± 0.34	99.77 ± 0.37
Swin-B	96.20 ± 1.67	96.40 ± 1.52	96.19 ± 1.68	96.22 ± 1.68

Table 3.8: Accuracy, precision, recall, and f_1 score (%) metrics of different models for the USC-HAD Dataset (Method 1).

Model	**Accuracy**	**Precision**	**Recall**	f_1 **Score**
ResNet18	96.25 ± 2.72	96.34 ± 2.71	96.22 ± 2.73	96.25 ± 2.72
ResNet152	97.73 ± 0.63	97.76 ± 0.63	97.67 ± 0.69	97.69 ± 0.67
DenseNet201	97.93 ± 0.48	97.97 ± 0.43	97.91 ± 0.40	97.92 ± 0.42
ViT-B	98.49 ± 0.36	98.54 ± 0.36	98.43 ± 0.39	98.47 ± 0.38
DeiT-B	99.28 ± 0.17	97.99 ± 2.53	97.98 ± 2.58	97.98 ± 2.55
Swin-B	95.70 ± 0.80	95.78 ± 0.90	95.67 ± 0.92	95.62 ± 0.91

Table 3.9: Accuracy, precision, recall, and f_1 score (%) metrics of different models for the MotionSense Dataset (Method 1).

Model	**Accuracy**	**Precision**	**Recall**	f_1 **Score**
ResNet18	99.69 ± 0.08	99.69 ± 0.08	99.70 ± 0.07	99.69 ± 0.07
ResNet152	99.67 ± 0.12	99.67 ± 0.13	99.66 ± 0.12	99.66 ± 0.12
DenseNet201	99.68 ± 0.12	99.68 ± 0.12	99.66 ± 0.11	99.67 ± 0.12
ViT-B	98.73 ± 0.46	98.72 ± 0.52	98.77 ± 0.45	98.74 ± 0.49
DeiT-B	99.69 ± 0.16	99.70 ± 0.13	99.68 ± 0.19	99.68 ± 0.16
Swin-B	95.37 ± 0.90	95.43 ± 0.83	95.37 ± 0.95	95.31 ± 0.92

Table 3.10: Accuracy, precision, recall, and f_1 score (%) metrics of different models for the UCI-HAR Dataset (Method 1).

Model	**Accuracy**	**Precision**	**Recall**	f_1 **Score**
ResNet18	98.04 ± 0.16	98.02 ± 0.18	97.96 ± 0.16	97.98 ± 0.17
ResNet152	98.39 ± 0.16	98.38 ± 0.15	98.35 ± 0.17	98.36 ± 0.16
DenseNet201	98.24 ± 0.11	98.24 ± 0.14	98.18 ± 0.13	98.20 ± 0.13
ViT-B	95.32 ± 1.20	95.47 ± 1.18	95.27 ± 1.18	95.30 ± 1.23
DeiT-B	99.00 ± 0.26	99.00 ± 0.23	98.97 ± 0.27	98.98 ± 0.25
Swin-B	85.66 ± 2.62	86.55 ± 2.72	85.42 ± 2.61	85.49 ± 2.78

3.2.3 Adapted Open-set User Identification Method

For consistency of our image generation technique in the UIR task, we adopt the open set UIR framework proposed by Benegui and Ionescu [20] on the four datasets (Fig. 3.11). Their framework includes both image generation for classification and the UIR task. They produce gray-scale images and use a custom CNN followed by a SVM as their model. Among the models that we have considered, we choose the DeiT-B as the feature extractor since it acquires the highest accuracy in our closed-set

experiments. Unlike the closed-set UIR task which we implement in this work where both train and test sets include data from all users, in open-set identification, we split the dataset into disjoint subsets of known and unseen users. Specifically, we use six train and two unseen users in the DSA Dataset, 10 train and four unseen users in the USC-HAD Dataset, 17 train and seven unseen users in the MotionSense Dataset, and 21 train and nine unseen users in the UCI-HAR Dataset. We train the DeiT-B model on the training set and extract the classification token (CLS) embeddings. We compute user centroids for the training users. We replicate the idea in Benegui and Ionescu's work for each user in the test set and build a binary SVM. We train a radial basis function (RBF) kernel SVM per unseen user. For each classifier, we interpret the embeddings of the unseen users as positive samples, while we take the embeddings of the other users as negatives. For testing, we pass each unseen user through the feature extractor. We feed CLS token embeddings to all binary SVMs. If a classifier recognizes the user, we assign that label. We calculate and tabulate the evaluation metrics for these unseen users (Table 3.11). Overall, the accuracy ranges from 89% to 97%. We observe the lowest performance in processing the DSA Dataset, likely due to its relatively small number of users, which increases intra-class variability and makes it hard to detect unseen users. The model performs superior on the USC-HAD Dataset as a result of moderate number of users. In contrast, for larger datasets such as MotionSense and UCI-HAR, the results are still high; however, the recall values show the difficulty of this open-set method. This is likely due to the increased number of classes and greater intra-class variability, which makes it harder for the model to generalize to unknown users.

Table 3.11: Accuracy, precision, recall, and f_1 score (%) metrics of the open-set method for the four datasets.

dataset	**Accuracy**	**Precision**	**Recall**	f_1 **Score**
DSA	89.17	91.10	89.17	89.04
USC-HAD	96.97	97.32	97.00	96.99
MotionSense	92.20	93.84	92.99	92.45
UCI-HAR	90.59	91.57	90.56	90.48

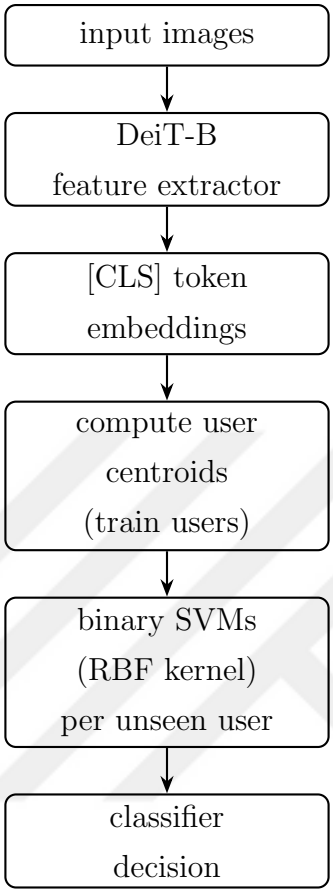


Figure 3.11: Open-set user UIR method.

3.2.4 A New Hybrid Structure

We attempt a new hybrid architecture which combines the DeiT-B and DenseNet201 models, since these two models attained high accuracy values individually; therefore it is more likely to obtain a better accuracy by using them in a hybrid structure in parallel form. Although hybrid models that integrate CNNs and transformers have been explored, a parallel combination of DeiT-B and DenseNet201 has not been considered to date. While DeiT-B deals with global relationships, DenseNet201 captures fine-grained, local details. These characteristics make this hybrid architecture both suitable to global and local feature extraction tasks. In this hybrid model, DeiT-B

extracts global features whereas DenseNet201 extracts local features. Once the features are extracted, we concatenate these features and pass them through a linear layer (Fig. 3.12). In this experiment, the DSA Dataset is processed.

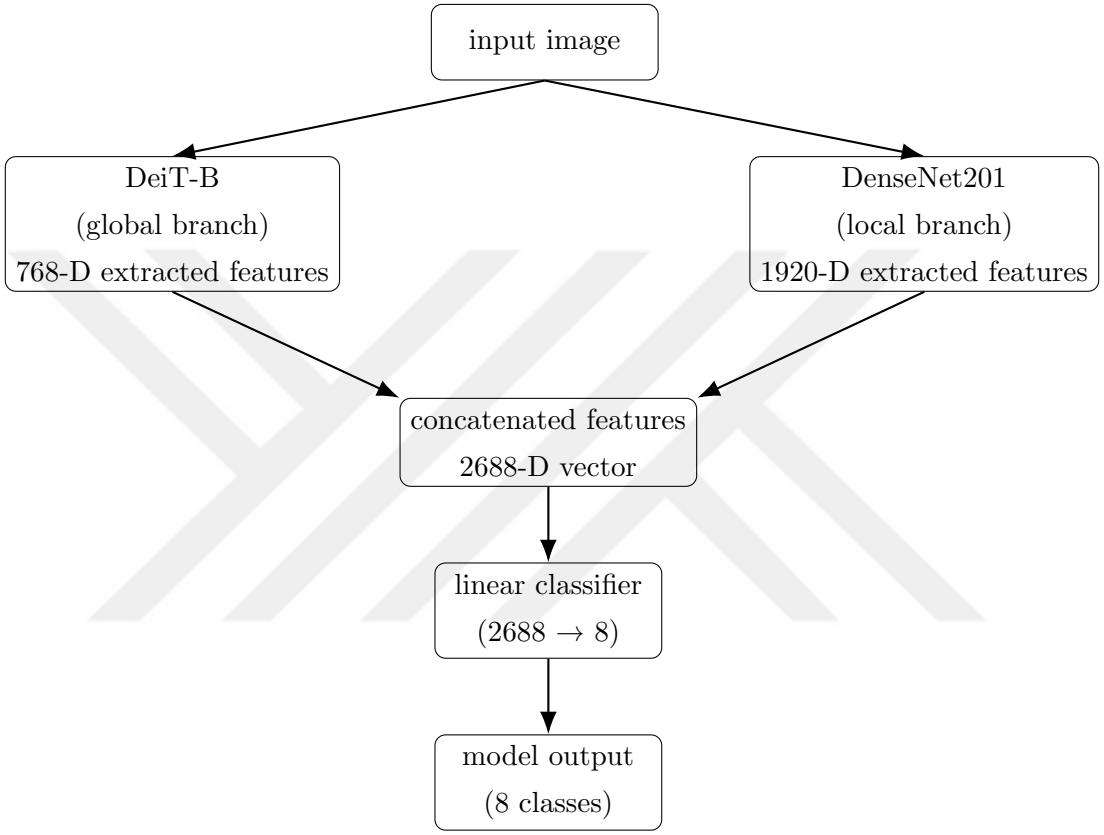


Figure 3.12: Block diagram of the proposed parallel hybrid architecture combining DeiT-B and DenseNet201 models.

3.2.4.1 Performance Comparison of the Hybrid Model with the 2D CNN Model

In this part, we compare the performance of our proposed hybrid model with that of 2D CNN. This is mainly because the 2D CNN model ranked the second-best in our earlier work [44]. Inspecting Fig. 3.13 indicates that our hybrid model exhibits superior performance compared to the 2D CNN model. Again, we obtain the accuracy values in Fig. 3.13 using hold-out cross validation; conversely, we acquire those in

Figs. 3.14, 3.15, and Table 3.12 employing k-fold cross validation. Table 3.12 displays the four performance metrics of our proposed hybrid model and 2D CNN on the DSA Dataset. Our hybrid model performs better than 2D CNN by about 0.5% in all four performance metrics.

The total training time of our proposed hybrid model lasts 2040.3 seconds as shown in Table 3.13. This training time is shorter than that of the 2D CNN model.

Table 3.12: Performance metrics (%) of the hybrid and 2D CNN models in processing the DSA Dataset.

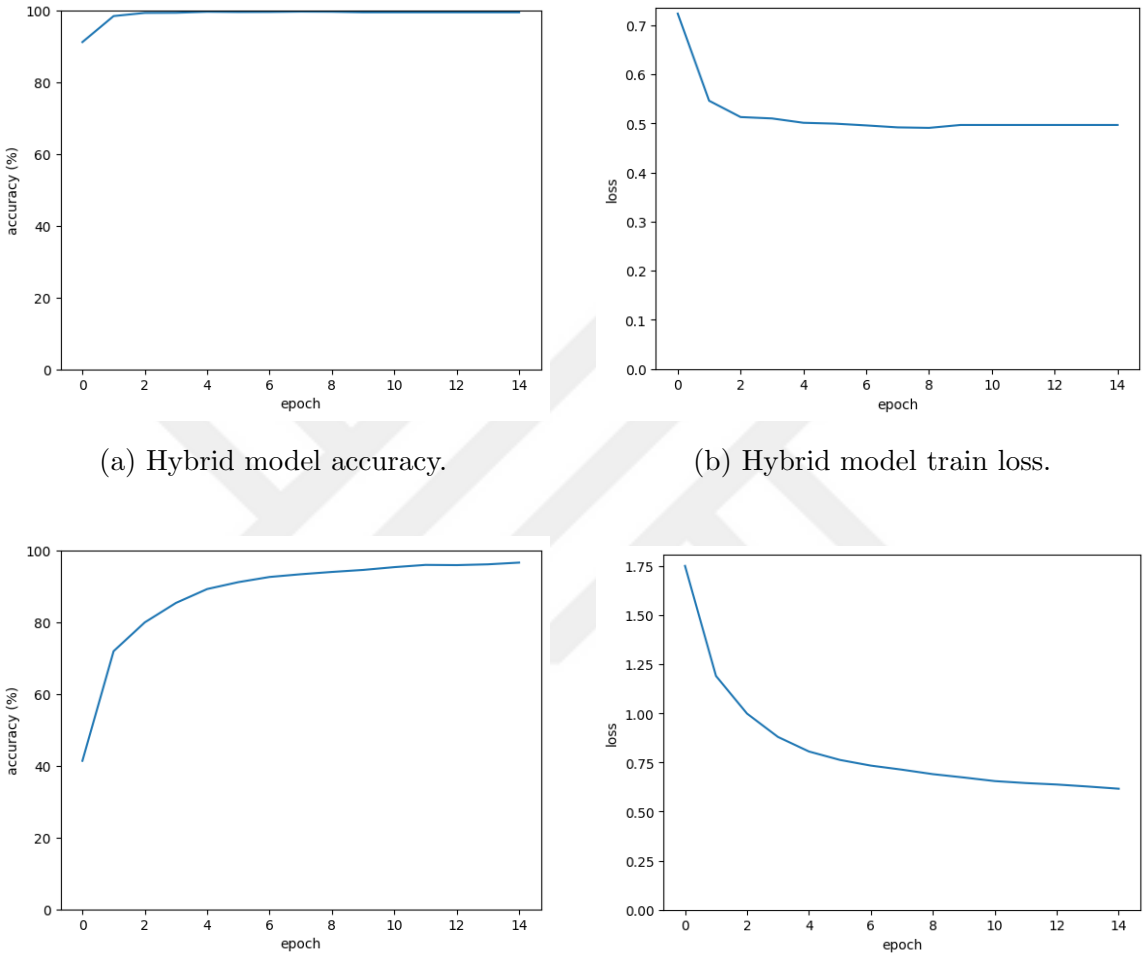
Model	**Accuracy**	**Precision**	**Recall**	f_1 **Score**
hybrid model	99.86 ± 0.33	99.86 ± 0.31	99.87 ± 0.32	99.86 ± 0.32
2D CNN	99.36 ± 0.11	99.37 ± 0.12	99.36 ± 0.12	99.36 ± 0.12

Table 3.13: Training time comparison of the hybrid and 2D CNN models in processing the DSA Dataset.

Model	**total training time (s)**	**training time per image (ms)**
hybrid model	2040.3	0.008174
2D CNN	2060.4	0.008255

Statistical Significance of the Comparison Results: To assess the statistical significance of the comparison results, we compare our hybrid model with the 2D CNN model by performing a statistical significance test on the DSA Dataset. To conduct this test, we use the accuracy values in Fig. 3.14. After applying a paired t-test [45], we obtain a highly significant difference between the models ($t = 6.626$, $p < 0.00001$), which proves that our hybrid model performs better. We perform the Wilcoxon test [46] to show whether this superiority could be a result of a random variation or not. The results ($W = 0$, $p = 0.00042$) also confirm that the observed performance is not due to a random variation. For consistency, we apply the paired t-test and Wilcoxon test on the UCI-HAR Dataset as well (Fig 3.15). The result for paired

t-test ($t = 9.410$, $p < 0.001$) indicates that our model performs better than the 2D CNN model. Also the Wilcoxon test ($W = 0$, $p = 0.00063$) supports this conclusion.



(a) Hybrid model accuracy.

(b) Hybrid model train loss.

(c) 2D CNN accuracy.

(d) 2D CNN train loss.

Figure 3.13: Performance of the hybrid and 2D CNN models on the DSA Dataset: (a) hybrid accuracy, (b) hybrid train loss, (c) 2D CNN accuracy, and (d) 2D CNN train loss.

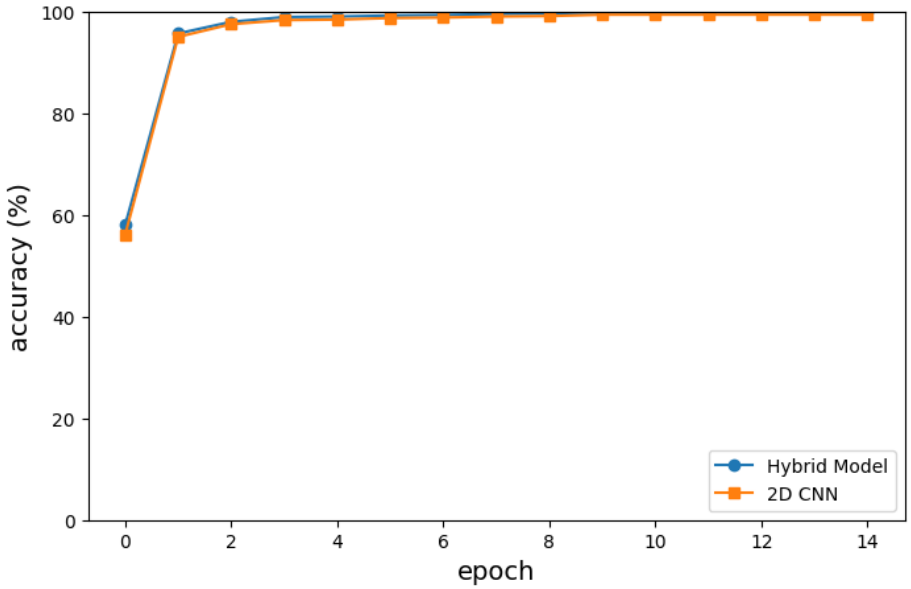


Figure 3.14: Accuracy of the hybrid and 2D CNN models on the DSA Dataset.

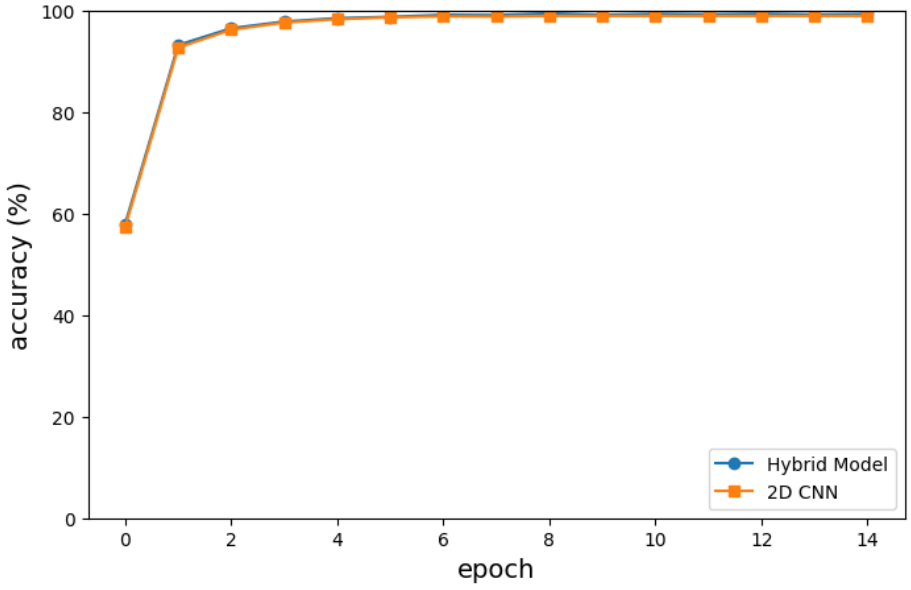


Figure 3.15: Accuracy of the hybrid and 2D CNN models on the UCI-HAR Dataset.

3.2.5 Comparison of Our Results with Existing Studies

For making a comparison between our model and the recent studies, we choose eight recent works [11], [12], [14], [16], [17], [18], [19], [20]. The works that we consider differ in their image generation techniques, but they may also vary in task type (e.g., HAR, UIR, user verification), dataset selection, and model architecture. Quin et al. [11] propose a hybrid architecture as it combines CNN and RNN for managing the UIR task. The accuracy is 96%. In Mekruksavanich and Jitpattanakul's work [12], their CNN-LSTM model gives 91.776% accuracy on the UCI-HAR Dataset and 92.432% accuracy on the USC-HAD Dataset. Jafari et al. [14] create images by converting time-series data into 2D matrix form and address the physical activity classification task by employing CNN, achieving an accuracy of 98%. Jiang and Yin [16], generate activity images in SI form and apply a new heterogeneous two-stream CNN model. This work also adopts the UCI-HAR Dataset and reports an accuracy of 97.90%. However, in Tables 3.7–3.10, we observe that, by using ResNet18, ResNet152, and DenseNet201 models, the UCI-HAR Dataset achieves higher accuracy. This result is expected since pretrained models generally perform superior. We evaluate our novel method on the DSA Dataset, therefore a direct comparison with [16] is not possible. Xu et al. [17] produce images by the help of GAF. Again, the only similarity is the use of the USC-HAD Dataset. The authors of the study adopt a Fusion-MdkResNet network which results in 89.48% accuracy. Ahmad and Khan [18] utilize ResNet18, which is relevant to our work and they also add CCF. They create images by adopting SI, GAF, MTF, and RP and their model achieves 98.3% accuracy. Bakshi [19] adopts ViT for the fall-type classification task and generates spectrogram images by applying STFT. The two publicly available datasets they have employed result in 99.9% and 99.8% accuracy. Even though the related studies are not identical and not directly comparable, they have some properties in common and as presented in Table 3.14, our hybrid model achieves the highest accuracy in the UIR task.

Study	**Task**	**Model**	**Image Generation**	**Accuracy (%)**
Qin et al. [11]	UIR	CNN + RNN	nowhere (raw time-series)	96
Mekruksavanich &Jitpattanakul [12]	UIR	CNN + LSTM	nowhere (raw time-series)	91.776 UCI-HAR - 92.432 USC-HAD
Jafari et al. [14]	HAR	SensorNet CNN	2D signals	98
Jiang & Yin [16]	HAR	Two-Stream CNN	SI images	97.90 UCI-HAR
Xu et al. [17]	HAR	Fusion-MdkResNet	GAF	89.48 USC-HAD
Ahmad & Khan [18]	HAR	ResNet18 + CCF	SI, GAF, MTF, RP	98.3
Bakshi [19]	Fall-type classification	ViT	spectrogram (STFT)	99.9 / 99.8
Benegui & Ionescu [20]	UIR	Pretrained CNN + SVM	gray-scale images	96.7
Proposed hybrid model	UIR	DeiT-B + DenseNet201	Method 1	**99.86 ± 0.33**

Table 3.14: Comparison of related studies in terms of task, model, image generation technique, and performance.

In this chapter, we have discussed the effect of TL and data augmentation on DL model performance. We evaluated the results based upon the CNN and transformer-based models. Furthermore, we have proposed a novel hybrid model structure that leverages the strengths of both DeiT-B and DenseNet201.

Chapter 4

Summary, Conclusions, and Future Work

In this thesis, we have conducted a comparative analysis on the performance of the CNN-based and transformer-based pretrained networks, in the image classification task by using four publicly available datasets which are DSA, USC-HAD, MotionSense, and UCI-HAR. The first dataset was acquired by our research group and the remaining three are benchmarking datasets. The considered pretrained models are: ResNet18, ResNet152, DenseNet201, ViT-B, DeiT-B, and Swin-B. In the first chapter, we have conducted a literature review addressing user identification with motion sensor data, image representation of motion sensor data, and transfer learning for user identification. In the second chapter, we first focus on two approaches that involve grouping datasets by sensor type (Method 1) and axis type (Method 2) to create the RGB components of the image. Then, we have proposed a novel image generation technique (Method 3) that fuses raw data, spectrogram, and wavelet spectrogram information. Besides this, we have made a comparison on image generation techniques. The benefit of this method is that it carries both time-domain and frequency-domain information. Another contribution of the thesis is creating spectrograms as input to the DL models. We have created spectrograms by using STFT and DWT. The wavelet spectrogram images affect the performance of the models more

than the spectrogram images.

In the third chapter, we evaluated the six models on four publicly available datasets and four performance metrics. DeiT-B and DenseNet-201 prove superior to the other models. We have proposed a new parallel hybrid model which combines DeiT-B and DenseNet201. To evaluate our proposed model's performance, we have trained each pretrained model on each dataset utilized in this work. DeiT-B model, in processing the DSA Dataset, shows better performance when compared with the other models; however, our new hybrid architecture performs superior. We designed four experimental scenarios to examine the effects of pretrained models and data augmentation. We observe that the use of data augmentation has more impact on the model's performance when compared with pretrained models.

The present study focused on comparing the performance of the pretrained models with different types of image generation methods. In future work, alternative image generation techniques and different model architectures can be investigated instead of employing only pretrained networks. During the evaluation phase of our hybrid model, we also trained the 2D CNN model using the DSA Dataset. As a potential direction for future research, we may connect our hybrid model with a 2D CNN. One can further investigate different input types for the RGB channels of the images.

Bibliography

[1] E. Davarci and E. Anarim, "User identification on smartphones with motion sensors and touching behaviors," in *Proc. 30th IEEE Signal Processing and Communications Applications Conference (SIU)*. Safranbolu, Turkey, 15–18 May 2022, https://doi.org/10.1109/siu55565.2022.9864837.

[2] F. Luo, S. Khan, Y. Huang, and K. Wu, "Activity-based person identification using multimodal wearable sensor data," *IEEE Internet of Things Journal*, vol. 10, no. 2, pp. 1711–1723. January 2023, https://doi.org/10.1109/jiot.2022.3209084.

[3] S. Mekruksavanich, P. Jantawong, and A. Jitpattanakul, "A deep learning-based model for human activity recognition using biosensors embedded into a smart knee bandage," *Procedia Computer Science*, vol. 214, pp. 621–627. *Proc. 9th International Conference on Information Technology and Quantitative Management*. Guangzhou, China, December 2022, https://doi.org/10.1016/j.procs.2022.11.220.

[4] A. S. Guinea, S. Heinrich, and M. Mühlhäuser, "VIDENS: Vision-based user identification from inertial sensors," in *Proc. ACM International Symposium on Wearable Computers*, 21–26 September 2021. pp. 153–155, virtual conference: ACM. http://tubiblio.ulb.tu-darmstadt.de/131727/.

[5] A. Sanchez Guinea, M. Sarabchian, and M. Mühlhäuser, "Improving wearable-based activity recognition using image representations," *Sensors*, vol. 22, no. 5. Art. no. 1840, February 2022, https://dx.doi.org/10.3390/s22051840.

[6] C. Han, L. Zhang, Y. Tang, W. Huang, F. Min, and J. He, "Human activity recognition using wearable sensors by heterogeneous convolutional neural networks," *Expert Systems with Applications*, vol. 198. Art. no. 116764, March 2022, https://www.sciencedirect.com/science/article/pii/S0957417422002299.

[7] A. Raval and M. Anwar, "Passive user identification and authentication with smartphone sensor data," in *Proc. Third International Conference on Transdisciplinary AI (TransAI)*. U.S.A., 20–22 September 2021, https://dx.doi.org/10.1109/TransAI51903.2021.00009.

[8] W. Shi, J. Yang, Y. Jiang, F. Yang, and Y. Xiong, "SenGuard: Passive user identification on smartphones using multiple sensors," in *Proc. 7th IEEE on Wireless and Mobile Computing, Networking and Communications*. pp. 141–148, Wuhan, China, 10–12 October 2011, https://infoscience.epfl.ch/handle/20.500.14299/72584.

[9] D.-Y. Kim, S.-H. Lee, and G.-M. Jeong, "Stack LSTM-based user identification using smart shoes with accelerometer data," *Sensors*, vol. 21, no. 23. Art. no. 8129, September 2021, https://doi.org/10.3390/s21238129.

[10] P. Fernandez-Lopez, J. Liu-Jimenez, K. Kiyokawa, Y. Wu, and R. Sanchez-Reillo, "Recurrent neural network for inertial gait user recognition in smartphones," *Sensors*, vol. 19, no. 18. Art. no. 4054, September 2019, https://doi.org/10.3390/s19184054.

[11] Z. Qin, L. Hu, N. Zhang, D. Chen, K. Zhang, Z. Qin, and K.-K. R. Choo, "Learning-aided user identification using smartphone sensors for smart homes," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7760–7772. February 2019, https://doi.org/10.1109/JIOT.2019.2900862.

[12] 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. Art. no. 308, January 2021, https://doi.org/10.3390/electronics10030308 .

[13] M. E. ul Haq, M. A. Azam, J. K.-K. Loo, K. Shuang, S. Islam, U. Naeem, and Y. Amin, "Authentication of smartphone users based on activity recognition and mobile sensing," *Sensors*, vol. 17. Art. no. 2043, September 2017, https://doi.org/10.3390/s17092043.

[14] A. Jafari, A. Ganesan, C. S. K. Thalisetty, V. Sivasubramanian, T. Oates, and T. Mohsenin, "SensorNet: A scalable and low-power deep convolutional neural network for multimodal data classification," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 1, pp. 274–287. January 2019, https://doi.org/10.1109/TCSI.2018.2848647.

[15] Z. Wang and T. Oates, "Encoding time series as images for visual inspection and classification using tiled convolutional neural networks," in *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*. Austin, TX, U.S.A., January 2015, https://api.semanticscholar.org/CorpusID:16409971.

[16] W. Jiang and Z. Yin, "Human activity recognition using wearable sensors by deep convolutional neural networks," *Proc. 23rd ACM on Multimedia*. New York, NY, U.S.A., 26–30 October 2015, https://doi.org/10.1145/2733373.2806333 .

[17] H. Xu, J. Li, H. Yuan, Q. Liu, S. Fan, T. Li, and X. Sun, "Human activity recognition based on Gramian angular field and deep convolutional neural network," *IEEE Access*, vol. 8, pp. 199393–199405. May 2020, https://doi.org/10.1109/ACCESS.2020.3032699.

[18] Z. Ahmad and N. Khan, "Inertial sensor data to image encoding for human action recognition." 2021, https://arxiv.org/abs/2105.13533.

[19] S. Bakshi, "Attention vision transformers for human fall detection," *Research Square*. Preprint, May 2022, https://doi.org/10.21203/rs.3.rs-1614908/v2.

[20] C. Benegui and R. T. Ionescu, "Convolutional neural networks for user identification based on motion sensors represented as images," *IEEE Access*, vol. 8, pp. 61255–61266, March 2020. https://doi.org/10.1109/ACCESS.2020.2984214.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*. pp. 770–778. Las Vegas, NV, U.S.A., 27–30 June 2016, https://doi.org/10.1109/CVPR.2016.90.

[22] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2017. pp. 2261–2269, Honolulu, HI, U.S.A., 21–26 July 2017, https://doi.org/10.1109/CVPR.2017.243.

[23] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," *Computing Research Repository*, 2020. https://arxiv.org/abs/2010.11929.

[24] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," *Computing Research Repository*, 2021. https://arxiv.org/abs/2012.12877.

[25] Z. L. et al., "Swin transformer: Hierarchical vision transformer using shifted windows," *arXiv preprint arXiv:2103.14030*, March 2021. https://arxiv.org/abs/2103.14030.

[26] B. Barshan and E. Koşar, "Bidirectional transfer learning between activity and user identity recognition via 2D CNN-LSTM model for wearables," *IEEE Internet of Things Journal*, Early Access, published online at IEEE Xplore, 10 June 2025. https://doi.org/10.1109/JIOT.2025.3577925.

[27] K. Altun and B. Barshan, "Daily and Sports Activities Data Set," IEEE Dataport, February 2019, https://dx.doi.org/10.21227/at1v-6f84.

[28] K. Altun and B. Barshan, "Daily and Sports Activities Data Set." UCI Machine Learning Repository. School Inf. Comput. Sci. California at Irvine, Irvine, CA, U.S.A., 2013. https://doi.org/10.24432/C5C59F.

[29] K. Altun, B. Barshan, and O. Tunçel, "Comparative study on classifying human activities with miniature inertial and magnetic sensors," *Pattern Recognition*, vol. 43, no. 10, pp. 3605–3620, October 2010. https://doi.org/10.1016/j.patcog.2010.04.019.

[30] M. Zhang and A. A. Sawchuk, "USC-HAD: A daily activity dataset for ubiquitous activity recognition using wearable sensors," in *Proc. ACM Conference on Ubiquitous Computing (UbiComp)*. Pittsburgh, PA, U.S.A., Association for Computing Machinery, pp. 1036–1043, September 2012, https://doi.org/10.1145/2370216.2370438.

[31] M. Malekzadeh, R. G. Clegg, A. Cavallaro, and H. Haddadi, "Mobile sensor data anonymization," in *Proc. Internet of Things Design and Implementation Conference (IoTDI)*. Montreal, Quebec, Canada, ACM, 15 April 2019, pp. 49–58, http://doi.acm.org/10.1145/3302505.3310068.

[32] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proc. The European Symposium on Artificial Neural Networks*. Bruges, Belgium, 24–26 April 2013, https://api.semanticscholar.org/CorpusID:6975432.

[33] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Upper Saddle River, NJ, U.S.A.: Prentice Hall, 3rd ed., 2009.

[34] S. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*. Burlington, MA, U.S.A.: Academic Press, 3rd ed., 2008.

[35] J. Rabbi, M. T. H. Fuad, and M. A. Awal, "Human activity analysis and recognition from smartphones using machine learning techniques," 2021. https://arxiv.org/abs/2103.16490.

[36] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, January, 2012. http://jmlr.org/papers/v13/bergstra12a.html.

[37] S. Paul, V. Kurin, and S. Whiteson, "Fast efficient hyperparameter tuning for policy gradient methods," in *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, (Vancouver, BC, Canada), pp. 4618–4628, December 2019. https://papers.neurips.cc/paper/8710-fast-efficient-hyperparameter-tuning-for-policy-gradient-methods.pdf.

[38] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2951–2959. Lake Tahoe, NV, U.S.A., December 2012, https://papers.nips.cc/paper/2012/hash/05311655a15b75fab86956663e1819cd-Abstract.html.

[39] A. Nugroho and H. Suhartanto, "Hyper-parameter tuning based on random search for Densenet optimization," in *Proc. 7th Information Technology, Computer, and Electrical Engineering (ICITACEE)*, pp. 96–99. Semarang, Indonesia, 24–25 September 2020, https://doi.org/110.1109/ICITACEE50144.2020.9239164.

[40] S. D. Team, "Spyder — The scientific Python development environment," https://www.spyder-ide.org.

[41] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, vol. 32, 2019. https://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[42] E. Bisong, "Google colaboratory," in *Proc. 2nd edition of the MobiCom Workshop on Mobile Big Data*. Los Cabos, Mexico, 21 October 2019, https://link.springer.com/chapter/10.1007/978-1-4842-4470-8_7.

[43] T. A. Korzhebin and A. D. Egorov, "Comparison of combinations of data augmentation methods and transfer learning strategies in image classification used in convolution deep neural networks," in *Proc. IEEE Russian Young Researchers in Electrical and Electronic Engineering (ElConRus)*. pp. 479–482, 26–29 January 2021, https://doi.org/10.1109/ElConRus51938.2021.9396724.

[44] E. Koşar and B. Barshan, "A new CNN-LSTM architecture for activity recognition employing wearable motion sensor data: Enabling diverse feature extraction," *Engineering Applications of Artificial Intelligence*. vol. 124, Art. no: 106529, September 2023, https://doi.org/10.1016/j.engappai.2023.106529.

[45] Student, "The probable error of a mean," *Biometrika*, vol. 6, no. 1, pp. 1–25, 1908.

[46] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.