applied sciences

MDPI

Article

Merging-Squeeze-Excitation Feature Fusion for Human Activity Recognition Using Wearable Sensors

Seksan Laitrakun

Sirindhorn International Institute of Technology, Thammasat University, Pathum Thani 12120, Thailand; seksan@siit.tu.ac.th

Abstract: Human activity recognition (HAR) has been applied to several advanced applications, especially when individuals may need to be monitored closely. This work focuses on HAR using wearable sensors attached to various locations of the user body. The data from each sensor may provide unequally discriminative information and, then, an effective fusion method is needed. In order to address this issue, inspired by the squeeze-and-excitation (SE) mechanism, we propose the merging-squeeze-excitation (MSE) feature fusion which emphasizes informative feature maps and suppresses ambiguous feature maps during fusion. The MSE feature fusion consists of three steps: pre-merging, squeeze-and-excitation, and post-merging. Unlike the SE mechanism, the set of feature maps from each branch will be recalibrated by using the channel weights also computed from the pre-merged feature maps. The calibrated feature maps from all branches are merged to obtain a set of channel-weighted and merged feature maps which will be used in the classification process. Additionally, a set of MSE feature fusion extensions is presented. In these proposed methods, three deep-learning models (LeNet5, AlexNet, and VGG16) are used as feature extractors and four merging methods (addition, maximum, minimum, and average) are applied as merging operations. The performances of the proposed methods are evaluated by classifying popular public datasets.

Keywords: convolutional neural networks (CNNs); deep learning; feature fusion; human activity recognition (HAR); inertial measurement units (IMUs); squeeze-and-excitation; wearable sensors

check for updates

Citation: Laitrakun, S. Merging-Squeeze-Excitation Feature Fusion for Human Activity Recognition Using Wearable Sensors. *Appl. Sci.* **2023**, *13*, 2475. https://doi.org/10.3390/app13042475

Academic Editors: Marc Kurz, Erik Sonnleitner and Clemens Holzmann

Received: 23 January 2023
Revised: 7 February 2023
Accepted: 13 February 2023
Published: 14 February 2023

1. Introduction

Human activity recognition (HAR) is an active and challenging research field [1] to specify human activities (e.g., sitting, walking, running) based on the data collected from devices such as cameras [2] and wearable sensors [3–5]. It has been essential in many applications, especially healthcare [6]. In addition, HAR helps an information–technology system to automatically monitor and record the activities of users such that we can analyze them and alert related persons (e.g., users, relatives, doctors) when an abnormal activity or an accident happens [7]. Due to the limitations of using cameras in HAR such as user privacy, using wearable devices (e.g., smart watches and smartphones) in HAR is receiving significant attention. These wearable devices commonly use sensors such as accelerometers, gyroscopes, and magnetometers to monitor the activities of the users [5,8,9]. In addition, many studies have been focused on using several inertial measurement units (IMUs) attached to different parts of the user body such that we can have data from different locations and utilize them together to obtain better recognition accuracy [10].

The HAR using wearable devices will receive sensor data from accelerometers, gyroscopes, and/or magnetometers and use them to classify the activities. Of the classification models/algorithms, two types are popularly applied to HAR: traditional machine learning (ML) algorithms and deep-learning (DL) models. By using a traditional ML algorithm (e.g., support vector machine, random forest), we will manually extract a set of useful features from the sensor data and pass them to the ML algorithm to specify the corresponding activities [11,12]. On the other hand, a DL model will automatically extract a set of features

from the sensor data and use them in the classification process. As a result, DL models such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been extensively studied in HAR [9,13–15].

A CNN model essentially consists of a series of convolutional layers and pooling layers to extract a set of features called feature maps, which will be used later in the classification. However, only some feature maps may be very useful for classifying the activities of interest. Therefore, informative feature maps should be emphasized while ambiguous feature maps should be suppressed. A channel–attention mechanism called the squeeze-and-excitation (SE) block [16] was proposed to solve this issue. The SE block will recalibrate each feature map by a weight value which is proportional to the importance of this feature map in the classification. The SE block has recently been applied to CNN and/or RNN models to better the HAR performances [17,18].

The performances of wearable-sensor HAR can be improved by implementing multi-branch DL architectures [19,20]. A multi-branch DL architecture consists of several parallel branches using DL models to extract different sets of feature maps independently. Specifically, each branch provides a set of feature maps denoting local information. Thereafter, these sets of feature maps will be fused by using feature fusion such as concatenation to generate a set of fused feature maps (denoting global information); which will be used later in the classification process. A traditional feature fusion method combines the local feature maps equally without being aware that, in each set, some feature maps may be informative while some feature maps are ambiguous. Motivated by this issue, we need a feature fusion method which is able to emphasize informative feature maps and suppress ambiguous feature maps in each branch during fusion such that we can combine the local feature maps efficiently and obtain useful discriminative fused feature maps.

Inspired by the squeeze-and-excitation (SE) mechanism [16], we propose a feature fusion method called the merging-squeeze-excitation (MSE) feature fusion. In each branch, the MSE feature fusion recalibrates the local feature maps by using a set of channel weights. Since the fused feature maps are the ones who enter the classification process, the channel weights could be computed such that the fused feature maps provide very discriminative information. Therefore, unlike the SE mechanism, we design the MSE feature fusion such that, at each branch, it computes the channel weights based on both local feature maps and fused feature maps. As a result, when we consider a set of C local feature maps, the c-th local feature map will be emphasized if either it is important to the classification or the corresponding c-th fused feature map is useful to the classification.

The MSE feature fusion consists of three steps: pre-merging step, squeeze-and-excitation step, and post-merging step. In the pre-merging step, the feature maps from all branches are merged together to obtain a set of pre-merged feature maps. Thereafter, during the squeeze-and-excitation step, the feature maps from each branch are recalibrated according to their importance measured from both the channel-wise statistics obtained from themselves and the channel-wise statistics obtained from the pre-merged feature maps. Finally, in the post-merging step, the MSE feature fusion applies the same merging operation used in the first step to combine the output feature maps from all branches and obtain a set of channel-weighted and merged feature maps which will be used to classify the activities of interest. In this work, we have applied three DL models (i.e., LeNet5, AlexNet, and VGG16) as feature extractors and four merging methods (addition, maximum, minimum, and average) as merging operations in the pre-merging step and post-merging step. Furthermore, we also modify the proposed MSE feature fusion by adding local skip connections, adding a global skip connection, using global channel attention, and stacking a series of the MSE feature fusions to create deep MSE feature fusions. Their performances are evaluated on three public HAR datasets: PAMAP2, DaLiAc, and DSAD.

The main contributions of this work are summarized as follows:

1. We propose five MSE feature fusion architectures for wearable-sensor HAR using multi-branch architectures such that the feature maps will be recalibrated according

to their importance during the fusion. Three DL models and four merging methods used in the MSE feature fusion are studied and investigated.

2. Extensive experiments are conducted to evaluate and compare the performances of the proposed methods and baseline architectures by using the PAMAP2, DaLiAc, and DSAD datasets. The results show the following findings:
   - The MSE feature fusion with a global skip connection when using the average merging and AlexNet achieves the highest accuracy score of 99.24% on classifying the PAMAP2 dataset.
   - The MSE feature fusion with local skip connections when using the minimum merging and AlexNet achieves the highest accuracy score of 98.59% on classifying the DaLiAc dataset.
   - The original MSE feature fusion using the average merging and AlexNet achieves the highest accuracy score of 98.04% on classifying the DSAD dataset.
   - Among the merging methods studied in the proposed methods, the addition merging offers the worst accuracy scores. The maximum, minimum, and average merging have similar performances.
   - All of the highest accuracy scores are from using AlexNet as the feature extractor.

The rest of this paper is organized as follows. Section 2 reviews the previous work focusing on using the SE block in HAR, proposing multi-branch DL architectures for HAR, and presenting SE-based feature fusion methods. The data collection and preparation are described in Section 3. Sections 4 and 5 present the proposed MSE feature fusion and its extensions, respectively. Their performances are evaluated and compared in Section 6. Finally, conclusions and future work are provided in Section 7.

The main symbols used in this paper are summarized as follows. Lower-case and upper-case bold letters represent vectors and three-dimensional (3D) arrays, respectively. The symbols $\mathbb{R}^{1\times C}$ and $\mathbb{R}^{C}$ denote the space of C-real-number row vectors and the space of C-real-number column vectors, respectively. The symbol $\mathbb{R}^{H\times W\times C}$ denotes the space of 3D arrays (of real numbers) whose height, width, and channel number are equal to H, W, and C, respectively. Tables 1 and 2 summarize the main symbols used in this paper.

Table 1. List of main symbols used in Sections 4, 5.1 and 5.2.

Symbol	Definition
$\mathbf{g}^{(n)} \in \mathbb{R}^{C}$	A vector of channel-wise statistics according to the local feature maps $\mathbf{A}^{(n)}$ at the n-th branch.
$\mathbf{h}^{(n)} \in \mathbb{R}^{C}$	A vector of channel-wise statistics according to the addition of $\mathbf{g}^{(n)}$ and $\mathbf{u}$ at the n-th branch.
$\mathbf{s}^{(n)} \in \mathbb{R}^{C}$	A vector of channel weights for the local feature maps $\mathbf{A}^{(n)}$ at the n-th branch.
$\mathbf{u} \in \mathbb{R}^{C}$	A vector of channel-wise statistics according to the pre-merged feature maps $\mathbf{B}$.
$\mathbf{v} \in \mathbb{R}^{C}$	A vector of channel weights for the pre-merged feature maps $\mathbf{B}$.
$\mathbf{A}^{(n)} \in \mathbb{R}^{1\times L\times C}$	A 3D array of local feature maps at the n-th branch.
$\mathbf{B} \in \mathbb{R}^{1\times L\times C}$	A 3D array of pre-merged feature maps.
$\mathbf{P}^{(n)} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted feature maps according to the local feature maps $\mathbf{A}^{(n)}$ at the n-th branch.
$\mathbf{Q}^{(n)} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted feature maps according to the addition of $\mathbf{P}^{(n)}$ and $\mathbf{B}$ at the n-th branch.
$\mathbf{R} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted feature maps according to the pre-merged feature maps $\mathbf{B}$.
$\mathbf{X}^{(n)} \in \mathbb{R}^{1\times W\times C}$	A 3D array of sensor data at the n-th branch (obtained from the n-th IMU).
$\mathbf{Y}^{gca} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted and merged feature maps, which is the output of the MSE feature fusion with global channel attention.
$\mathbf{Y}^{gsc} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted and merged feature maps, which is the output of the MSE feature fusion with a global skip connection.
$\mathbf{Y}^{lsc} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted and merged feature maps, which is the output of the MSE feature fusion with local skip connections.
$\mathbf{Y}^{mse} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted and merged feature maps, which is the output of the MSE feature fusion.

Table 2. List of main symbols used in Section 5.3.

Symbol	Definition
$\tilde{\mathbf{g}}^{(d,n)} \in \mathbb{R}^C$	A vector of channel-wise statistics according to the feature maps $\tilde{\mathbf{P}}^{(d-1,n)}$ at the n-th branch in the d-th MSE feature fusion block.
$\tilde{\mathbf{h}}^{(d,n)} \in \mathbb{R}^C$	A vector of channel-wise statistics according to the addition of $\tilde{\mathbf{g}}^{(d,n)}$ and $\tilde{\mathbf{u}}^{(d)}$ at the n-th branch in the d-th MSE feature fusion block.
$\tilde{\mathbf{s}}^{(d,n)} \in \mathbb{R}^C$	A vector of channel weights for the feature maps $\tilde{\mathbf{P}}^{(d,n)}$ at the n-th branch in the d-th MSE feature fusion block.
$\tilde{\mathbf{u}}^{(d)} \in \mathbb{R}^C$	A vector of channel-wise statistics according to the merged and channel-weighted feature maps $\tilde{\mathbf{Y}}^{deep,(d)}$ in the d-th MSE feature fusion block.
$\mathbf{A}^{(n)} \in \mathbb{R}^{1\times L\times C}$	A 3D array of local feature maps at the n-th branch.
$\mathbf{B} \in \mathbb{R}^{1\times L\times C}$	A 3D array of pre-merged feature maps.
$\tilde{\mathbf{P}}^{(d,n)} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted feature maps according to the feature maps $\tilde{\mathbf{P}}^{(d-1,n)}$ at the n-th branch in the d-th MSE feature fusion block.
$\mathbf{X}^{(n)} \in \mathbb{R}^{1\times W\times C}$	A 3D array of sensor data at the n-th branch (obtained from the n-th IMU).
$\tilde{\mathbf{Y}}^{mse,(d)} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted and merged feature maps, which is the output of the weighted feature merging in the d-th MSE feature fusion block.
$\tilde{\mathbf{Y}}^{deep,(d)} \in \mathbb{R}^{1\times L\times C}$	A 3D array of channel-weighted and merged feature maps, which is the output of the d-th MSE feature fusion.

2. Related Work

The SE block was proposed to improve the performances of the CNNs and to demonstrated its potential in image classification [16]. It consists of two successive operations: squeeze and excitation. In the squeeze operation, the inputted feature maps will be passed to a global-average pooling (GAP) layer to generate channel-wise statistics, where each value is an average of the corresponding feature map. Thereafter, in the excitation operation, the SE block will compute an appropriate weight for each feature map by using the channel-wise statistics and fully connected (FC) layers. The SE block multiplies inputted feature maps by their weights and obtains the channel-weighted feature maps. Due to its success in image classification, the SE block has been adopted in many applications, including HAR. Zhongkai et al. [17] investigated the potential of the SE blocks by adding them to a list of state-of-the-art CNN models (e.g., VGG16, Inception, ReNet18, and PyramidNet18) and comparing the corresponding HAR performances. Mekruksavanich et al. [18] proposed a DL model called the SEResNet-BiGRU, which is a combination of residual blocks, SE blocks, and bidirectional gate recurrent units (BiGRUs), and applied it for transitional activity recognition. Khan et al. [21] proposed a multi-branch DL architecture where each branch uses a CNN model with an SE block to extract and re-weight feature maps. The above DL models with SE blocks are summarized in Table 3.

Table 3. A summary of DL models with SE blocks for HAR using sensor data.

Year	Ref.	Dataset	Device	DL Model
2021	[21]	UCI HAR, WISDM	Smartphone	CNN with an SE block
2022	[17]	HASC, UCI HAR, WISDM	Smartphone	State-of-the-art CNNs with SE blocks
2022	[18]	HAPT, MobiAct v2.0	Smartphone	CNN with a residual block, an SE block, and BiGRU

Several DL architectures have been extensively proposed and investigated in HAR using sensor data. In order to improve the HAR performances, instead of using only one branch, we can implement DL architectures with multiple branches such that several different and unique sets of feature maps will be obtained and helpful in classifying the activities. There are two common categories of the multi-branch architectures. In the first category, we consider a scenario wherein there is a set of wearable sensors (e.g., IMUs) attached to parts of the user body (e.g., a wrist, an ankle, the chest). Therefore, different sets of sensor

data are obtained initially. These sets of sensor data are inputted into a multi-branch DL architecture. Each branch will receive each set and extract the corresponding features by using the same DL model independently [19,22,23]. In order to obtain a set of feature maps on each branch, Rueda et al. [19] applied a series of convolutional layers and max pooling layers, Liu et al. [22] implemented stacked convolutional layers, and Al-qaness et al. [23] employed a CNN model with residual blocks. The feature maps of all branches are fused (i.e., feature fusion) by using concatenation. In the second category, we apply a set of sensor data to a multi-branch DL architecture where each branch uses a different DL model and results in a different set of features. Three-branch DL architectures were proposed in [20,21,24–28], where a CNN model [20,24,27,28], a hybrid of a CNN model, and a bidirectional long short-term memory (LSTM) layer [25], as well as a CNN model with an SE block [21], and a hybrid of convolutional layers and gated recurrent unit (GRU) layers [26] were used on each branch to extract a set of features. The differences among these branches are the kernel sizes of the convolutional layers [20,21,24–28] and the number of layers [20]. Similarly, the output feature sets are combined by using concatenation. We summarize the aforementioned multi-branch DL architectures in Table 4.

Table 4. A summary of multi-branch DL architectures for HAR using sensor data.

Year	Ref.	Dataset	Device	Category	DL Model	Feature Fusion
2018	[19]	Opportunity, Order Picking, PAMAP2	IMUs	Multiple Inputs	CNN	Concatenation
2019	[20]	UCI HAR, WISDM	Smartphone	Multiple DL Models	CNN	Concatenation
2020	[22]	DG, DSAD, PAMAP2, RealWorld-HAR	IMUs	Multiple Inputs	CNN	Concatenation
2020	[24]	WISDM	Smartphone	Multiple DL Models	CNN	Concatenation
2020	[25]	MHEALTH, WISDM	IMUs, Smartphone	Multiple DL Models	CNN and LSTM	Concatenation
2021	[21]	UCI HAR, WISDM	Smartphone	Multiple DL Models	CNN with an SE block	Concatenation
2021	[26]	PAMAP2, UCI HAR, WISDM	IMUs, Smartphone	Multiple DL Models	CNN and GRU	Concatenation
2021	[27]	Self-Recorded Data, UCI HAR	IMUs, Smartphone	Multiple DL Models	CNN	Concatenation
2022	[28]	PAMAP2, UCI HAR, WISDM	IMUs, Smartphone	Multiple DL Models	CNN	Concatenation
2023	[23]	Opportunity, PAMAP2, UniMiB-SHAR	IMUs, Smartphone	Multiple Inputs	CNN and Residual Blocks	Concatenation

Recently, the SE mechanism (i.e., squeeze and excitation operations) has been applied to feature fusion in multi-branch DL architectures where feature maps from each branch will be recalibrated before fusing them together. Li et al. [29] proposed a model called the temporal-spectral-based squeeze-and-excitation feature fusion network (TS-SEFFNet) to classify motor imagery tasks by using electroencephalography (EEG) signals. The TS-SEFFNet receives EEG signals and uses two branches with different DL models called the deep-temporal convolution block and the multi-spectral convolution block to extract two different sets of feature maps. The feature maps of each set are recalibrated by using the SE mechanism. The TS-SEFFNet combines the outputs of these two branches by using concatenation.

Instead of using sensor data from one modality, multimodal classification [30] receives data from multiple modalities and has gained a significant amount of attention [31–33]. Essentially, a multi-modal classification model will be implemented based on a multi-branch architecture where each branch will receive different modal data and extract the corresponding features. These features obtained from various modalities will be combined and sent to the classification process. Since the feature maps from each modality contribute information unequally, an efficient fusion method must be investigated [31–33].

In addition, several SE-based feature fusion methods were extensively investigated in multi-modal classification. Jia et al. [34] proposed a feature fusion method called the

multi-modal SE feature fusion module to combine feature maps from EEG signals and feature maps from electrooculogram (EOG) signals for sleep-staging classification. Unlike [16,29] where each branch computes the weights for the feature-map calibration in the excitation operation separately, the multi-modal SE feature fusion module will calculate the channel weights based on the channel-wise statistics from both EEG feature maps and EOG feature maps.

Shu et al. [35] proposed a DL model called the expansion-squeeze-excitation fusion network (ESE–FN) for elderly activity recognition using RGB videos and skeleton sequences. The ESE–FN applies two successive fusion modules (modal fusion and channel fusion) to combine RGB features and skeleton features properly. The modal-fusion module performs the modal attention where modal-wise weights are computed and multiplied to the corresponding modalities' feature maps. The channel-fusion module obtains the channel attention by calculating channel-wise weights and multiplying them to the feature maps. Both modules apply a new attention mechanism called the expansion-squeeze-excitation, which consists of three operations: expansion, squeeze, and excitation. The expansion is operated by using convolutional layers to expand the depth along the modality dimension for the modal fusion and expand the depth along the channel dimension for the channel fusion. The squeeze and excitation operations are similar to those in [16]. A summary of the work [29,34,35] is shown in Table 5.

Table 5. A summary of related SE fusion.

Year	Ref.	Modality	Dataset	Classification	Fusion Mechanism
2021	[29]	EEG	BCI IV 2a, HGD	Motor imagery tasks	SE mechanism
2022	[34]	EEG, EOG	MASS-SS3	Sleep staging	Multimodal SE mechanism
2022	[35]	RGB videos, skeleton sequences	ETRI-Activity3D	Elderly activities	Expansion SE mechanism

3. Data Collection and Preparation

In order to evaluate the proposed HAR classification architectures in Sections 4 and 5, we select the datasets whose sensor data are from IMUs attached to various locations of the user body. Thereafter, we preprocess the sensor data by scaling and segmentation. The details are explained as follows. The sensor data are from the following three wearable-sensor datasets:

- PAMAP2: The PAMAP2 dataset [36] contains the sensor data collected from nine subjects who performed 18 physical activities. However, here, only 12 activities are considered: lying, sitting, standing, ironing, vacuum cleaning, descending stairs, walking, Nordic walking, cycling, ascending stairs, running, and rope jumping. Three IMUs were attached to a wrist, the chest, and an ankle of each subject. Each IMU was equipped with two triaxial accelerometers, one triaxial gyroscope, and one triaxial magnetometer. As a result, 12 types of sensor data ($M = 12$) were obtained from each IMU. The sampling rate was set to 100 Hz.
- DaLiAc: The DaLiAc dataset [37] contains the sensor data collected from 19 subjects who performed 13 physical activities: sitting, lying, standing, washing dishes, vacuuming, sweeping, walking, ascending stairs, descending stairs, treadmill running (8.3 km/h), bicycling on ergometer (50 Watt), bicycling on ergometer (100 Watt), and rope jumping. A total of four IMUs were attached to the right hip, the right wrist, the chest, and the left ankle. Each IMU was equipped with one triaxial accelerometer and one triaxial gyroscope. As a result, six types of sensor data ($M = 6$) were obtained from each IMU. The sampling rate was set to approximately 200 Hz.
- DSAD: The DSAD dataset [38] contains the sensor data collected from eight subjects who performed 19 physical activities: sitting, standing, lying on back, lying on right side, ascending stairs, descending stairs, standing in an elevator still, moving around in an elevator, walking in a parking lot, walking on a treadmill with a speed of 4 km/h

in flat, walking on a treadmill with a speed of 4 km/h at 15 degree inclined positions), running on a treadmill with a speed of 8 km/h, exercising on a stepper, exercising on a cross trainer, cycling on an exercise bike in horizontal position, cycling on an exercise bike in vertical positions, rowing, jumping, and playing basketball. Five IMUs were attached to the torso, right arm, left arm, right leg, and left leg. Each IMU was equipped with one triaxial accelerometer, one triaxial gyroscope, and one triaxial magnetometer. As a result, nine types of sensor data ($M = 9$) were obtained from each IMU. The sampling rate was set to 25 Hz.

The sensor data used to predict the current activity are obtained from different sensor types and varied within different ranges. It is a common step to apply the data scaling such that the values of these sensor data will be within the same range. In this work, the standardization method is applied to transform the sensor data such that their mean and standard deviation are zero and one, respectively. Let $z_{t,m}^{(n)}$ be the sensor value at the t-th point obtained from the m-th sensor data of the n-th IMU. Its standardized value is obtained from:

$$\tilde{z}_{t,m}^{(n)} = \frac{z_{t,m}^{(n)} - \mu_m^{(n)}}{\sigma_m^{(n)}}, \tag{1}$$

where $\mu_m^{(n)}$ and $\sigma_m^{(n)}$ are the mean and standard deviation, respectively, of the values from the m-th sensor data of the n-th IMU.

Next, a series of the standardized values $\tilde{z}_{t,m}^{(n)}$ is divided into segments by using a non-overlapping window method. Each segment consists of L values. Let $\mathbf{x}_m^{(n)} \in \mathbb{R}^{1\times L}$ be a segment of the standardized values from the m-th sensor data of the n-th IMU. The row vector $\mathbf{x}_m^{(n)}$ can be expressed as

$$\mathbf{x}_m^{(n)} = \left[\tilde{z}_{1,m}^{(n)}, \tilde{z}_{2,m}^{(n)}, \ldots, \tilde{z}_{L,m}^{(n)}\right]. \tag{2}$$

The length L is set to 300, 600, and 125 data points for PAMAP2, DaLiAc, and DSAD, respectively (which are equal to three-second window, three second window, and five-second window, respectively). A summary of the sensor data which will be used in the evaluation is shown in Table 6.

Table 6. A summary of sensor data from three datasets.

	PAMAP2	**DaLiAc**	**DSAD**
Sensor	2 accelerometers, 1 gyroscope, 1 magnetometer	1 accelerometer, 1 gyroscope	1 accelerometer, 1 gyroscope, 1 magnetometer
Sampling Rate	100 Hz	200 Hz	25 Hz
No. IMUs (N)	3	4	5
Positions	wrist, chest, ankle	right wrist, chest, right hip, left ankle	torso, right arm, left arm, right leg, left leg
No. Sensor Data Types per IMU (M)	12	6	9
No. Subjects	9	19	8
No. Activities	12	13	19
Window Size	3 s	3 s	5 s
Segment Length (L)	300 data points	600 data points	125 data points
No. Segments	5764	7802	9120

4. Proposed Architecture

The proposed architecture is shown in Figure 1, which is based on a multi-branch architecture. There are N branches to receive the inputs from N IMUs. The number N will be equal to 3, 4, and 5, for the PAMAP2, DaLiAc, and DSAD datasets, respectively, as shown in Table 6. Each branch receives the input $\mathbf{X}^{(n)}$ from an IMU and uses a one-

dimensional (1D) CNN model to extract a set of feature maps $\mathbf{A}^{(n)}$. Since each feature map owns different significance of information, we propose the merging-squeeze-excitation (MSE) feature fusion to combine these N sets of feature maps $\mathbf{A}^{(n)}$ by applying the SE mechanisms [16] and produce the channel-weighted and merged feature maps $\mathbf{Y}^{mse}$, which will be used to predict the corresponding activity. The details are provided as follows:

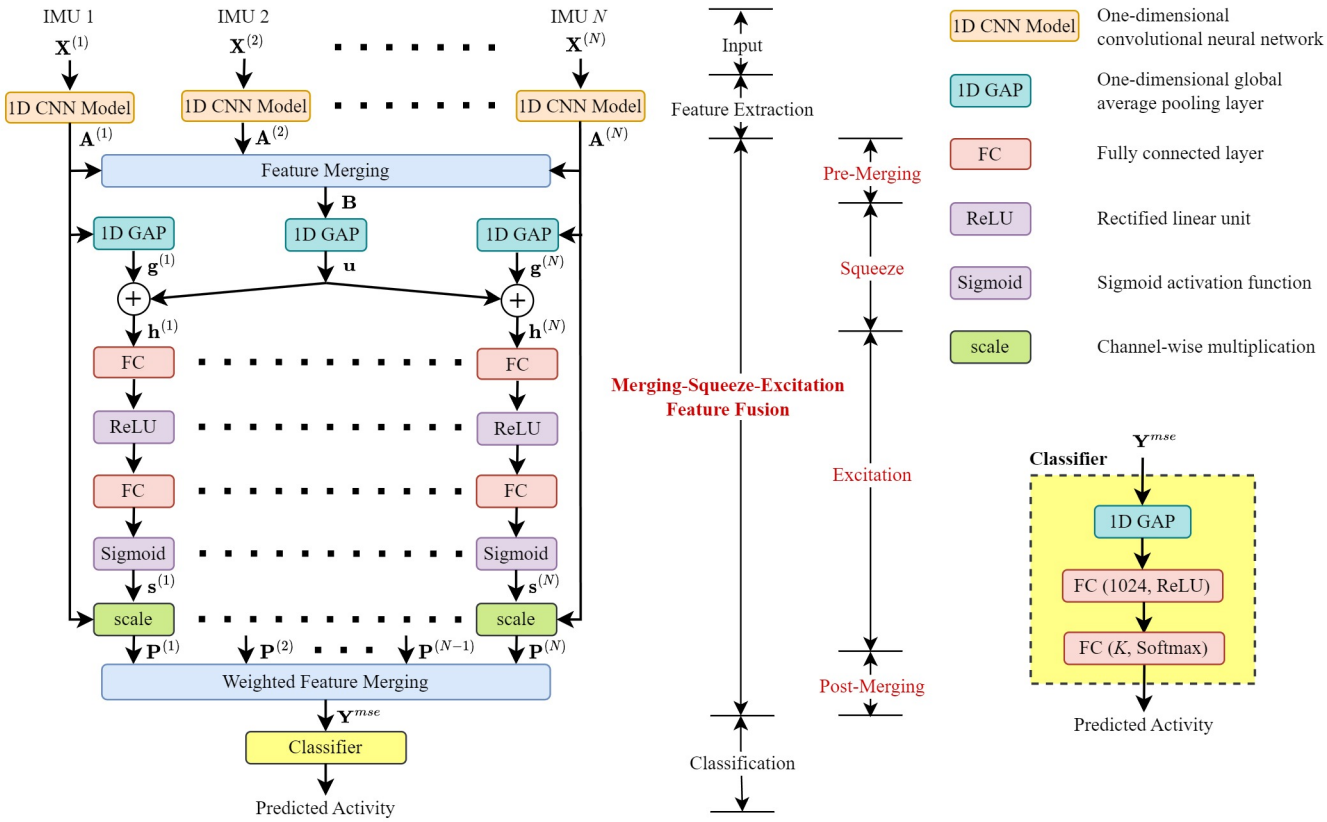

Figure 1. Proposed MSE feature fusion architecture. It consists of the following stages: data inputs, feature extraction, merging-squeeze-excitation feature fusion, and classification. The inputs are the sensor data $\mathbf{X}^{(n)}$ from N IMUs attached to several parts of the user body. Each branch extracts a set of feature maps $\mathbf{A}^{(n)}$ independently by using a 1D CNN model. In the merging-squeeze-excitation feature fusion stage, each set of feature maps is calibrated by using a set of channel weights. A merging method combines the sets of channel-weighted feature maps $\mathbf{P}^{(n)}$ and produces a new set of channel-weighted and merged feature maps $\mathbf{Y}^{mse}$, which is used later in the classification process.

4.1. Input and Feature Extraction

The input $\mathbf{X}^{(n)} \in \mathbb{R}^{1\times L\times M}$ is a three-dimensional (3D) array (consisting of the height, width, and channel dimensions) storing data segments of all sensor data from the n-th IMU, where M is the number of sensor data types per IMU and L is the number of data points in one segment. It can be expressed as $\mathbf{X}^{(n)} = [\mathbf{x}_1^{(n)}; \mathbf{x}_2^{(n)}; \ldots; \mathbf{x}_M^{(n)}]$, where $\mathbf{x}_m^{(n)}$ is a data segment of the m-th sensor from the n-th IMU and expressed in Equation (2). Note that $[(\cdot); (\cdot); \ldots, (\cdot)]$ denotes that the elements inside are arranged along the channel dimension. Each branch will apply a 1D CNN model to extract feature maps $\mathbf{A}^{(n)} \in \mathbb{R}^{1\times W\times C}$, where W and C are the width and number of channels, respectively. The feature maps $\mathbf{A}^{(n)}$ can be expressed as $\mathbf{A}^{(n)} = [\mathbf{a}_1^{(n)}; \mathbf{a}_2^{(n)}; \ldots; \mathbf{a}_C^{(n)}]$, where the row vector $\mathbf{a}_c^{(n)} = [a_{1,c}^{(n)}, a_{2,c}^{(n)}, \ldots, a_{W,c}^{(n)}]$ is a 1D feature map and $a_{w,c}^{(n)}$ is a value at the w-th data point of the c-th channel. The following CNN models are considered as feature extractors due to their simplicity, low number of layers, and low computational complexities: LeNet5 [39], AlexNet [40], and VGG16 [41]. Note that these models originally consist of two-dimensional (2D) layers since they are

applied to image processing. Here, we implement their 1D versions by changing all 2D layers to be 1D layers. For example, 2D convolutional layers are replaced by 1D convolutional layers and 2D max pooling layers are replaced by 1D max pooling layers. The other parameters are unchanged such as numbers of filters and kernel sizes. These 1D CNN structures are summarized in Appendix A. The width W and the number of channels C of $\mathbf{A}^{(n)}$ according to the considered CNN models are shown in Table 7. In addition to these three CNN models, other CNN models can be applied to extract $\mathbf{A}^{(n)}$.

Table 7. The width W and the number of channel C of the feature maps $\mathbf{A}^{(n)}$ obtained from LeNet5, AlexNet, and VGG16 by using the PAMAP2, DaLiAc, and DSAD datasets.

CNN Model	PAMAP2		DaLiAc		DSAD	
	W	C	W	C	W	C
LeNet5	69	120	144	120	25	120
AlexNet	8	256	17	256	2	256
VGG16	9	512	18	512	3	512

4.2. Merging-Squeeze-Excitation Feature Fusion

Conventional feature fusion methods [19–28] combine all feature maps from all branches equally without considering which feature maps are useful. However, some feature maps in $\mathbf{A}^{(n)}$ may be unhelpful for the classification and they should be suppressed while the informative feature maps in $\mathbf{A}^{(n)}$ should be emphasized. Therefore, in this work, inspired by the SE mechanism [16], we propose a feature fusion method called the merging-squeeze-excitation, which is aware of this issue. As shown in Figure 1, all sets of feature maps $\mathbf{A}^{(n)}$, for $n = 1, 2, \ldots, N$, are firstly combined in the pre-merging step to create a set of pre-merged feature maps $\mathbf{B}$. Unlike [16], in the squeeze step, the channel-wise statistics $\mathbf{h}^{(n)}$ used to compute the channel weights $\mathbf{s}^{(n)}$ are computed according to the feature maps $\mathbf{A}^{(n)}$ and pre-merged feature maps $\mathbf{B}$. This implies that the importance of each feature map in $\mathbf{A}^{(n)}$ is measured not only from $\mathbf{A}^{(n)}$ but also from $\mathbf{B}$. Accordingly, we find the corresponding channel weights $\mathbf{s}^{(n)}$, multiply them to $\mathbf{A}^{(n)}$, and obtain the channel-weighted feature maps $\mathbf{P}^{(n)}$ in the excitation step. Finally, in the post-merging step, we recombine $\mathbf{P}^{(n)}$, for $n = 1, 2, \ldots, N$, using the same merging method in the pre-merging step to obtain the channel-weighted and merged feature maps $\mathbf{Y}^{mse}$, which will be used in the classification process later. The details of these steps are explained as follows.

4.2.1. Pre-Merging

We use the pre-merging step to initially combine feature maps $\mathbf{A}^{(n)}$ from all N branches together and to produce the pre-merged feature maps $\mathbf{B} \in \mathbb{R}^{1\times W\times C}$, which will be used along with $\mathbf{A}^{(n)}$ to compute the channel weights. The feature maps $\mathbf{B}$ can be expressed as $\mathbf{B} = [\mathbf{b}_1; \mathbf{b}_2; \ldots; \mathbf{b}_C]$, where the row vector $\mathbf{b}_c \in \mathbb{R}^{1\times W}$ is expressed as $\mathbf{b}_c = [b_{1,c}, b_{2,c}, \ldots, b_{W,c}]$ and $b_{w,c}$ is a value at the w-th data point of the c-th channel. Several feature merging methods are available [42]. Here, we investigate and compare the following methods:

- Addition merging creates feature maps $\mathbf{B}$ by using the element-wise addition. The value $b_{w,c}$ is obtained from
$$b_{w,c} = \sum_{n=1}^{N} a_{w,c}^{(n)}, \tag{3}$$
where $a_{w,c}^{(n)}$ is a value at the w-th data point of the c-th channel of $\mathbf{A}^{(n)}$.
- Maximum merging creates feature maps $\mathbf{V}$ by using the element-wise maximum operation. The value $b_{w,c}$ is obtained from
$$b_{w,c} = \max\left\{a_{w,c}^{(1)}, a_{w,c}^{(2)}, \ldots, a_{w,c}^{(N)}\right\}. \tag{4}$$

- Minimum merging creates feature maps $\mathbf{B}$ by using the element-wise minimum operation. The value $b_{w,c}$ is obtained from

$$b_{w,c} = \min\left\{a_{w,c}^{(1)}, a_{w,c}^{(2)}, \ldots, a_{w,c}^{(N)}\right\}. \tag{5}$$

- Average merging creates feature maps $\mathbf{B}$ by using the element-wise averaging operation. The value $a_{w,c}$ is obtained from

$$b_{w,c} = \frac{1}{N}\sum_{n=1}^{N} a_{w,c}^{(n)}. \tag{6}$$

For a future usage, we denote the merging operation as $\mathbb{F}_{Merge}(\cdot)$. Specifically, we have

$$\mathbf{B} = \mathbb{F}_{Merge}\left(\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}\right). \tag{7}$$

4.2.2. Squeeze and Excitation

In the squeeze-and-excitation step, we recalibrate each set of feature maps $\mathbf{A}^{(n)}$ such that the informative feature maps will be emphasized and ambiguous feature maps will be suppressed by using channel weights, which will be computed according to both $\mathbf{A}^{(n)}$ and $\mathbf{B}$. First, we obtain the channel-wise statistics $\mathbf{u} \in \mathbb{R}^C$ by passing $\mathbf{B}$ to a 1D GAP layer and the channel-wise statistics $\mathbf{g}^{(n)} \in \mathbb{R}^C$ by passing $\mathbf{A}^{(n)}$ to another 1D GAP. The statistics $\mathbf{u}$ are expressed as $\mathbf{u} = [u_1, u_2, \ldots, u_C]^T$ and the statistics $\mathbf{g}^{(n)}$ are expressed as $\mathbf{g}^{(n)} = [g_1^{(n)}, g_2^{(n)}, \ldots, g_C^{(n)}]^T$, where $[\cdot, \cdot, \ldots, \cdot]^T$ is the transpose, u_c is obtained by averaging the values in the c-th 1D feature map of $\mathbf{B}$, and $g_c^{(n)}$ is obtained by averaging the values in the c-th 1D feature map of $\mathbf{U}^{(n)}$. Specifically, we have

$$u_c = \frac{1}{W}\sum_{w=1}^{W} b_{w,c}, \tag{8}$$

and

$$g_c^{(n)} = \frac{1}{W}\sum_{w=1}^{W} a_{w,c}^{(n)}. \tag{9}$$

Thereafter, we obtain a channel-wise statistics $\mathbf{h}^{(n)} \in \mathbb{R}^C$ from

$$\mathbf{h}^{(n)} = \mathbf{u} + \mathbf{g}^{(n)}. \tag{10}$$

The statistics $\mathbf{h}^{(n)}$ are expressed as $\mathbf{h}^{(n)} = [h_1^{(n)}, h_2^{(n)}, \ldots, h_C^{(n)}]^T$.

Next, a set of channel weights $\mathbf{s}^{(n)} \in \mathbb{R}^C$, where $\mathbf{s}^{(n)} = [s_1^{(n)}, s_2^{(n)}, \ldots, s_C^{(n)}]^T$, for individual $\mathbf{A}^{(n)}$, is obtained by using two fully connected (FC) layers with the ReLU activation after the first FC layer and the Sigmoid activation after the second FC layer [16]:

$$\mathbf{s}^{(n)} = \sigma\left(\mathbf{W}_2^{(n)}\delta\left(\mathbf{W}_1^{(n)}\mathbf{h}^{(n)}\right)\right), \tag{11}$$

where $\sigma(\cdot)$ is the Sigmoid activation function, $\delta(\cdot)$ is the ReLU activation function, $\mathbf{W}_1^{(n)} \in \mathbb{R}^{\frac{C}{r}\times C}$ is the weight matrix of the first FC layer, $\mathbf{W}_2^{(n)} \in \mathbb{R}^{C\times\frac{C}{r}}$ is the weight matrix of the second FC layer, and r is the reduction ratio which is used to reduce the first FC layer's output dimension.

Finally, we recalibrate the feature maps $\mathbf{A}^{(n)}$ according to the channel weights $\mathbf{s}^{(n)}$ to emphasize useful feature maps and suppress ambiguous feature maps and, then, obtain a set of channel-weighted feature maps $\mathbf{P}^{(n)} \in \mathbb{R}^{1\times W\times C}$. The feature maps $\mathbf{P}^{(n)}$ can be expressed as $\mathbf{P}^{(n)} = [\mathbf{p}_1^{(n)}; \mathbf{p}_2^{(n)}; \ldots; \mathbf{p}_C^{(n)}]$, where $\mathbf{p}_c^{(n)} = [p_{1,c}^{(n)}, p_{2,c}^{(n)}, \ldots, p_{W,c}^{(n)}]$ is a 1D feature

map and obtained from multiplication between the channel weight $s_c^{(n)}$ and the 1D feature map $\mathbf{a}_c^{(n)}$:

$$\mathbf{p}_c^{(n)} = s_c^{(n)} \mathbf{a}_c^{(n)} \tag{12}$$

For a future usage, we denote the squeeze-and-excitation operation to compute $\mathbf{P}^{(n)}$ as

$$\mathbf{P}^{(n)} = \mathbb{F}_{SE}\left(\mathbf{A}^{(n)}, \mathbf{B}\right). \tag{13}$$

4.2.3. Post-Merging

The post-merging step will apply the merging method used in the pre-merging step to combine the N sets of channel-weighted feature maps $\mathbf{P}^{(n)}$ and obtain the channel-weighted and merged feature maps $\mathbf{Y}^{mse}$. Similar to Section 4.2.1, we can express

$$\mathbf{Y}^{mse} = \mathbb{F}_{Merge}\left(\mathbf{P}^{(1)}, \mathbf{P}^{(2)}, \ldots, \mathbf{P}^{(N)}\right). \tag{14}$$

The set of feature maps $\mathbf{Y}^{mse}$ will be used in the classification process.

4.3. Classification

In this work, the classifier as shown in Figure 1 consists of a 1D GAP layer and two FC layers whose ReLU activation functions are used in the first FC layer while the Softmax activation function is used in the second FC layer. The numbers of neurons in the first and second FC layers are 1024 and K, respectively, where K is the number of classes (depending on the datasets). As specified in Section 3, the number of classes K is 12, 13, and 19 for the PAMAP2, DaLiAc, and DSAD datasets, respectively. Note that other classifiers' structures are applicable.

5. Extensions of Merging-Squeeze-Excitation Feature Fusion

In this section, we present four extensions of the MSE feature fusion: MSE feature fusion with local skip connections, MSE feature fusion with a global skip connection, MSE feature fusion with global channel attention, and deep MSE feature fusion. Their performances will be evaluated and compared in Section 6.

5.1. MSE Feature Fusion with Skip Connections

Skip connections were used in ResNet models [43] to solve the vanishing-gradient issue. Here, we will apply this technique to the MSE feature fusion such that feature maps entering the classification will be at least as good as the feature maps obtained from the earlier step. We consider two possible positions to add skip connections.

- The MSE feature fusion with local skip connections is shown in Figure 2a, where we add a skip connection to each branch of $\mathbf{A}^{(n)}$. As a result, we have

$$\mathbf{Q}^{(n)} = \mathbf{A}^{(n)} + \mathbf{P}^{(n)}. \tag{15}$$

  The feature maps $\mathbf{Y}^{lsc}$ that will enter the classifier are obtained from

$$\mathbf{Y}^{lsc} = \mathbb{F}_{Merge}\left(\mathbf{Q}^{(1)}, \mathbf{Q}^{(2)}, \ldots, \mathbf{Q}^{(N)}\right). \tag{16}$$

- The MSE feature fusion with a global skip connection is shown in Figure 2b. We create a skip connection on the MSE feature fusion such that the pre-merged feature maps $\mathbf{B}$ from the pre-merging step will be added to the channel-weighted and merged feature maps $\mathbf{Y}^{mse}$. Thereafter, we have $\mathbf{Y}^{gsc}$ entering to the classifier as follows:

$$\mathbf{Y}^{gsc} = \mathbf{Y}^{mse} + \mathbf{B}, \tag{17}$$

  where $\mathbf{Y}^{mse}$ is defined in (14). The prediction will be based on both $\mathbf{Y}^{mse}$ and $\mathbf{B}$.

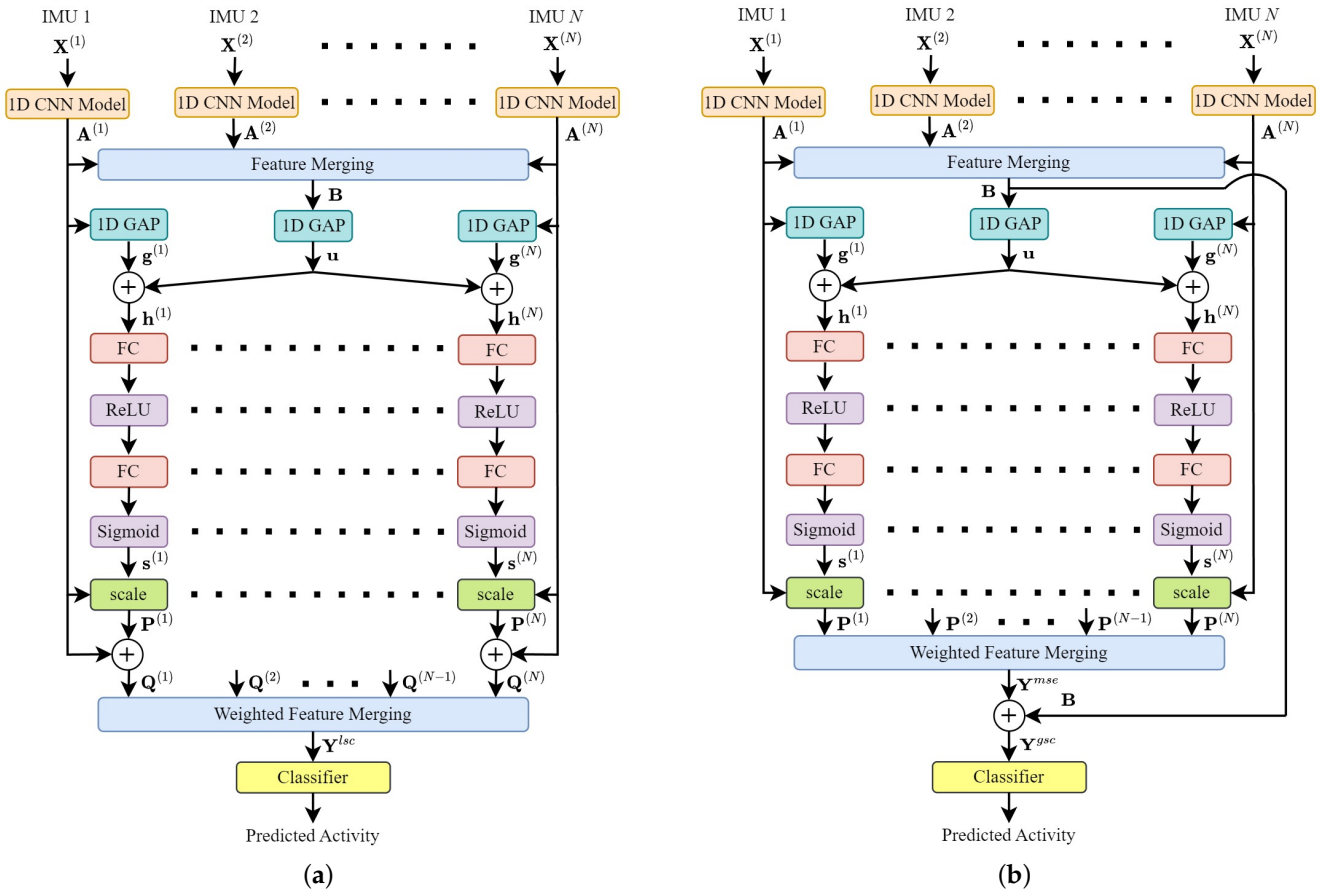

Figure 2. MSE feature fusion architectures with skip connections. There are two types: (**a**) MSE feature fusion with local skip connections. A skip connection is added to each branch. As a result, we combine the feature maps $\mathbf{Q}^{(n)} = \mathbf{P}^{(n)} + \mathbf{A}^{(n)}$ instead of $\mathbf{P}^{(n)}$. The output $\mathbf{Y}^{lsc}$ still contains the feature maps directly obtained from 1D CNN models and (**b**) MSE feature fusion with a global skip connection. The pre-merged feature maps $\mathbf{B}$ are added to the feature maps $\mathbf{Y}^{mse}$. As a result, the output feature maps $\mathbf{Y}^{gsc}$ contain both pre-merged feature maps (no channels weighted) and channel-weighted feature maps.

5.2. MSE Feature Fusion with Global Channel Attention

In the proposed MSE feature fusion shown in Figure 1, the channel-weighted and merged feature maps $\mathbf{Y}^{mse}$ are obtained from $\mathbb{F}_{Merge}(\mathbf{P}^{(1)}, \mathbf{P}^{(2)}, \ldots, \mathbf{P}^{(N)})$. In addition, we may compute a different set of channel-weighted feature maps based on the channel dependency of $\mathbf{B}$ (the output of the pre-merging step) directly. Figure 3 shows the MSE feature fusion with a global channel attention, where we create an additional set of channel-weighted feature maps $\mathbf{R} \in \mathbb{R}^{1 \times W \times C}$ according to $\mathbf{B}$. The set of feature maps $\mathbf{R}$ is denoted as $\mathbf{R} = [\mathbf{r}_1; \mathbf{r}_2; \ldots; \mathbf{r}_C]$, $\mathbf{r}_c = [r_{1,c}, r_{2,c}, \ldots, r_{W,c}]$, and $r_{w,c}$ is a value. Similar to the previous calculation, we can obtain $\mathbf{R}$ according to the following steps. We find the channel weights $\mathbf{v} \in \mathbb{R}^C$, where $\mathbf{v} = [v_1, v_2, \ldots, v_C]^T$ and v_c is a value, from

$$\mathbf{v} = \sigma(\mathbf{W}_2^{\dagger} \delta(\mathbf{W}_1^{\dagger} \mathbf{u})), \tag{18}$$

where $\mathbf{u}$ is the channel-wise statistics as shown in Section 4.2.2, $\mathbf{W}_1^{\dagger} \in \mathbb{R}^{\frac{C}{r} \times C}$ is the weight matrix of the first FC layer, and $\mathbf{W}_2^{\dagger} \in \mathbb{R}^{C \times \frac{C}{r}}$ is the weight matrix of the second FC layer. The c-th 1D feature map $\mathbf{r}_c$ is equal to the feature map $\mathbf{b}_c$ weighted by v_c:

$$\mathbf{r}_c = v_c \mathbf{b}_c. \tag{19}$$

Finally, the set of channel-weighted and merged feature maps $\mathbf{Y}^{gca}$ entering the classifier is from

$$\mathbf{Y}^{gca} = \mathbf{Y}^{mse} + \mathbf{R}, \tag{20}$$

where $\mathbf{Y}^{mse}$ is defined in (14). As a result, the prediction will be computed from both local-channel-attention and global-channel-attention feature maps.

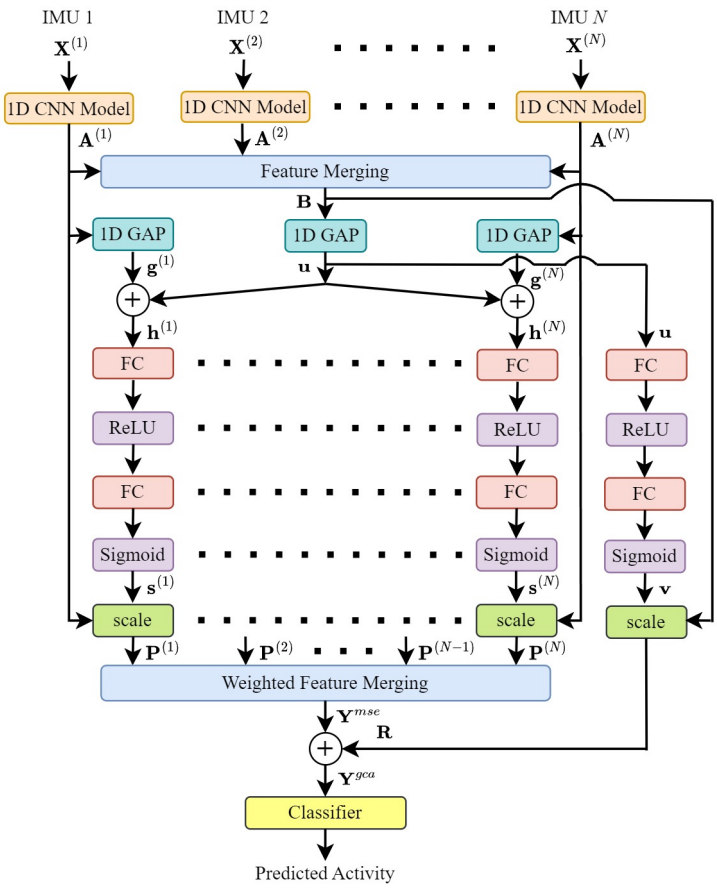

Figure 3. MSE feature fusion architecture with global channel attention. We compute an additional set of channel-weighted feature maps $\mathbf{R}$ which is obtained from the pre-merged feature maps $\mathbf{B}$. The output feature maps $\mathbf{Y}^{gca}$ contain both $\mathbf{Y}^{mse}$ (where feature maps are calibrated and, then, merged) and $\mathbf{R}$ (where feature maps are merged and, then, calibrated).

5.3. Deep MSE Feature Fusion

Instead of using only one-level MSE feature fusion to combine and recalibrate the feature maps $\mathbf{A}^{(n)}$ as shown in Figure 1, we can stack a series of MSE feature fusion blocks to create deep MSE feature fusion, where feature maps are merged and weighted multiple times. The structure of deep MSE feature fusion is shown in Figure 4a, where D MSE feature fusion blocks are connected in series. The d-th block as shown in Figure 4b receives the channel-weighted feature maps $\tilde{\mathbf{P}}^{(d-1,n)}$, for $n = 1, 2, \ldots, N$, and the channel-weighted and merged feature maps $\tilde{\mathbf{Y}}^{deep,(d-1)}$ from the previous block to create the new channel-weighted feature maps $\tilde{\mathbf{P}}^{(d,n)}$ and the new channel-weighted and merged feature maps $\tilde{\mathbf{Y}}^{mse,(d)}$. Note that $\tilde{\mathbf{P}}^{(0,n)}$ and $\tilde{\mathbf{Y}}^{deep,(0)}$ are equal to $\mathbf{A}^{(n)}$ and $\mathbf{B}$ (defined in (7)), respectively. Similar to Section 4.2.2, we have $\tilde{\mathbf{P}}^{(d,n)} = \mathbb{F}_{SE}\left(\tilde{\mathbf{P}}^{(d-1,n)}, \tilde{\mathbf{Y}}^{deep,(d-1)}\right)$ and $\tilde{\mathbf{Y}}^{mse,(d)} = \mathbb{F}_{Merge}\left(\tilde{\mathbf{P}}^{(d,1)}, \tilde{\mathbf{P}}^{(d,2)}, \ldots, \tilde{\mathbf{P}}^{(d,N)}\right)$. Thereafter, the new merged feature maps $\tilde{\mathbf{Y}}^{deep,(d)}$ will be obtained from

$$\tilde{\mathbf{Y}}^{deep,(d)} = \tilde{\mathbf{Y}}^{deep,(d-1)} + \tilde{\mathbf{Y}}^{mse,(d)}, \tag{21}$$

where a skip connection is used to keep the deep MSE feature fusion stable. The feature maps $\tilde{\mathbf{Y}}^{deep,(D)}$ of the last block will be used in the classification process.

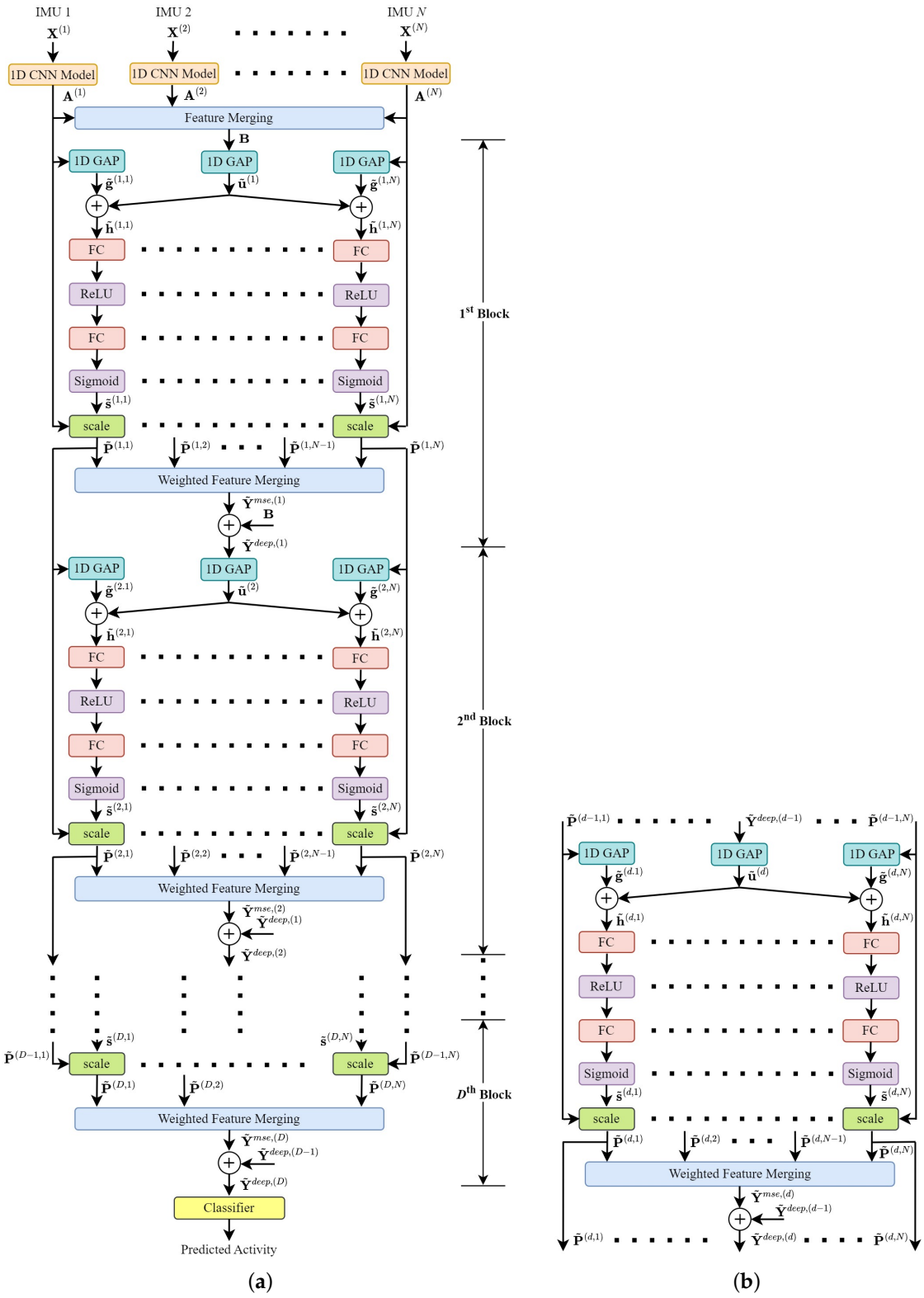

Figure 4. Deep MSE feature fusion. We implement a series of D MSE feature fusion blocks such that the feature maps $\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}$ are calibrated and merged several times as shown in (**a**) deep MSE feature fusion architecture, where the structure of the d-th MSE feature fusion block is shown in (**b**). The output feature maps $\tilde{\mathbf{Y}}^{deep,(D)}$ are used in the classification process.

6. Experimental Results and Discussion

6.1. Experimental Setup

All experiments were implemented by using Python programming language and Python libraries such as Scikit-learn, TensorFlow, Keras, etc. They were run on the Google Colab Pro+ platform. The performances of the investigated models were measured by the accuracy score, which is obtained from

$$Accuracy = \left[\frac{1}{K} \sum_{k=1}^{K} \frac{TP_k + TN_k}{TP_k + FP_k + TN_k + FN_k} \right] \times 100, \tag{22}$$

where K is the number of classes, TP_k is the number of true positives of the k-th class, FP_k is the number of false positives of the k-th class, TN_k is the number of true negatives of the k-th class, and FN_k is the number of false negatives of the k-th class. There are two basic approaches to evaluate the model performances: training–validation–testing split and k-fold cross validation. The training–validation–testing split will divide a dataset into three separated parts: training set, validation set, and testing set. Therefore, the performance results of the investigated model will highly depend on the data in the testing set. In order to avoid this problem, similar to [18,22,25,27], we applied the k-fold cross validation, where k is set to 10, to the experiments. The 10-fold cross validation will divide a dataset into 10 parts. One part will be selected as the testing set while the remaining nine parts will be the training set. We evaluate an investigated model 10 times. For each time, we select a different part to be the testing set. Thereafter, the performance results will be the average of the testing scores. The investigated models were trained by minimizing the categorical cross-entropy using the Adam optimizer with the settings $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-7}$. The training rate was set to 0.001. The batch size was 32. The number of epochs was 40. We did not experience an overfitting issue. Our training scores are slightly higher than the testing scores.

6.2. Baseline Architectures

We consider a single-branch DL architecture and a multi-branch DL architecture shown in Figure 5 as our baseline architectures for the performance comparison. The classifiers in these two architectures are similar to those used in the proposed MSE feature fusion as shown in Figure 1 and explained in Section 4.3.

- For a single-branch DL architecture, all available sensor data will be combined first before we extract a set of features [13]. Here, all sensor data $\mathbf{X}^{(n)}$ (from N IMUs) are concatenated together along the channel dimension. We denote this new array as $\mathbf{X} \in \mathbb{R}^{1 \times L \times NM}$. Thereafter, a 1D CNN model extracts a set of feature maps $\mathbf{A} \in \mathbb{R}^{1 \times W \times C}$ which will be used in the classification process.
- A multi-branch DL architecture consists of N branches to receive the sensor data $\mathbf{X}^{(n)}$ individually [19,22,23]. Each branch extracts a set of feature maps $\mathbf{A}^{(n)}$ using a 1D CNN model. Here, we concatenate these N sets of feature maps together along the channel dimension and obtain a new array $\mathbf{Y}^{mb} \in \mathbb{R}^{1 \times W \times NC}$, which will be sent to the classifier.

Note that the sensor data $\mathbf{X}^{(n)}$ and feature map $\mathbf{A}^{(n)}$ were defined in Section 4. The values W and C were shown in Table 7.

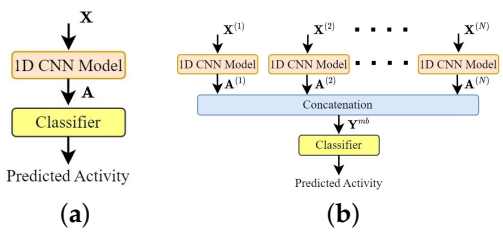

Figure 5. Baseline architectures. (**a**) Single-branch architecture. (**b**) Multi-branch architecture.

The performances of these architectures are evaluated by classifying the PAMAP2, DaLiAc, and DSAD datasets where three CNN models (including LeNet5, AlexNet, and VGG16) are used as feature extraction. The accuracy scores are shown in Table 8, which will be compared to those achieved by the proposed architectures. We observe that the single-branch architectures outperform the multi-branch architectures in many cases. A reason is that the multi-branch architectures have extracted too many features (the output of the GAP in the classifier) and some of them may be ambiguous. The number of features out of the GAP in the multi-branch architectures is equal to NC while the number of features out of the GAP in the single-branch architectures is equal to C. As seen in Table 8, the single-branch architectures using AlexNet offer the highest accuracy scores of 98.77%, 97.60%, and 97.18% for the PAMAP2, DaLiAc, and DSAD datasets, respectively.

Table 8. Accuracy scores (%) of the baseline architectures on classifying the PAMAP2, DaLiAc, and DSAD datasets where LeNet5, AlexNet, and VGG16 are applied as feature extractors. The asterisk (*) indicates the highest accuracy score of each dataset.

Model	PAMAP2			DaLiAc			DSAD		
	LeNet5	AlexNet	VGG16	LeNet5	AlexNet	VGG16	LeNet5	AlexNet	VGG16
Single-Branch Model	96.79	**98.77** *	98.75	95.37	**97.60** *	95.53	91.39	**97.18** *	96.04
Multi-branch Model	98.73	98.51	97.85	96.09	97.10	94.49	94.54	96.59	93.31

6.3. Proposed Merging-Squeeze-Excitation Feature Fusion

The performances of the proposed MSE feature fusion in Section 4 and its extensions in Section 5 are shown in the following subsections. For each proposed architecture, we will compare the accuracy scores among the merging methods (addition, maximum, minimum, and average) and DL models (LeNet5, AlexNet, and VGG16) to determine which combination offers the highest accuracy score on classifying each dataset. Thereafter, the highest accuracy scores of the proposed architectures are compared to determine the best architecture. Note that the reduction ratio r is fixed to eight for all experiments. Varying r is considered as future work.

6.3.1. MSE Feature Fusion

Table 9 presents the accuracy scores of the MSE feature fusion proposed in Section 4 according to the merging methods, DL models, and datasets. We have the following results:

- The highest accuracy score in each dataset is indicated by the asterisk (*). The MSE feature fusion using the minimum merging and AlexNet achieves the highest accuracy score of 99.17% for the PAMAP2 dataset. The MSE feature fusion using the average merging and AlexNet achieves the highest accuracy scores of 98.32% and 98.04% for the DaLiAc and DSAD datasets, respectively.
- We compare the accuracy scores of the MSE feature fusion to those of the baseline architectures in Section 6.2. According to the highest accuracy scores obtained from these architectures, the results show that the MSE feature fusion outperforms the baseline models.
- Among the considered merging methods, the MSE feature fusion using the addition merging offers the worst accuracy scores. The MSE feature fusion architectures using the other merging methods provide the same level of performance. Their accuracy scores are rather close to each other. We do not have a conclusive result on which merging method is the best.
- Among the considered DL models used as feature extractors, the MSE feature fusion using ALexNet outperforms the MSE feature fusion using the other DL models.

Table 9. Accuracy scores (%) of the MSE feature fusion on classifying the PAMAP2, DaLiAc, and DSAD datasets where LeNet5, AlexNet, and VGG16 are applied as feature extractors. The asterisk (*) indicates the highest accuracy score of each dataset.

Merging	PAMAP2			DaLiAc			DSAD		
	LeNet5	AlexNet	VGG16	LeNet5	AlexNet	VGG16	LeNet5	AlexNet	VGG16
Addition	98.59	98.91	97.99	96.82	97.63	96.22	95.53	97.45	94.64
Maximum	98.92	99.06	98.82	98.03	98.04	96.92	97.42	97.68	96.00
Minimum	98.84	**99.17** *	98.72	97.59	98.18	97.72	97.34	97.75	97.50
Average	98.91	99.06	98.79	97.86	**98.32** *	97.67	97.00	**98.04** *	97.92

6.3.2. MSE Feature Fusion with Skip Connections

Tables 10–12 show the accuracy scores of the MSE feature fusion with skip connections proposed in Section 5.1 on classifying the PAMAP2, DaLiAc, and DSAD datasets, respectively. Each table presents the accuracy scores according to the merging methods, the DL models, and skip-connection methods. We have the following results:

- On classifying the PAMAP2 dataset (Table 10), the MSE feature fusion with local skip connections achieves the highest accuracy score of 99.18% when using the minimum merging and AlexNet, while the MSE feature fusion with a global skip connection offers the highest accuracy score of 99.24% when using the average merging and AlexNet. Both architectures outperform the original MSE feature fusion (whose highest accuracy score is 99.17%).
- On classifying the DaLiAc dataset (Table 11), the MSE feature fusion with local skip connections achieves the highest accuracy score of 98.59% when using the minimum merging and AlexNet, while the MSE feature fusion with a global skip connection offers the highest accuracy score of 98.42% when using the minimum merging and AlexNet. Both architectures outperform the original MSE feature fusion (whose highest accuracy score is 98.32%).
- On classifying the DSAD dataset (Table 12), the MSE feature fusion with local skip connections achieves the highest accuracy score of 98.02% when using the average merging and AlexNet while the MSE feature fusion with a global skip connection offers the highest accuracy score of 97.97% when using the average merging and AlexNet. Both architectures offer lower accuracy scores than that of the original MSE feature fusion (whose highest accuracy score is 98.04%).
- Since the results are not conclusive, we cannot indicate whether the MSE feature fusion with skip connections is better than the original MSE feature fusion nor which skip connection method is the best.

Table 10. PAMAP2 dataset: Accuracy scores (%) of the MSE feature fusion with skip connections on classifying the PAMAP2 dataset where LeNet5, AlexNet, and VGG16 are applied as feature extractors. The asterisk (*) indicates the highest accuracy score of each type of skip connections.

Merging	Local Skip Connections			Global Skip Connection		
	LeNet5	AlexNet	VGG16	LeNet5	AlexNet	VGG16
Addition	98.68	98.75	98.16	98.49	98.77	98.23
Maximum	98.99	99.20	98.66	98.89	98.99	98.85
Minimum	99.05	**99.18** *	98.70	98.87	99.13	98.77
Average	98.91	99.15	98.70	98.72	**99.24** *	98.94

Table 11. DaLiAc dataset: Accuracy scores (%) of the MSE feature fusion with skip connections on classifying the DaLiAc dataset where LeNet5, AlexNet, and VGG16 are applied as feature extractors. The asterisk (*) indicates the highest accuracy score of each type of skip connections.

Merging	Local Skip Connections			Global Skip Connection		
	LeNet5	AlexNet	VGG16	LeNet5	AlexNet	VGG16
Addition	96.67	97.71	93.30	96.08	97.12	94.59
Maximum	98.30	97.78	97.13	98.19	97.97	97.00
Minimum	98.13	**98.59** *	97.55	98.12	**98.42** *	97.60
Average	98.12	97.92	97.68	98.05	98.06	98.00

Table 12. DSAD dataset: Accuracy scores (%) of the SE feature fusion with skip connections on classifying the DSAD dataset where LeNet5, AlexNet, and VGG16 are applied as feature extractors. The asterisk (*) indicates the highest accuracy score of each type of skip connections.

Merging	Local Skip Connections			Global Skip Connection		
	LeNet5	AlexNet	VGG16	LeNet5	AlexNet	VGG16
Addition	94.98	96.61	95.96	95.16	97.16	94.57
Maximum	97.65	97.40	97.00	97.57	97.89	97.27
Minimum	97.42	97.72	97.60	97.42	97.27	97.48
Average	97.18	**98.02** *	97.83	97.12	**97.97** *	97.97

6.3.3. MSE Feature Fusion with Global Channel Attention

Table 13 shows the accuracy scores of the MSE feature fusion with global channel attention proposed in Section 5.2 according to the merging methods, DL models, and datasets. We have the following results:

- On classifying the PAMAP2 and DaLiAc datasets, the MSE feature fusion with global channel attention achieves the highest accuracy score of 99.17% and 98.08%, respectively, when using the minimum merging and AlexNet.
- On classifying the DSAD dataset, the MSE feature fusion with global channel attention achieves the highest accuracy scores of 97.87% when using the average merging and AlexNet.
- By comparing these accuracy scores to those of the original MSE feature fusion, we see that the original MSE feature fusion outperforms the MSE feature fusion with global channel attention. The feature maps obtained by using the global channel attention do not provide any additional information.

Table 13. Accuracy scores (%) of the MSE feature fusion with global channel attention on classifying the PAMAP2, DaLiAc, and DSAD datasets where LeNet5, AlexNet, and VGG16 are applied as feature extractors. The asterisk (*) indicates the highest accuracy score of each dataset.

Merging	PAMAP2			DaLiAc			DSAD		
	LeNet5	AlexNet	VGG16	LeNet5	AlexNet	VGG16	LeNet5	AlexNet	VGG16
Addition	98.33	98.32	96.88	96.65	97.10	95.95	95.09	97.38	94.65
Maximum	98.94	99.06	98.04	97.73	97.55	96.85	97.54	97.81	97.12
Minimum	99.01	**99.17** *	98.73	97.47	**98.08** *	97.69	97.24	97.27	97.55
Average	98.82	98.99	98.73	97.60	97.78	97.28	97.08	**98.03** *	97.97

6.3.4. Deep MSE Feature Fusion

Tables 14–16 show the accuracy scores of the deep MSE feature fusion (in Section 5.3) using AlexNet as the feature extractor on classifying the PAMAP2, DaLiAc, and DSAD datasets, respectively. We consider only AlexNet since it outperforms the other DL models as shown in the previous subsections. Each table presents the accuracy scores according

to the merging methods and the number of MSE feature fusion blocks (D). We see that the deep MSE feature fusion with $D = 1$ offers the highest accuracy scores for all three datasets (i.e, 99.17% for the PAMAP2 dataset, 98.32% for the DaLiAc dataset, and 98.04% for the DSAD dataset). In fact, with $D = 1$, the deep MSE feature fusion is equivalent to the original MSE feature fusion. This indicates that, for the investigated datasets, increasing the number of MSE feature fusion blocks does not provide any further useful information to the output feature maps $\tilde{\mathbf{Y}}^{deep,(D)}$ which are used in the classification process.

Table 14. Accuracy scores (%) of the deep MSE feature fusion on classifying the PAMAP2 dataset where AlexNet is applied as the feature extractor.

Merging	Number of MSE Feature Fusion Blocks (D)				
	$D = 1$	$D = 2$	$D = 3$	$D = 4$	$D = 5$
Addition	98.91	98.54	98.84	98.77	98.79
Maximum	99.06	98.73	98.94	99.08	99.06
Minimum	**99.17** *	99.03	99.03	99.03	98.99
Average	99.06	98.79	98.72	99.03	99.10

Table 15. Accuracy scores (%) of the deep MSE feature fusion on classifying the DaLiAc dataset where AlexNet is applied as the feature extractor.

Merging	Number of MSE Feature Fusion Blocks (D)				
	$D = 1$	$D = 2$	$D = 3$	$D = 4$	$D = 5$
Addition	97.63	97.69	97.32	96.90	98.03
Maximum	98.04	97.90	97.83	98.21	98.17
Minimum	98.18	98.04	98.03	97.74	98.06
Average	**98.32** *	98.06	97.49	97.41	98.08

Table 16. Accuracy scores (%) of the deep MSE feature fusion on classifying the DSAD dataset where AlexNet is applied as the feature extractor.

Merging	Number of MSE Feature Fusion Blocks (D)				
	$D = 1$	$D = 2$	$D = 3$	$D = 4$	$D = 5$
Addition	97.45	97.53	97.05	97.52	97.28
Maximum	97.68	97.19	97.60	97.32	97.52
Minimum	97.75	97.31	97.18	97.28	97.45
Average	**98.04** *	97.61	97.50	97.58	97.49

6.4. Computational Complexity Comparison

Tables 17–19 show the numbers of trainable parameters of the baseline architectures, the proposed MSE feature fusion, and the extensions of the MSE feature fusion on classifying the PAMAP2, DaLiAc, and DSAD datasets, respectively. We do not specify the numbers of trainable parameters for individual merging methods since they are the same. The following results are obtained:

- The numbers of trainable parameters of the proposed MSE feature fusion are higher than those of the single-branch architecture since the proposed MSE feature fusion consists of several branches using CNN models as feature extractors. On the other hand, the proposed MSE feature fusion requires lower numbers of trainable parameters than the multi-branch architecture does since the MSE feature fusion reduces the number of features which will enter the classification process by using the addition, maximum, minimum, and average merging instead of the concatenation merging.
- The numbers of trainable parameters of the extensions (of the MSE feature fusion) are slightly higher than those of the proposed MSE feature fusion since the modification parts in the extensions require few trainable parameters.

Table 17. PAMAP2 dataset: The numbers of trainable parameters of the baseline architectures and the proposed MSE feature fusion architectures on classifying the PAMAP2 dataset.

	Architecture	LeNet5	AlexNet	VGG16
Baselines	Single-Branch Model	147,506	1,472,684	5,460,108
	Multi-branch Model	413,710	4,315,372	16,339,852
Proposed MSE Feature Fusion		179,155	3,841,100	15,489,612
Extensions	Local Skip Connections	179,155	3,841,100	15,489,612
	Global Skip Connection	179,155	3,841,100	15,489,612
	Global Channel Attention	182,890	3,857,772	15,555,724
	Deep MSE (D = 2)	-	3,891,116	-
	Deep MSE (D = 3)	-	3,941,132	-
	Deep MSE (D = 4)	-	3,991,148	-
	Deep MSE (D = 5)	-	4,041,164	-

Table 18. DaLiAc dataset: The numbers of trainable parameters of the baseline architectures and the proposed MSE feature fusion architectures on classifying the DaLiAc dataset.

	Architecture	LeNet5	AlexNet	VGG16
Baseline	Single-Branch Model	148,171	1,461,037	5,458,829
	Multi-branch Model	547,477	5,725,069	21,778,445
Proposed MSE Feature Fusion		193,777	5,005,325	20,470,029
Extensions	Local Skip Connections	193,777	5,005,325	20,470,029
	Global Skip Connection	193,777	5,005,325	20,470,029
	Global Channel Attention	197,512	5,021,997	20,536,141
	Deep MSE (D = 2)	-	5,072,013	-
	Deep MSE (D = 3)	-	5,138,701	-
	Deep MSE (D = 4)	-	5,205,389	-
	Deep MSE (D = 5)	-	5,272,077	-

Table 19. DSAD dataset: The numbers of trainable parameters of the baseline architectures and the proposed MSE feature fusion architectures on classifying the DSAD dataset.

	Architecture	LeNet5	AlexNet	VGG16
Baseline	Single-Branch Model	154,951	1,489,363	5,469,011
	Multi-branch Model	687,359	7,174,739	27,228,499
Proposed MSE Feature Fusion		214,514	6,209,523	25,461,907
Extensions	Local Skip Connections	214,514	6,209,523	25,461,907
	Global Skip Connection	214,514	6,209,523	25,461,907
	Global Channel Attention	218,249	6,226,195	25,528,019
	Deep MSE (D = 2)	-	6,292,883	-
	Deep MSE (D = 3)	-	6,376,243	-
	Deep MSE (D = 4)	-	6,459,603	-
	Deep MSE (D = 5)	-	6,542,963	-

6.5. Performance Comparison to Other HAR Approaches

Table 20 shows the accuracy scores of other HAR approaches which were evaluated by using the PAMAP2, DaLiAc, and DSAD datasets. These accuracy scores were presented in their publications. Note that the evaluation setups and pre-processing may be different from ours. We compare them to the highest accuracy scores achieved by the original MSE feature fusion (Section 6.3.1). The proposed MSE feature fusion offers higher accuracy scores than those obtained from the other approaches.

Table 20. The accuracy scores (%) of related work who were evaluated on classifying the PAMAP2, DaLiAc, and DSAD datasets.

Dataset	Year	Reference and Model Name	Accuracy
PAMAP2	2018	[19] CNN-IMU	93.13
	2020	[22] GlobalFusion	90.86
	2021	[26] Multi-Input CNN-GRU	95.27
	2022	[28] Multibranch CNN-BiLSTM	94.29
	2023	[23] Multi-ResAtt	93.19
	2023	Proposed MSE Feature Fusion	99.17
DaLiAc	2018	[44] Iss2Image	96.40
	2021	[45] DeepFusionHAR	97.20
	2023	Proposed MSE Feature Fusion	98.32
DSAD	2020	[22] GlobalFusion	94.28
	2021	[45] DeepFusionHAR	96.10
	2023	Proposed MSE Feature Fusion	98.04

7. Conclusions and Future Work

In this work, we proposed a feature fusion method called the merging-squeeze-excitation (MSE) feature fusion for wearable-sensor-based HAR using multibranch architectures. The MSE feature fusion will calibrate the feature maps during the fusion. Each feature map will be emphasized or suppressed according to its importance measured from both itself and the corresponding pre-merged feature map. In addition, we presented the following four extensions of the MSE feature fusion: the MSE feature fusion with local skip connections, the MSE feature fusion with a global skip connection, the MSE feature fusion with global channel attention, and deep MSE feature fusion. LeNet5, AlexNet, and VGG16 were applied as feature extractors. The addition, maximum, minimum, and average merging were used in the pre-merging and post-merging steps. According to the experimental results, the MSE feature fusion with a global skip connection (using the average merging and AlexNet), the MSE feature fusion with local skip connections (using the minimum merging and AlexNet), and the original MSE feature fusion (using the average merging and AlexNet) achieve the highest accuracy scores of 99.24%, 98.59%, and 98.04% on the PAMAP2, DaLiAc, and DSAD datasets, respectively. For future work, in addition to the channel–attention mechanism, other attention techniques such as spatial attention, modal attention, convolutional block attention, and selective kernel convolution can be applied to feature fusion in order to combine feature maps from different branches effectively.

Funding: This research was funded by the SIIT Young Researcher Grant, under a contract No. SIIT 2019-YRG-SL04, the Sirindhorn International Institute of Technology, Thammasat University, Thailand.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets used to support the findings of this work are available in [36–38].

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Architectures of 1D LeNet5, 1D AlexNet, and 1D VGG16 Models

The architectures of 1D LeNet5, 1D AlexNet, and 1D VGG16 models used as feature extractors in Section 4.1 are presented in Tables A1–A3. Note that, for the AlexNet and VGG16 models, we added a batch normalization layer between the 1D convolutional layer and its activation layer to normalize the output of the convolutional layer before passing it to the activation layer.

Table A1. Architecture of 1D LeNet5. The column names # Filters, K Size, Pad, and Activ are short for number of filters, kernel size, padding, and activation, respectively.

Name	Layer	# Filters	K Size	Stride	Pad	Activ	Size of Output					
							PAMAP2		DaLiAc		DSAD	
							W	C	W	C	W	C
In	Input	-	-	-	-	-	300	12	600	6	125	9
C1	Conv1D	6	5	1	same	tanh	300	6	600	6	125	6
S2	AveragePooling1D	-	2	2	valid	-	150	6	300	6	62	6
C3	Conv1D	16	5	1	valid	tanh	146	16	296	16	58	16
S4	AveragePooling1D	-	2	2	valid	-	73	16	148	16	29	16
C5	Conv1D	120	5	1	valid	tanh	69	120	144	120	25	120

Table A2. Architecture of 1D AlexNet with batch normalization (BatchNorm) layers. The column names # Filters, K Size, Pad, and Activ are short for number of Filters, kernel size, padding, and activation, respectively.

Name	Layer	# Filters	K Size	Stride	Pad	Activ	Size of Output					
							PAMAP2		DaLiAc		DSAD	
							W	C	W	C	W	C
In	Input	-	-	-	-	-	300	12	600	6	125	9
C1	Conv1D + BatchNorm	96	11	4	valid	ReLU	73	96	148	96	29	96
S2	MaxPooling1D	-	3	2	valid	-	36	96	73	96	14	96
C3	Conv1D + BatchNorm	256	5	1	same	ReLU	36	256	73	256	14	256
S4	MaxPooling1D	-	3	2	valid	-	17	256	36	256	6	256
C5	Conv1D + BatchNorm	384	3	1	same	ReLU	17	384	36	384	6	384
C6	Conv1D + BatchNorm	384	3	1	same	ReLU	17	384	36	384	6	384
C7	Conv1D + BatchNorm	256	3	1	same	ReLU	17	384	36	384	6	384
S8	MaxPooling1D	-	3	2	valid	-	8	256	17	256	2	256

Table A3. Architecture of 1D VGG16 with batch normalization (BatchNorm) layers. The column names # Filters, K Size, Pad, and Activ are short for number of filters, kernel size, padding, and activation, respectively.

Name	Layer	# Filters	K Size	Stride	Pad	Activ	Size of Output					
							PAMAP2		DaLiAc		DSAD	
							W	C	W	C	W	C
In	Input	-	-	-	-	-	300	12	600	6	125	9
C1	Conv1D + BatchNorm	64	3	1	same	ReLU	300	64	600	64	125	64
C2	Conv1D + BatchNorm	64	3	1	same	ReLU	300	64	600	64	125	64
S3	MaxPooling1D	-	2	2	valid	-	150	64	300	64	62	64
C4	Conv1D + BatchNorm	128	3	1	same	ReLU	150	128	300	128	62	128
C5	Conv1D + BatchNorm	128	3	1	same	ReLU	150	128	300	128	62	128
S6	MaxPooling1D	-	2	2	valid	-	75	128	150	128	31	128
C7	Conv1D + BatchNorm	256	3	1	same	ReLU	75	256	150	256	31	256
C8	Conv1D + BatchNorm	256	3	1	same	ReLU	75	256	150	256	31	256
C9	Conv1D + BatchNorm	256	3	1	same	ReLU	75	256	150	256	31	256
S10	MaxPooling1D	-	2	2	valid	-	37	256	75	256	15	256
C11	Conv1D + BatchNorm	512	3	1	same	ReLU	37	512	75	512	15	512
C12	Conv1D + BatchNorm	512	3	1	same	ReLU	37	512	75	512	15	512
C13	Conv1D + BatchNorm	512	3	1	same	ReLU	37	512	75	512	15	512
S14	MaxPooling1D	-	2	2	valid	-	18	512	37	512	7	512
C15	Conv1D + BatchNorm	512	3	1	same	ReLU	18	512	37	512	7	512
C16	Conv1D + BatchNorm	512	3	1	same	ReLU	18	512	37	512	7	512
C17	Conv1D + BatchNorm	512	3	1	same	ReLU	18	512	37	512	7	512
S18	MaxPooling1D	-	2	2	valid	-	9	512	18	512	3	512

References

1. Yudav, S.K.; Tiwari, K.; Pandey, H.M.; Akbar, S.A. A Review of Multimodal Human Activity Recognition with Special Emphasis on Classification, Applications, Challenges and Future Directions. *Knowl. Based Syst.* **2021**, *223*, 106970. [CrossRef]
2. Özyer, T.; Ak, D.S.; Alhajj, R. Human Action Recognition Approaches with Video Datasets—A Survey. *Knowl. Based Syst.* **2021**, *222*, 106995. [CrossRef]
3. Bulling, A.; Blanke, U.; Schiele B. A Tutorial on Human Activity Recognition Using Body-Worn Inertial Sensors. *ACM Comput. Surv.* **2014**, *46*, 1–33. [CrossRef]
4. Bouchabou, D.; Nguyen, S.M.; Lohr, C.; LeDuc, B.; Kanellos, I. A Survey of Human Activity Recognition in Smart Homes Based on IoT Sensors Algorithms: Taxonomies, Challenges, and Opportunities with Deep Learning. *Sensors* **2021**, *21*, 6037. [CrossRef] [PubMed]
5. Chaurasia, S.K.; Reddy, S.R.N. State-of-the-art Survey on Activity Recognition and Classification Using Smartphones and Wearable Sensors. *Multimed. Tools Appl.* **2022**, *81*, 1077–1108. [CrossRef]
6. Yang, Y.; Wang, H.; Jiang, R.; Guo, X.; Cheng, J.; Chen, Y. A Review of IoT-Enabled Mobile Healthcare: Technologies, Challenges, and Future Trends. *IEEE Internet Things J.* **2022**, *9*, 9478–9502. [CrossRef]
7. Achirei, S.-D.; Heghea, M.-C.; Lupu, R.-G.; Manta, V.-I. Human Activity Recognition for Assisted Living Based on Scene Understanding. *Appl. Sci.* **2022**, *12*, 10743. [CrossRef]
8. Sousa Lima, W.; Souto, E.; El-Khatib, K.; Jalali, R.; Gama, J. Human Activity Recognition Using Inertial Sensors in a Smartphone: An Overview. *Sensors* **2019**, *19*, 3213. [CrossRef]
9. Ramanujam, E.; Perumal, T.; Padmavathi, S. Human Activity Recognition with Smartphone and Wearable Sensors Using Deep Learning Techniques: A Review. *IEEE Sens. J.* **2021**, *21*, 13029–13040. [CrossRef]
10. Pannurat, N.; Thiemjarus, S.; Nantajeewarawat, E.; Anantavrasilp, I. Analysis of Optimal Sensor Positions for Activity Classification and Application on a Different Data Collection Scenario. *Sensors* **2017**, *17*, 774. [CrossRef]
11. Ahmed, N.; Rafiq, J.I.; Islam, M.R. Enhanced Human Activity Recognition Based on Smartphone Sensor Data Using Hybrid Feature Selection Model. *Sensors* **2020**, *20*, 317. [CrossRef] [PubMed]
12. Chen, L.; Fan, S.; Kumar, V.; Jia, Y. A Method of Human Activity Recognition in Transitional Period. *Information* **2020**, *11*, 416. [CrossRef]
13. Chen, K.; Zhang, D.; Yao, L.; Guo, B.; Yu, Z.; Liu, Y. Deep Learning for Sensor-Based Human Activity Recognition: Overview, Challenges, and Opportunities. *ACM Comput. Surv.* **2021**, *54*, 77. [CrossRef]
14. Gu, F.; Chung, M.-H.; Chignell, M.; Valaee, S.; Zhou, B.; Liu, X. A Survey on Deep Learning for Human Activity Recognition. *ACM Comput. Surv.* **2021**, *54*, 177. [CrossRef]
15. Zhang, S.; Li, Y.; Zhang, S.; Shahabi, F.; Xia, S.; Deng, Y.; Alshurafa, N. Deep Learning in Human Activity Recognition with Wearable Sensors: A Review on Advances. *Sensors* **2022**, *22*, 1476. [CrossRef] [PubMed]
16. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
17. Zhongkai, Z.; Kobayashi, S.; Kondo, K.; Hasegawa, T.; Koshino, M. A Comparative Study: Toward an Effective Convolutional Neural Network Architecture for Sensor-Based Human Activity Recognition. *IEEE Access* **2022**, *10*, 20547–20558. [CrossRef]
18. Mekruksavanich, S.; Hnoohom, N.; Jitpattanakul, A. A Hybrid Deep Residual Network for Efficient Transitional Activity Recognition Based on Wearable Sensors. *Appl. Sci.* **2022**, *12*, 4988. [CrossRef]
19. Moya Rueda, F.; Grzeszick, R.; Fink, G.A.; Feldhorst, S.; Ten Hompel, M. Convolutional Neural Networks for Human Activity Recognition Using Body-Worn Sensors. *Informatics* **2018**, *5*, 26. [CrossRef]
20. Avilés-Cruz, C.; Ferreyra-Ramírez, A.; Zúñiga-López, A.; Villegas-Cortéz, J. Coarse-Fine Convolutional Deep-Learning Strategy for Human Activity Recognition. *Sensors* **2019**, *19*, 1556. [CrossRef]
21. Khan, Z.N.; Ahmad, J. Attention Induced Multi-Head Convolutional Neural Network for Human Activity Recognition. *Appl. Soft Comput.* **2021**, *110*, 107671. [CrossRef]
22. Liu, S.; Yao, S.; Li, J.; Liu, D.; Wang, T.; Shao, H.; Abdelzaher, T. GIobalFusion: A Global Attentional Deep Learning Framework for Multisensor Information Fusion. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* **2020**, *4*, 19. [CrossRef]
23. Al-qaness, M.A.A.; Dahou, A.; Elaziz, M.A.; Helmi, A.M. Multi-ResAtt: Multilevel Residual Network with Attention for Human Activity Recognition Using Wearable Sensors. *IEEE Trans. Industr. Inform.* **2023**, *19*, 144–152. [CrossRef]
24. Zhang, H.; Xiao, Z.; Wang, J.; Li, F.; Szczerbicki, E. A Novel IoT-Perceptive Human Activity Recognition (HAR) Approach Using Multihead Convolutional Attention. *IEEE Internet Things J.* **2020**, *7*, 1072–1080. [CrossRef]
25. Ihianle, I.K.; Nwajana, A.O.; Ebenuwa, S.H.; Otuka, R.I.; Owa, K.; Orisatoki, M.O. A Deep Learning Approach for Human Activities Recognition from Multimodal Sensing Devices. *IEEE Access* **2020**, *8*, 179028–179038. [CrossRef]
26. Dua, N.; Singh, S.N.; Semwal, V.B. Multi-Input CNN-GRU Based Human Activity Recognition Using Wearable Sensors. *Computing* **2021**, *103*, 1461–1478. [CrossRef]
27. Yen, C.-T.; Liao, J.-X.; Huang, Y.-K. Feature Fusion of a Deep-Learning Algorithm into Wearable Sensor Devices for Human Activity Recognition. *Sensors* **2021**, *21*, 8294. [CrossRef]
28. Challa, S.K.; Kumar, A.; Semwal, V.B. A Multibranch CNN-BiLSTM Model for Human Activity Recognition Using Wearable Sensor Data. *Vis. Comput.* **2022**, *38*, 4095–4109. [CrossRef]

29. Li, Y.; Guo, L.; Liu, Y.; Liu, J.; Meng, F. A Temporal-Spectral-Based Squeeze-and- Excitation Feature Fusion Network for Motor Imagery EEG Decoding. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2021**, *29*, 1534–1545. [CrossRef]
30. Sleeman, W.C.; Kapoor, R.; Ghosh, P. Multimodal Classification: Current Landscape, Taxonomy and Future Directions. *ACM Comput. Surv.* **2022**, *55*, 150. [CrossRef]
31. Arevalo, J.; Solorio, T.; Montes-y-Gómez, M.; González, F.A. Gated Multimodal Units for Information Fusion. *arXiv* **2017**, arXiv:1702.01992.
32. Yuan, Z.; Zhang, W.; Tian, C.; Rong, X.; Zhang, Z.; Wang, H.; Fu, K.; Sun, X. Remote Sensing Cross-Modal Text-Image Retrieval Based on Global and Local Information. *IEEE Trans Geosci. Remote Sens.* **2022**, *60*, 1–16. [CrossRef]
33. Yuan, Z.; Zhang, W.; Tian, C.; Mao, Y.; Zhou, R.; Wang, H.; Fu, K.; Sun, X. MCRN: A Multi-Source Cross-Modal Retrieval Network for Remote Sensing. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *115*, 103071. [CrossRef]
34. Jia, Z.; Cai, X.; Jiao, Z. Multi-Modal Physiological Signals Based Squeeze-and-Excitation Network With Domain Adversarial Learning for Sleep Staging. *IEEE Sensors J.* **2022**, *22*, 3464–3471. [CrossRef]
35. Shu, X.; Yang, J.; Yan, R.; Song, Y. Expansion-Squeeze-Excitation Fusion Network for Elderly Activity Recognition. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 5281–5292. [CrossRef]
36. Reiss, A.; Stricker D. Introducing a New Benchmarked Dataset for Activity Monitoring. In Processings of the 6th International Symposium on Wearable Computers, Newcastle, UK, 18–22 June 2012; pp. 108–109.
37. Leutheuser, H.; Schuldhaus, D.; Eskofier, B.M. Hierarchical, Multi-Sensor Based Classification of Daily Life Activities: Comparison with State-of-the-Art Algorithms Using a Benchmark Dataset. *PLoS ONE* **2013**, *8*, e75196. [CrossRef] [PubMed]
38. Altun, K.; Barshan, B.; Tunçel, O. Comparative Study on Classifying Human Activities with Miniature Inertial and Magnetic Sensors. *Pattern Recognit.* **2010**, *43*, 3605–3620. [CrossRef]
39. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-Based Learning Applied to Document Recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
40. Krizhevsky, A.; Sutskever, I.; Hinton, G. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
41. Simonyan, K.; Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
42. Merging Layers. Available online: https://keras.io/api/layers/merging_layers (accessed on 22 January 2023).
43. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. *arXiv* **2015**, arXiv:1512.03385.
44. Hur, T.; Bang, J.; Huynh-The, T.; Lee, J.; Kim, J.-I.; Lee, S. Iss2Image: A Novel Signal-Encoding Technique for CNN-Based Human Activity Recognition. *Sensors* **2018**, *18*, 3910. [CrossRef]
45. Huynh-The, T.; Hua, C.-H.; Tu, N. A.; Kim, D.-S. Physical Activity Recognition With Statistical-Deep Fusion Model Using Multiple Sensory Data for Smart Health. *IEEE Internet Things J.* **2021**, *8*, 1533–1543. [CrossRef]