

Article

Asymmetric Residual Neural Network for Accurate Human Activity Recognition

Jun Long ^{1,2} , Wuqing Sun ¹ , Zhan Yang ^{1,*}  and Osolo Ian Raymond ¹ 

¹ School of Computer Science and Engineering, Central South University, Changsha 410083, China; junlong@csu.edu.cn (J.L.); wq_sun@csu.edu.cn (W.S.); osoloian@csu.edu.cn (O.I.R.)

² Network Resources Management and Trust Evaluation Key Laboratory of Hunan Province, Changsha 410083, China

* Correspondence: zyang22@csu.edu.cn; Tel.: +86-0731-8253-9926

Received: 21 May 2019; Accepted: 5 June 2019; Published: 6 June 2019



Abstract: Human activity recognition (HAR) using deep neural networks has become a hot topic in human–computer interaction. Machines can effectively identify human naturalistic activities by learning from a large collection of sensor data. Activity recognition is not only an interesting research problem but also has many real-world practical applications. Based on the success of residual networks in achieving a high level of aesthetic representation of automatic learning, we propose a novel asymmetric residual network, named ARN. ARN is implemented using two identical path frameworks consisting of (1) a short time window, which is used to capture spatial features, and (2) a long time window, which is used to capture fine temporal features. The long time window path can be made very lightweight by reducing its channel capacity, while still being able to learn useful temporal representations for activity recognition. In this paper, we mainly focus on proposing a new model to improve the accuracy of HAR. In order to demonstrate the effectiveness of the ARN model, we carried out extensive experiments on benchmark datasets (i.e., OPPORTUNITY, UniMiB-SHAR) and compared the results with some conventional and state-of-the-art learning-based methods. We discuss the influence of networks parameters on performance to provide insights about its optimization. Results from our experiments show that ARN is effective in recognizing human activities via wearable datasets.

Keywords: human activity recognition; deep neural network; residual networks; spatial features; temporal features

1. Introduction

Human activity recognition (HAR) is very important for human–computer interaction and is an indispensable part of many current real-world applications. To overcome the awareness of human–computer interaction, the potential features in the on-body device must be learned. HAR using wearable device data is at the core of intelligent assistive technology due to its proliferative applications in smart homes [1], intelligent traffic control [2], medical/health assistance [3,4], skill-based checks [5], and even in the security field [6]. Particularly for elderly people who are in remote areas and need to be continuously monitored, HAR can greatly increase their safety [7].

Nowadays, accelerators, gyroscopes, and magnetic field sensors are widely utilized in smart phones (e.g., Apple iPhone, Samsung Galaxy, Huawei P/Mate) and smart bracelets (Apple Watch, Fitbit). With the increasing number of wearable sensors and Internet of Things (IoT) devices, there is a growing trend towards collecting the activity data of users in real time. The key technology in HAR includes a sliding time window of time-series data captured with on-body sensors, manually designed feature extraction procedures, and a wide variety of supervised learning methods.

In the past years, researchers have made lots of progress in wearable activity recognition, using algorithms such as logistic regression [8], decision trees [9], and hidden Markov models [10]. Ref. [7] carried out experiments to evaluate the recognition performance of supervised and unsupervised machine learning techniques. For task identification, many of the traditional methods (i.e., hand-crafted features [11–13] and codebook approaches [14,15]) are characterized by manual feature extraction and discover features with expert knowledge. The performance of those methods often depends on the quality of the obtained feature representation. However, manual feature extraction is not always possible in practice, especially when we are unsure about the structure of the input data because of a lack of expert knowledge. Compared with manual feature extraction methods, deep learning techniques can discover adequate features without expert knowledge and systematic exploration of the feature space. In many fields, deep learning techniques have achieved remarkable results, such as in image recognition [16], speech recognition [17], natural language processing [18], and so on. Deep learning methods for HAR can be further divided into two categories: deep neural networks (DNNs) [19] and convolutional neural networks (CNNs) [20]. DNNs (i.e., containing an input layer, at least two hidden layers, and an output layer) can extract highly abstract features by stacking hidden layers. It allows us to add more possible connections between input and output neurons so that the ways to re-use learned features can be increased. Researchers who have used DNN methods for HAR include [21], who investigated deep neural networks with wearable sensor data, and [22], who explored temporal deep neural networks for active biometric authentication.

Recently, many researchers have adopted CNNs to deploy human activity recognition systems, such as [6,21,23,24]. CNNs can model the entire sequence by sharing the weights from local to global, extracting abstract features at hierarchical layers through a series of convolutional operations, and processing the raw activity signals to capture potential features. CNNs are based on the discovery of visual cortical cells and retain the spatial information of the data through a receptive field. It is known that the power of CNNs stems in large part from their ability to exploit symmetries through a combination of weight sharing and translation equivariance. Moreover, with their ability to act as feature extractors, a plurality of convolution operators is stacked to create a hierarchy of progressive abstract features. Ref. [20] proposed CNNs-based approaches to automatically extract discriminative features for HAR. More and more researchers are using variants of CNNs to learn sensor-based data representations for human activity recognition and have achieved remarkable performances. The model in [25] consists of two or three temporal-convolution layers with a ReLU activation function followed by a max-pooling layer and a softmax classifier, which can be applied over all sensors simultaneously. Yang et al. [23] introduce four temporal-convolutional layers on a single sensor, followed by a fully-connected layer and softmax classifier, showing that deeper networks can find correlations between different sensors.

However, all of the deep learning methods we mentioned above are identified by a single-path neural net without considering spatial and temporal features of the data. Inspired by the biological study of retinal ganglion cells in the primate visual system (Figure 1 illustrates the basic motivation source of our proposed ARN), there are parvocellular (P-cells) that provide good spatial detail and color in the visual system, but the resolution is very low. In addition, there are high-frequency magnocellular (M-cells), which are very sensitive to time changes but not sensitive to spatial details and colors. In this paper, inspired by the facts above, we propose a model to handle a set of activity data, synchronized by an asymmetric net, using a short time window to capture spatial features and a long time window to capture fine temporal features, corresponding to the P-cells and the M-cells, respectively. Our network is an end-to-end network, and the input of the network is the original sensor data. The data collected from the wearable device can be directly input into the network. The ARN model is applicable for supervised learning approaches and unsupervised learning approaches because it is based on ResNet [26], which has been proven to be applicable for supervised and unsupervised learning approaches. In this paper, we use a supervised learning method because it can make label information bridge the heterogeneous gap.

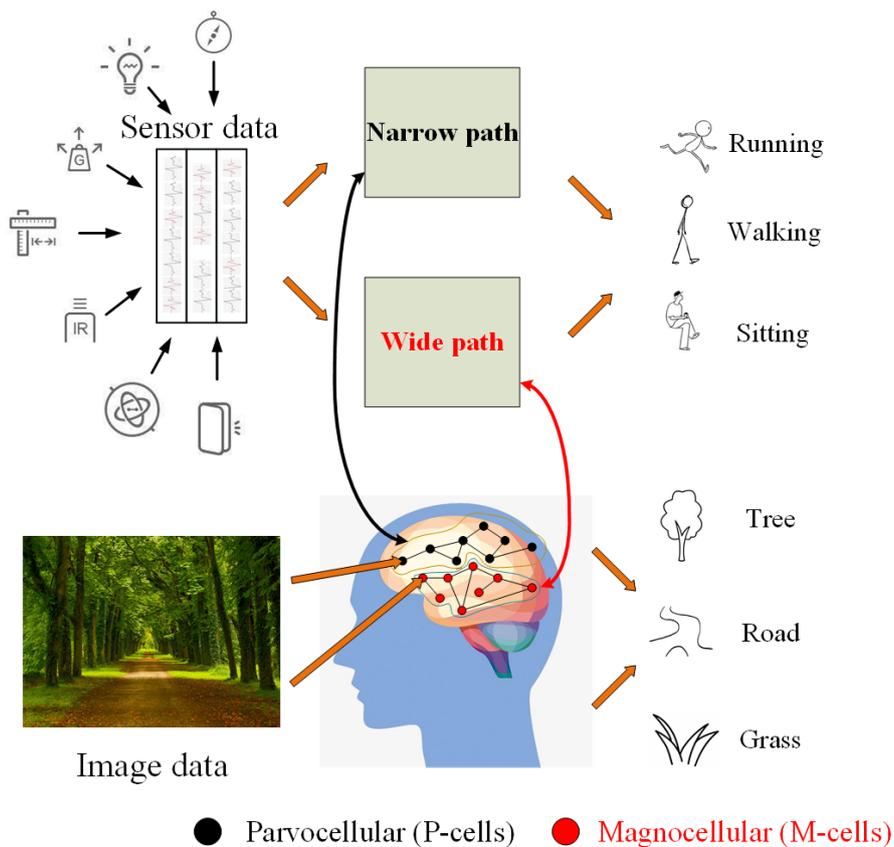


Figure 1. The basic motivation source of our proposed asymmetric residual network (ARN).

We propose a novel asymmetric residual network, named ARN. As a new kind of deep learning network, the components of activity recognition in ARN are divided into two parts: (1) a residual net using a short time window (i.e., 32 or 64); (2) a residual net using a long time window (i.e., 64 or 96). The lengths of the time windows ($(\text{Length}) = (\text{Time}) \times (\text{Sampling Frequency})$) are 32, 64, and 96. The last layer representations of two parts will be concatenated, then the fusion representations are used for accurate activity recognition. The superior advantages of the ARN over other existing methods are listed in Table 1. To the best of our knowledge, this is the first work that applies an asymmetric residual net for activity recognition.

The main contributions of the paper are as follows:

1. We propose a novel symmetric neural network based on ResNet for HAR, termed ARN, which is an asymmetric network and has two paths separately working at short and long slide windows. Our wide path is designed to capture global features but few spatial details, analogous to M-cells, and our narrow path is lightweight, similar to the small receptive field of P-cells.
2. We design a network that consists of an asymmetric residual net that not only can effectively manage information flow, but will also automatically learn effective activity feature representation, while capturing the fine feature distribution in different activities from wearable sensor data.
3. We compare the performance of our method to other relevant methods by carrying out extensive experiments on benchmark datasets. The results show that our method outperforms other methods.

The remainder of this paper is structured as follows. In Section 2, we briefly introduce related works. In Section 3, we highlight the motivation for our method and provide some theoretical analyses for its implementation. In Section 4, we introduce our experimental results and corresponding analysis, and finally, Section 5 concludes the paper.

Table 1. Comparison of different human activity recognition (HAR) technologies

Method	Manual f. ¹	High-Level f. ¹	Spatial f. ¹	Temporal f. ¹	Unsupervised	Supervised
HC [11]	✓	×	×	×	✓	×
CBH [14]	✓	×	×	×	✓	×
CBS [15]	✓	×	×	×	✓	×
AE [12]	✓	×	×	×	✓	×
MLP [13]	✓	×	×	×	✓	×
CNN [20]	×	✓	×	×	✓	✓
LSTM [27]	×	✓	×	×	×	✓
Hybrid [28]	✓	✓	×	×	×	✓
ResNet [26]	✓	✓	×	×	✓	✓
ARN (this work)	✓	✓	✓	✓	✓	✓

¹ f. denotes features.

2. Related Work

2.1. Methods for Human Activity Recognition

This section introduces the feature extraction methods for HAR selected in our comparative study. There are two main directions for HAR methods: conventional recognition methods and learning-based methods. In conventional methods, we extract features manually with expert knowledge. In learning-based methods, we can discover adequate features without expert knowledge and systematic exploration of the feature space. Conventional methods in our comparative study include hand-crafted features (HC) [11] and the codebook approach (CB) [14]. The learning-based methods include the autoencoders approach (AE) [12], multi-layer perceptron (MLP) [13], convolutional neural network (CNN) [20], long-short term memory networks (LSTM) [27], hybrid convolutional and recurrent networks (hybrid) [28], and deep residual learning (ResNet) [26].

2.1.1. Hand-Crafted Features

HC comprises simple metrics computed on data and uses simple statistical values (e.g., std, avg, mean, max, min, median, etc.) or frequency domain correlation features based on the signal Fourier transform to analyze the time series of human activity recognition data. Due to its simplicity to set up and low computational cost, it is still being used in some areas, but the accuracy cannot satisfy the requirements of modern AI games. In addition, when faced with the activity recognition of complex high-level behavior tasks, identifying the relevant features through these traditional approaches is time-consuming.

2.1.2. Codebook Approach

CB consists of two consecutive steps. The first step is codebook construction, which is to construct a codebook by using a cluster algorithm to process a set of subsequences extracted from the original data sequence. Each center of the cluster is considered a codeword that represents a distinct subsequence. The second step is codeword assignment, which aims to build a feature vector that is associated with a data sequence. Subsequences are first extracted from the sequence, and then each subsequence is assigned to the most similar codeword. Finally, a histogram-based feature representing the distribution of all codewords is built by using this information. During the codebook construction, a set of subsequences can be first extracted from the original data sequence by using a sliding time window approach with window size w and sliding stride h . Then, a k-means clustering [29] algorithm can be applied on subsequences to obtain n clusters of similar subsequences. The similarity metric

between two subsequences is the Euclidean distance. Finally, we can get a codebook that consists of n codewords.

During the codebook assignment, we should firstly extract subsequences from a sequence using the same sliding window approach as the codebook construction, and the most similar codeword needs to be assigned for each subsequence. Then, a histogram of the frequencies of codewords can be built by using this information. Finally, we can get a probabilistic feature presentation by normalizing the histogram.

The approach described above can be called codebook with hard assignment (CBH) because each subsequence is assigned to a codeword deterministically. However, this approach may lack flexibility in some uncertain situations where a subsequence is similar to two or more codewords. In order to solve this problem, we use a soft assignment variant (CBS). CBS can exploit kernel density estimation [15] to perform a smooth assignment of subsequences to multiple codewords. It allows us to obtain a feature that represents a smooth distribution of codewords and that considers the similarity between all codewords and subsequences.

2.1.3. Autoencoders Approach

AE is a specific architecture that consists of an encoder and a decoder as depicted in Figure 2. The encoder can project input data in a feature space of lower dimension, while the decoder can map the encoded features back to the input space. Then, the AE can reproduce the input data on the output according to a loss function like mean squared errors after training.

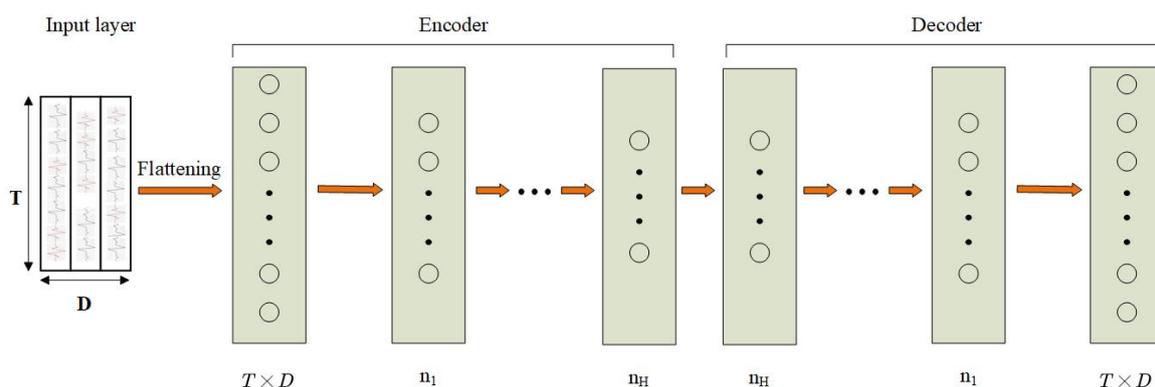


Figure 2. Architecture of an autoencoders (AE) approach for HAR. T in the input layer denotes the length of the time window. D denotes the number of sensor channels. H denotes the number of hidden layers.

2.1.4. Multi-Layer Perceptron

MLP is one of the most simplest neural networks. An important feature of the MLP is that it has multiple layers. As show in Figure 3, we call the first layer the input layer, the last layer the output layer, and the middle layers are called hidden layers. MLP does not specify the number of hidden layers, so you can choose the appropriate number of hidden layers according to your needs. Each neuron in a fully-connected layer takes the outputs of the previous layer as its inputs. Stacking layers can be seen as extracting features of an increasingly higher level of abstraction, and the output features of neurons at the n -th layer can be calculated by neurons at the $(n - 1)$ -th layer.

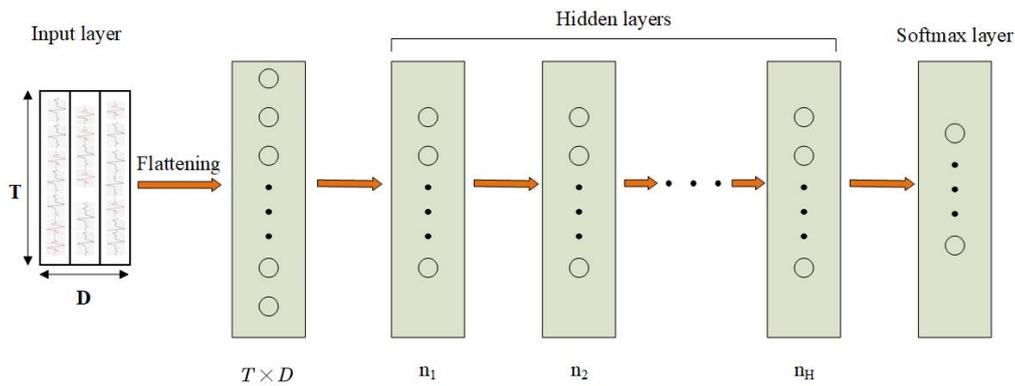


Figure 3. Architecture of a multi-layer perceptron (MLP) for HAR. T in the input layer denotes the length of the time window. D denotes the number of sensor channels. H denotes the number of hidden layers. Input data are first flattened into a $(T \times D)$ -dimensional vector. The vector is used as data for the hidden layers. All layers are fully connected.

2.1.5. Convolutional Neural Networks

CNNs can automatically extract the features from raw sensor data without the need for very professional expert knowledge [30]. A standard convolutional neural network consists of convolutional layers, max-pooling layers, fully-connection layers (FC), and a softmax layer. Instead of using predefined filters as in traditional feature extracting methods, CNNs can learn locally connected neurons that represent data-specific filters. As CNNs can share weights of neurons, the parameters of CNNs are much fewer than those of the traditional neural networks [31].

Convolutional layers are an important component of CNNs. Using several convolution filters (or kernels), which aim to learn feature representations of the raw input, complex operations can be easily performed by the convolution operation in the convolutional layer. The dimension of filters (or kernels) is determined by the input dimension. Convolution kernel is a function that generalizes a linear model for the underlying local patch. It works well for abstraction, when instances of latent concepts are linearly separable. In each convolutional layer, neurons of the current layer are connected to the neurons of the previous layer through a feature mapping operation. Thus, feature mapping of the upper layer can be obtained from the convolved results of the previous layer by adopting an element-wise nonlinear activation function. So, the value of the feature map j in the l -th layer, x_j^{l+1} , is calculated by:

$$x_j^{l+1} = \sigma \left(\sum_{i \in \text{Maps}} x_i^l \otimes K_{ij}^l + b_j^l \right), \quad (1)$$

where Maps are the total number of feature maps in the l -th layer, and b_j^l is a bias vector. $\sigma(\cdot)$ is the activation function to improve the performance of CNNs. The most notable non-linear activation function is ReLU, which is defined as $\sigma(x) = \max(x, 0)$. The ReLU activation operation allows networks to compute much faster than sigmoid or tanh activation functions, induces the sparsity in the hidden neurons, and allows networks to obtain sparse representations more easily. Adopting ReLU may bring zero value to affect the performance of backpropagation, but many research results have shown that ReLU [32] works much better than sigmoid and tanh [33].

Pooling layers, coming after the convolutional layer, are another component of CNNs. In the pooling layer, a pooling operation is used to reduce the number of neuron connections between neighboring convolutional layers, thus reducing computational complexity.

Fully-connected layers, which aim to convert the matrix-feature (2-D) to a vector-feature (1-D) for anastomosis classification tasks, contains about 90% of the parameters of the entire CNNs.

Loss functions play an important role in different classification tasks. The most common loss function is softmax. Given a training set $\{x^{(i)}, y^{(i)}; i \in [1, N], y^{(i)} \in [1, K]\}$, where $x^{(i)} \in \mathbb{R}^D$ is the input

patch, $y^{(i)}$ is the target label that belongs to the total number of labels (K). The prediction $a_j^{(i)}$ of the j -th class for the i -th input is transformed with the softmax function:

$$p(a^{(i)} = j) = \frac{e^{a_j^{(i)}}}{\sum_K e^{a_k^{(i)}}}. \quad (2)$$

Softmax normalizes the predictions to a probability distribution over the total classes. Softmax represents loss as follows:

$$\ell = -\frac{1}{N} \left(\sum_{l=1}^N \sum_{j=1}^K 1\{y^{(i)} = j\} \cdot \log p(a^{(i)} = j) \right). \quad (3)$$

Regularization is required in CNNs. Overfitting is an unavoidable problem in convolutional neural networks, but it can be effectively reduced by regularization. As a means of regularization, dropout can prevent the dependence of different neurons in a network and force the network to be more accurate, even in the absence of certain information.

2.1.6. Recurrent Neural Networks and Long-Short Term Memory Networks

Recurrent neural networks (RNNs) are a specific architecture where connections between neurons have directed cycles, and the output of the neurons is dependent on the state of the network at the previous timestamps. RNNs can find patterns with long-term dependencies because its specific behavior that memorizes the information extracted from past data. But in practice, there is a phenomenon called gradient explosion or vanish that will have a great affect on the performance of RNNs. The problem of vanishing or exploding gradients refers to the derivate of the error function with respect to the network weight becoming very large or close to zero [34]. This problem will result in an adverse impact on the weight update by the backpropagation algorithm. Therefore, LSTM is designed to solve the problem of vanishing or exploding gradients in RNNs. LSTM extends RNNs with memory cells and remembers information over time by storing it in an internal memory. The internal state can be updated and erased depending on their input and the state at the previous time step. As show in Figure 4, this mechanism is achieved by introducing an internal processor called a cell. A cell contains three gates called the input gate (i_t), output gate (o_t), and forget gate (f_t). c_t is the cell state. Gates are used to regulate the information update to the cell state. Their equations are mentioned below:

$$f_t = \sigma_f (W_f x_t + U_f h_{t-1} + b_f), \quad (4)$$

$$i_t = \sigma_i (W_i x_t + U_i h_{t-1} + b_i), \quad (5)$$

$$o_t = \sigma_o (W_o x_t + U_o h_{t-1} + b_o), \quad (6)$$

$$\tilde{c}_t = \sigma_c (W_c x_t + U_c h_{t-1} + b_c), \quad (7)$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t, \quad (8)$$

$$h_t = o_t \cdot \sigma_h (c_t), \quad (9)$$

where \cdot represents the element-wise multiplication of two vectors. x_t refers to the input vector to the LSTM cell at time t , and h_t is the hidden state vector. σ designates the activation function. W , U , and b are the matrices of weights and biases.

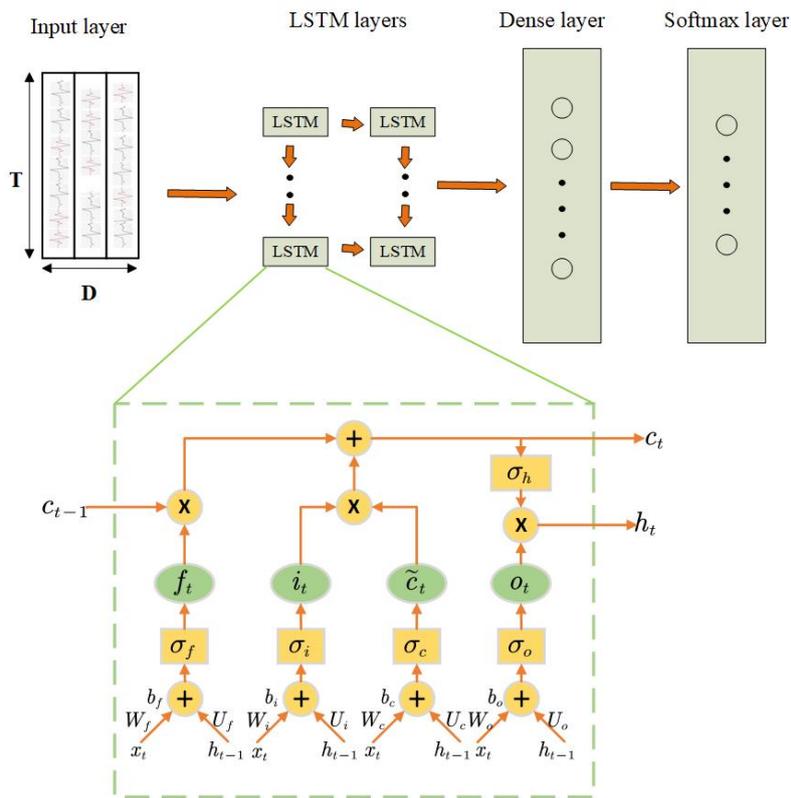


Figure 4. Architecture of a long-short term memory (LSTM) network for HAR. T in the input layer denotes the length of the time window. D denotes the number of sensor channels. Each LSTM layer is composed of an LSTM cell. \otimes denotes the element-wise multiplication. \oplus denotes the element-wise addition. The output of the last LSTM layer is passed to a dense and a softmax layer.

2.1.7. Hybrid Convolutional and Recurrent Networks

Hybrids comprise convolutional, LSTM, and softmax layers as depicted in Figure 5. Convolutional layers have the ability to extract the features from input data and create a hierarchy of progressively more abstract features by stacking several convolutional operators. LSTM includes a memory to model temporal dependencies in time series problems. Therefore, the combination of CNN and LSTM can capture time dependencies on features extracted by convolutional operations.

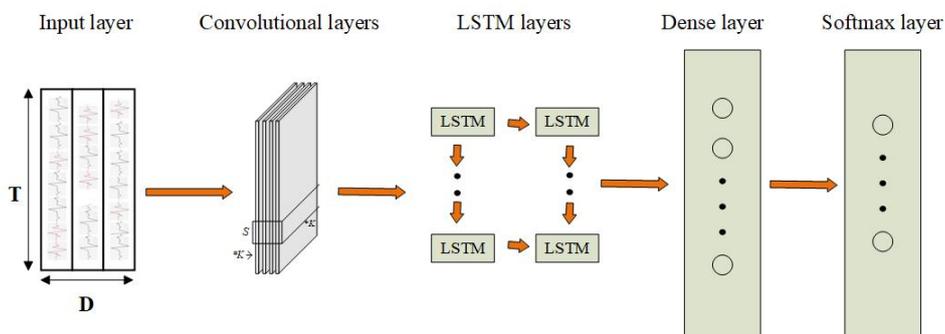


Figure 5. Architecture of a hybrid for human activity recognition. T in the input layer denotes the length of the time window. D denotes the number of sensor channels. In convolutional layers, K denotes the kernels in layers, and S is the length of kernels. The convolutional layers can extract features from input data and provide abstract presentations of data in feature maps. LSTM layers learn temporal dynamics from data.

2.1.8. Deep Residual Learning

ResNet is designed to address the degradation problem of the accuracy of training sets getting saturated or even decreasing with the network depth increasing. Different from the ordinary convolutional neural network, ResNet has many stacked residual blocks in which identity mappings are added to connect input and output. Residual blocks with identity mapping can be expressed in a general form:

$$y_l = h(x_l) + F(x_l, W_l), \tag{10}$$

$$x_{l+1} = f(y_l), \tag{11}$$

where x_{l+1} and x_l are the output and input of the l -th unit, F is a residual function, and W_l are parameters of the unit. $h(x_l) = x_l$ is an identity mapping, and f is a ReLU activation function. The key idea of ResNet is to learn the additive residual function F with respect to $h(x_l)$. ResNet can make the element-wise addition on input and output by attaching a shortcut connection. This simple addition can increase the training speed of the model and improve the training effect, and will not add additional parameters to the network. With the network depth increasing, this simple structure is a good solution to the degradation problem.

3. Asymmetric Residual Network

Our proposed model has a narrow path (see Section 3.2) and a wide path(see Section 3.3), which are concatenated and sent to the fully-connected layer. A loss function is introduced in Section 3.5.

3.1. Network Architecture

As shown in Figure 6, convolutional layers and residual layers in our architecture are used to model the recognition task. In convolutional layers, the general activity features are extracted from raw sensor data. In residual layers, the special features can be extracted from general features, and the special features are used for human activity recognition.

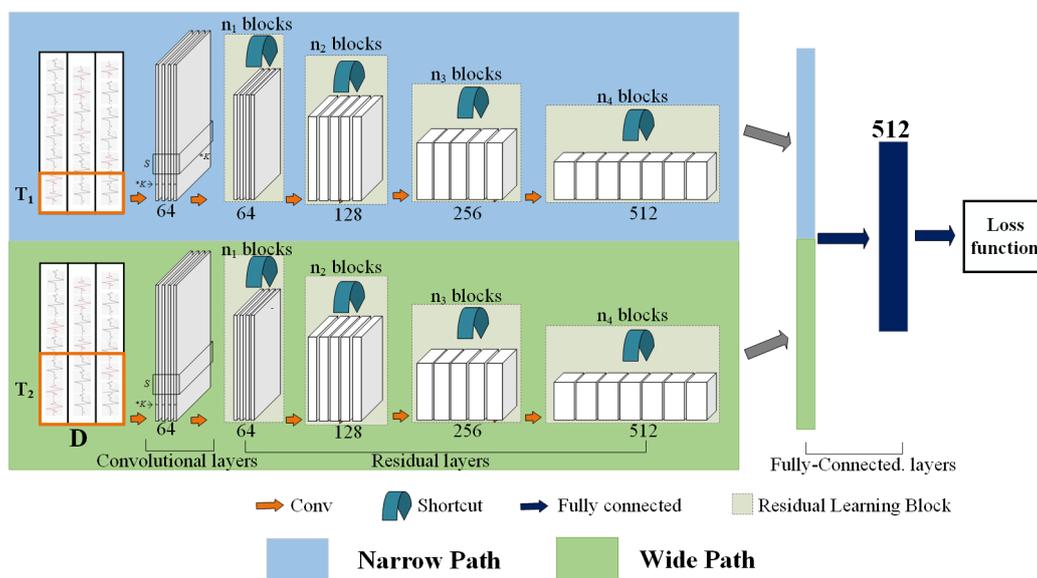


Figure 6. The proposed ARN architecture for human activity recognition. T_1 and T_2 in the raw data layer denote the length of the time window corresponding to the narrow path and wide path, respectively ($T_1 < T_2$). D denotes the number of sensor channels. In the conv. layers, K denotes the kernels in the layer, and the length of kernels is $s = 5$. In the res. layers, the dimension of each block doubles feature maps to the input signal, which is processed by a number of preset building blocks for residual learning.

The convolutional layers (i.e., conv. layers) of our architecture consist of 1 layer, including 64 sliding windows (filters) whose size is $s = 5 \times 1$, a batch normalization layer, and a ReLU layer with the use of a pooling layer [30]. The residual layers contain four “blocks”. The details of the residual block are shown in Table 2, and the values of n_1, n_2, n_3, n_4 are set to 3, 4, 6, 3, respectively.

Table 2. The layer parameters of the ARN model.

Stage	Narrow Path	Wide Path
Conv1	$64 \times (5, 1)$	$64 \times (5, 1)$
Max-pooling	/2	/2
res1	$\begin{bmatrix} 1 \times 1 & 64 \\ 3 \times 1 & 64 \\ 1 \times 1 & 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1 & 64 \\ 3 \times 1 & 64 \\ 1 \times 1 & 256 \end{bmatrix} \times 3$
res2	$\begin{bmatrix} 1 \times 1 & 512 \\ 3 \times 1 & 512 \\ 1 \times 1 & 2048 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1 & 512 \\ 3 \times 1 & 512 \\ 1 \times 1 & 2048 \end{bmatrix} \times 4$
res3	$\begin{bmatrix} 1 \times 1 & 256 \\ 3 \times 1 & 256 \\ 1 \times 1 & 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 & 256 \\ 3 \times 1 & 256 \\ 1 \times 1 & 1024 \end{bmatrix} \times 6$
res4	$\begin{bmatrix} 1 \times 1 & 128 \\ 3 \times 1 & 128 \\ 1 \times 1 & 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1 & 128 \\ 3 \times 1 & 128 \\ 1 \times 1 & 512 \end{bmatrix} \times 3$
concat	global average pool	global average pool
fc.	512	512

3.2. Narrow Path

The narrow path can be any convolutional model (e.g., ref. [35] introduced a new two-stream inflated 3D convolutional network: filters and pooling kernels of very deep image classification convolutional networks are expanded into 3D, [36] introduced spatiotemporal ResNets as a combination of two-stream convolutional networks and ResNets, [37] introduced non-local operations as a generic family of building blocks for capturing long-range dependencies) that works on sequence data as a spatiotemporal volume. The key concept in our narrow path is a short slide time window to scan the sequence activity data. We know from [38] that feature learning methods can obtain good performance when $T = 32, 64, 96$. Therefore, a typical value of T we study is 32 [38]. Denoting the number of sensor channels as S , the raw clip length is $T \times S$. The function of this path is to throw compact information into the net, the purpose is to capture spatial features.

3.3. Wide Path

In parallel to the narrow path, the wide path is another convolutional model with a long slide time window. The operations of the two paths’ nets work on the same raw activity data sequences, so the wide path uses an αT slide time window, α times longer than the narrow path. A typical value is $\alpha = 3$ [39] in our experiments. The presence of α is in the key of the NarrowWide concept. It explicitly indicates that the two paths work on different time windows. Our wide path enters a long sequence of activity data into the net in order to pursue global functionality throughout the net hierarchy. Our wide path is distinguished from existing methods in that it can use a significantly lower channel capacity to achieve good accuracy for the ARN model. The low channel capacity can also be interpreted as a weaker ability to represent spatial semantics. Our wide path not only has a long slide time window, but also pursues high-dimension features throughout the network hierarchy, maintaining temporal fidelity as much as possible.

3.4. Lateral Concatenation

Our lateral concatenation fuses from the narrow path to the wide path. We denote the representation shape of the narrow path as $\{T, S\}$, and the representation shape of the wide path is $\{\alpha T, S\}$. The output of the lateral concatenation is fused into the narrow path by concatenation. Therefore, the shape of the concatenation layer is $\{(1 + \alpha)T, S\}$.

3.5. Loss Function

In order to train classification models, classification objectives (such as logistic loss and softmax loss) have been widely explored. For accurate human activity recognition, using labels that are different from the ground-truth for prediction cannot contribute to the update of the network parameters. For depth estimates, predictions that are close to the ground-truth labels also help to update network parameters. In this work, we employ softmax loss for training the human activity recognition model. For each training sequence x , the probability of each label $k \in \{1, 2, \dots, \mathcal{K}\}$ in our model is computed via softmax:

$$p(k|x) = \frac{\exp\{z_k\}}{\sum_{i=1}^{\mathcal{K}} \exp\{z_i\}}, \quad (12)$$

where z_i are the logits or unnormalized log probabilities. Here, the z_i are computed by adding a fully-connected layer on top of the sequence data embedding, i.e., $z_i = W_i^T \phi(x) + b_i$, where W_i and b_i are the weight and bias for the target label, respectively. Let $q(k|x)$ denote the ground-true distribution over classes for this training example such that $\sum_{i=1}^{\mathcal{K}} q(k|x) = 1$. The cross-entropy loss for the example is computed as

$$\ell = - \sum_{k=1}^{\mathcal{K}} \log(p(k|x))q(k|x). \quad (13)$$

4. Experiments

In order to demonstrate the performance of our proposed ARN method, we carried out extensive experiments on two widely used benchmark datasets, i.e., OPPORTUNITY and UniMiB-SHAR, to verify the effectiveness of our method.

4.1. Dataset

Human activity features are usually unique and cyclical, and natural human activities include walking, running, jumping, and so on. Therefore, a set of active data that includes a variety of types of natural human activities should be considered in dataset construction.

We use benchmark datasets to validate the model performance and use different action sequences to verify whether they belong to the same person. There are many benchmark activity datasets, such as OPPORTUNITY [40], WISDM [8], UniMiB-SHAR [41], MHEALTH [42], and PAMAP2 [43]. In this paper, we evaluate our method by using the following two datasets.

The OPPORTUNITY dataset has been widely used in research. It contains four subjects performing 17 different (morning) activities of daily living (ADLs) in a sensor-rich environment, as listed in Tables 3 and 4. These were acquired at a sampling frequency of 30 Hz equipping 7 wireless body-worn inertial measurement units (IMUs). Each IMU consists of a 3D accelerometer, 3D gyroscope, and a 3D magnetic sensor, as well as 12 additional 3D accelerometers placed on the back, arms, ankles, and hips, accounting for a total of 145 different sensor channels. During the data collection process, each subject performed 5 ADL sessions and 1 drill session. During each ADL session, subjects were asked to naturally perform the activities named "ADL1" to "ADL5". During the drill sessions, subjects performed 20 repetitions of each of the 17 ADLs in the dataset. The dataset contains about 6 h of information in total, and the data are labeled on a timestamp level. The dataset can be used in an open activity recognition challenge where participants compete to achieve the highest performance on the

recognition. In our experiment, the training and testing sets had 63 dimensions (36-D on the hand, 9-D on the back, and 18-D on the ankle).

The UniMiB-SHAR dataset has data from 30 healthy subjects (6 male and 24 female) acquired using the 3D accelerometer on a Samsung Galaxy Nexus I9250 with Android OS version 5.1.1. It contains 11,771 samples of both human activities and falls performed by 30 subjects aged from 18 to 60. The data are sampled at a constant sampling rate of 50 Hz and split into 17 different activity classes, 9 safety activities and 8 dangerous activities (e.g., a falling action), as shown in Tables 3–5. Unlike the OPPORTUNITY dataset, this dataset does not have a NULL class and remains relatively balanced. It allows researchers to work to more robust features and classification schemes. In our experiments, the training and testing sets had 3 dimensions.

Table 3. The details of the experimental datasets.

Datasets	OPPORTUNITY	UniMiB-SHAR
Types of sensors	Custom bluetooth wireless accelerometers, gyroscopes, Sun SPOts and InertiaCube3, Ubisense localization system, a custom-made magnetic field sensor	A Bosh BMA220 acceleration sensor
Number of sensors	72	3
Number of samples	473 K	11,771
Acquisition periods	10–20 (min)	0.6 (min)

Table 4. Classes and proportions for the OPPORTUNITY dataset.

Class	Proportion	Class	Proportion
Open Door 1/2	1.87%/1.26%	Open Fridge	1.60%
Close Door 1/2	6.15%/1.54%	Close Fridge	0.79%
Open Dishwasher	1.85%	Close Dishwasher	1.32%
Open Drawer 1/2/3	1.09%/1.64%/0.94%	Clean Table	1.23%
Close Drawer 1/2/3	0.87%1.69%/2.04%	Drink from Cup	1.07%
Toggle Switch	0.78%	NULL	72.28%

Table 5. Classes and proportions for the UniMiB-SHAR dataset.

Class	Proportion	Class	Proportion
StandingUpfromSitting	1.30%	Walking	14.77%
StandingUpfromLaying	1.83%	Running	16.86%
LyingDownfromStanding	2.51%	Going Up	7.82%
Jumping	6.34%	Going Down	11.25%
F(alling) Forward	4.49%	F and Hitting Obstacle	5.62%
F Backward	4.47%	Syncope	4.36%
F Right	4.34%	F with ProStrategies	4.11%
F Backward SittingChair	3.69%	F Left	4.54%
Sitting Down	1.70%		

The OPPORTUNITY dataset and UniMiB-SHAR dataset were collected from real environment. The two datasets have their own characteristics and contain different sensors. The UniMiB-SHAR dataset only contains the accelerometer data, and it has a low power cost. The OPPORTUNITY dataset combines accelerometers, gyroscopes, and magnetic sensor data, and it can provide accurate limb orientation.

4.2. Baseline

We compared our proposed ARN method against some classic or state-of-the-art activity recognition methods. We roughly divided these methods into categories: conventional recognition methods include HC [11], CBH [14], and CBS [15]. The learning-based methods include AE [12], MLP [13], CNN [20], LSTM [27], hybrid [28], and ResNet [26]. As in conventional methods, we used hand-crafted features; readers can find more details in [38]. For learning-based methods, we used raw activity data as input. Following [38], the hyper-parameters of these learning-based baseline models, except ResNet (The hyper-parameters used by ResNet are the hyper-parameters used in one of the paths in the proposed ARN model.), for the OPPORTUNITY and UniMiB-SHAR datasets are provided in Table 6.

Table 6. Hyper-parameters of the learning-based methods on the OPPORTUNITY dataset and the UniMiB-SHAR dataset.

Model	Parameters
AE	5000 ^a 5000 ^a
MLP	2000 ^a 2000 ^a 2000 ^a
CNN	((11,1),(1,1),50,(2,1)) ^b ((10,1),(1,1),40,(3,1)) ^b ((6,1),(1,1),30,(1,1)) ^b 1000 ^a
LSTM	(64,600) ^c (64,600) ^c 512 ^a
Hybrid	((11,1),(1,1),50,(2,1)) ^b (27,600) ^c (27,600) ^c 512 ^a

^a dense units. ^b ((kernel size), (siding stride), number of kernels, (pool)). ^c (number of cells, output-dim).

4.3. Implementation and Setting

Our ARN model was implemented in TensorFlow [44], a system that transfers complex data structures to artificial intelligence neural networks for analysis and processing. The computing platform was equipped with an Intel 2 × Intel E5-2600 CPU, 128G RAM, and a NVIDIA TITAN Xp 12G GPU. The model was trained using the ADADELTA gradient decent algorithm with default parameters (i.e., initial learning rate of 1), for 50 epochs. The batch size was set to 128. The hyper-parameters of the proposed model are provided in Table 2.

Sliding Time Window Size: The length of the sliding window T is an important hyper-parameter of the proposed model. As in baseline methods, we carried out two more comparative studies using $T = 32$ (approximately 1s), $T = 64$ (approximately 2s), and $T = 96$ (approximately 3s). For the proposed model, we use $T = 32$ or $T = 64$ as the hyper-parameter of the narrow path and $T = 64$ or $T = 96$ as the hyper-parameter of the wide path.

4.4. Performance Measure

ADL datasets are often highly unbalanced. The OPPORTUNITY dataset is extremely unbalanced, as the NULL class represents more than 75% of the recorded data. For this dataset, the overall classification accuracy is not an appropriate measure of performance, because the activity recognition rate of the majority classes might skew the performance statistics to the detriment of the least

represented classes. As a result, many previous works such as [30] show the use of an evaluation metric independent of the class repartition— F_1 -score. The F_1 -score combines two measures, the precision p and the recall r : p is the number of correct positive examples divided by the number of all positive examples returned by the classifier, and r is the number of correct positive results divided by the number of all positive samples. The F_1 -score is the harmonic average of p and r , where the best value is at 1 and the worst at 0. In this paper, we use an additional evaluation metric to make the comparisons easier, the weighted F_1 -score (sum of class F_1 -scores, weighted by the class proportion):

$$F_w = 2 \sum_G w_g \frac{p_g \cdot r_g}{p_g + r_g}, \quad (14)$$

where $w_g = N_g/N_{total}$, N_g is the number of samples in class g , and N_{total} is the total number of samples.

4.5. Results and Discussion

In this section, we present and discuss the results. To get insight into how these methods are applied to the domain, we show the performance of these methods and evaluate some key parameters.

The weighted F_1 -score of all models on OPPORTUNITY and UniMiB-SHAR are listed in Table 7. Results on these datasets show that the proposed ARN method substantially outperforms all other methods against which it was compared. Compared to conventional recognition methods, such as CBS, the best conventional method achieves an absolute boost of 4.98% and 14.65% corresponding to the OPPORTUNITY dataset and the UniMiB-SHAR dataset, respectively. In addition, most of the learning-based recognition methods outperform the conventional recognition methods. In particular, for the OPPORTUNITY dataset, the hybrid method achieves the best performance among all the learning-based methods. Compared to the hybrid method, our ARN method achieves boosts of 2.4%. For the UniMiB-SHAR dataset, the MLP method achieves the best performance among all the learning-based methods. Compared to the MLP method, our ARN method achieves a boost of 2.48%. We also compared a single-path residual network, i.e., ResNet, and our ARN achieves an absolute boost of 1.26% and 1.32% corresponding to the OPPORTUNITY dataset and the UniMiB-SHAR dataset, respectively.

From Table 7, we can observe that the gap between the learning-based methods and conventional methods is larger for the UniMiB-SHAR dataset than the OPPORTUNITY dataset. The reason is that the sensor channels in the OPPORTUNITY dataset are more numerous than those in the UniMiB-SHAR dataset. By carefully comparing the results, we found that our proposed method showed a higher degree of performance improvement when tested on the UniMiB-SHAR dataset compared to the OPPORTUNITY dataset. This means that our method is effective.

We also observe from Table 7 that different lengths of slide time windows have an impact on the performance of activity recognition. The short time window contains too little information. With the growth of the time window, the window contains more and more information, and the accuracy is improved accordingly. But using a longer slide time window does not yield better recognition performances [38]. Most methods perform best in human activity recognition tasks when $T = 64$. The reason is that longer frames potentially contain data related to a higher number of activities, making their majority-labeling more inaccurate.

Table 7. Weighted F_1 -score performances of different methods on the OPPORTUNITY and UniMiB-SHAR datasets. (n) and (w) denote the narrow and wide paths, respectively.

Method	T (Time Window)	OPPORTUNITY	UniMiB-SHAR
HC [11]	32	84.95	22.83
	64	85.56	22.19
	96	85.69	21.96
CBH [14]	32	84.37	64.51
	64	85.21	65.03
	96	84.66	64.36
CBS [15]	32	85.53	67.54
	64	86.01	67.97
	96	85.39	67.36
AE [12]	32	82.87	68.37
	64	84.54	68.24
	96	83.39	68.39
MLP [13]	32	87.32	73.33
	64	87.34	75.36
	96	86.65	74.82
CNN [20]	32	87.51	74.01
	64	88.03	73.04
	96	87.62	73.36
LSTM [27]	32	85.33	69.24
	64	86.89	69.49
	96	86.21	68.81
Hybrid [28]	32	87.91	73.19
	64	88.17	73.22
	96	87.67	72.26
ResNet [26]	32	88.91	76.19
	64	89.17	76.22
	96	87.67	75.26
ARN	32–96 (n)-(w)	90.29	76.39

4.6. Hyper-Parameter Evaluation

Impact of the length of the narrow and wide path selection: The model we proposed has two paths. One is the narrow path (i.e., the length of the slide time window is short), and the other is the wide path (i.e., the length of the slide window is wide). In order to verify the impact of different lengths of slide time window combinations on the results, we leveraged a combination of slide time windows of different lengths for comparison experiments (i.e., 32–64, 32–96, 64–96). ARN_1, ARN_2, and ARN_3 refer to the 32–64, 32–96, and 64–96 slide time window combinations, respectively. We carried out experiments on the two datasets. The weighted F_1 -score results are shown in Table 8. From Table 8, we can observe that on the OPPORTUNITY dataset, ARN_2 outperforms ARN_1 and ARN_3; the minimum image size is $63(D)*32(T)$, and the accuracy is 90.29%. For the UniMiB-SHAR dataset, ARN_1 outperforms ARN_2 and ARN_3; the minimum image size is $3(D)*32(T)$, and the accuracy is 76.39%. The performance gap between the three experiments was very small. This indicates that ARN is a stable model that is not sensitive to the lengths of slide time windows.

Table 8. Weighted F_1 -score performances comparison of ARN with the combinations of different slide window lengths on the OPPORTUNITY and UniMiB-SHAR datasets. (n) and (w) denote the narrow and wide paths, respectively.

Method	T (n)-(w) (Time Window)	OPPORTUNITY	UniMiB-SHAR
ARN_1	32–64	90.21	77.23
ARN_2	32–96	90.29	76.39
ARN_3	64–96	90.19	76.04

5. Conclusions

In this paper, we propose a novel asymmetric residual network for activity recognition using wearable device data, named ARN. To improve the accuracy of activity recognition, our method consists of two paths. The first path uses a short time window to capture spatial features, and the second path uses a long time window to capture fine temporal features. Unlike other learning-based methods, ARN considers the spatial and temporal features of the data at the same time. It can effectively manage information flow and automatically learn activity feature representation. ARN is an end-to-end network, and the data collected by the wearable device can be directly input into the network. Comprehensive experiments on the two benchmark human activity recognition datasets demonstrate that the ARN outperforms the compared baselines. This method has a good application prospect. However, biometric identification, fingerprint recognition, iris recognition, and other technologies have achieved more than 98% accuracy and are widely used in people's life. The accuracy of ARN cannot meet the requirements of the practical applications of the recognition field in society. Therefore, there is still a lot of room for progress.

For future work, we may research how to extract more fine-gained features by using an attention strategy and focus on the research of data dynamic fusion algorithms for maximizing the retention of the data features and obtaining higher recognition accuracy. In addition, we will recognize dangerous human activities, but this type of activity recognition involves many fine-gained feature extractions. Therefore, we may take advantage of the memory mechanism to design a memory-augmented neural network [45] that can learn to find supporting pre-stored clues (i.e., representations of historical activities or others).

Author Contributions: Conceptualization, J.L. and W.S.; formal analysis, Z.Y. and W.S.; funding acquisition, J.L.; investigation, W.S. and Z.Y.; methodology, J.L. and W.S.; project administration, Z.Y. and W.S.; software, O.I. and W.S.; supervision, Z.Y.; validation, J.L.; visualization, W.S.; writing (original draft), Z.Y. and W.S.; writing (review and editing), W.S. and Z.Y.

Funding: This work was supported in part by the Key Technology R&D Program of Hunan Province (2018GK2052) and the Science and Technology Plan of Hunan (2016TP1003).

Acknowledgments: The authors would also like to thank the associate editor and anonymous reviewers for their comments to improve the paper.

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

References

1. Sukor, A.S.A.; Zakaria, A.; Rahim, N.A.; Kamarudin, L.M.; Setchi, R.; Nishizaki, H. A hybrid approach of knowledge-driven and data-driven reasoning for activity recognition in smart homes. *J. Intell. Fuzzy Syst.* **2019**, *36*, 4177–4188. [[CrossRef](#)]
2. Xiao, Z.; Lim, H.B.; Ponnambalam, L. Participatory Sensing for Smart Cities: A Case Study on Transport Trip Quality Measurement. *IEEE Trans. Ind. Inform.* **2017**, *13*, 759–770. [[CrossRef](#)]
3. Fortino, G.; Ghasemzadeh, H.; Gravina, R.; Liu, P.X.; Poon, C.C.Y.; Wang, Z. Advances in multi-sensor fusion for body sensor networks: Algorithms, architectures, and applications. *Inf. Fusion* **2019**, *45*, 150–152. [[CrossRef](#)]

4. Qiu, J.X.; Yoon, H.; Fearn, P.A.; Tourassi, G.D. Deep Learning for Automated Extraction of Primary Sites From Cancer Pathology Reports. *IEEE J. Biomed. Health Inform.* **2018**, *22*, 244–251. [[CrossRef](#)] [[PubMed](#)]
5. Oh, I.; Cho, H.; Kim, K. Playing real-time strategy games by imitating human players' micromanagement skills based on spatial analysis. *Expert Syst. Appl.* **2017**, *71*, 192–205. [[CrossRef](#)]
6. Lisowska, A.; O'Neil, A.; Poole, I. Cross-cohort Evaluation of Machine Learning Approaches to Fall Detection from Accelerometer Data. In Proceedings of the 11th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2018)—Volume 5: HEALTHINF, Funchal, Madeira, Portugal, 19–21 January 2018; pp. 77–82. [[CrossRef](#)]
7. Attal, F.; Mohammed, S.; Dedabrishvili, M.; Chamroukhi, F.; Oukhellou, L.; Amirat, Y. Physical Human Activity Recognition Using Wearable Sensors. *Sensors* **2015**, *15*, 31314–31338. [[CrossRef](#)] [[PubMed](#)]
8. Kwapisz, J.R.; Weiss, G.M.; Moore, S. Activity recognition using cell phone accelerometers. *SIGKDD Explor.* **2010**, *12*, 74–82. [[CrossRef](#)]
9. Bao, L.; Intille, S.S. Activity Recognition from User-Annotated Acceleration Data. In Proceedings of the Pervasive Computing, Second International Conference—PERVASIVE 2004, Vienna, Austria, 21–23 April 2004; pp. 1–17. [[CrossRef](#)]
10. Ordóñez, F.J.; Englebienne, G.; de Toledo, P.; van Kasteren, T.; Sanchis, A.; Kröse, B.J.A. In-Home Activity Recognition: Bayesian Inference for Hidden Markov Models. *IEEE Pervasive Comput.* **2014**, *13*, 67–75. [[CrossRef](#)]
11. Ramamurthy, S.R.; Roy, N. Recent trends in machine learning for human activity recognition—A survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2018**, *8*. [[CrossRef](#)]
12. Gu, F.; Khoshelham, K.; Valaee, S.; Shang, J.; Zhang, R. Locomotion Activity Recognition Using Stacked Denoising Autoencoders. *IEEE Internet Things J.* **2018**, *5*, 2085–2093. [[CrossRef](#)]
13. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 448–456.
14. Shirahama, K.; Grzegorzec, M. On the Generality of Codebook Approach for Sensor-Based Human Activity Recognition. *Electronics* **2017**, *6*, 44. [[CrossRef](#)]
15. Van Gemert, J.C.; Veenman, C.J.; Smeulders, A.W.M.; Geusebroek, J. Visual Word Ambiguity. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1271–1283. [[CrossRef](#)] [[PubMed](#)]
16. Yang, Z.; Raymond, O.I.; Sun, W.; Long, J. Deep Attention-Guided Hashing. *IEEE Access* **2019**, *7*, 11209–11221. [[CrossRef](#)]
17. Sarkar, A.; Dasgupta, S.; Naskar, S.K.; Bandyopadhyay, S. Says Who? Deep Learning Models for Joint Speech Recognition, Segmentation and Diarization. In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing, Calgary, AB, Canada, 15–20 April 2018; pp. 5229–5233. [[CrossRef](#)]
18. Kumar, A.; Irsoy, O.; Ondruska, P.; Iyyer, M.; Bradbury, J.; Gulrajani, I.; Zhong, V.; Paulus, R.; Socher, R. Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. In Proceedings of the 33rd International Conference on Machine Learning, New York City, NY, USA, 19–24 June 2016; pp. 1378–1387.
19. Shi, Z.; Zhang, J.A.; Xu, R.; Fang, G. Human Activity Recognition Using Deep Learning Networks with Enhanced Channel State Information. In Proceedings of the IEEE Global Communications Conference, Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–6. [[CrossRef](#)]
20. Xu, W.; Pang, Y.; Yang, Y.; Liu, Y. Human Activity Recognition Based On Convolutional Neural Network. In Proceedings of the 24th International Conference on Pattern Recognition, Beijing, China, 20–24 August 2018; pp. 165–170. [[CrossRef](#)]
21. Hammerla, N.Y.; Halloran, S.; Plötz, T. Deep, Convolutional, and Recurrent Models for Human Activity Recognition Using Wearables. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, New York, NY, USA, 9–15 July 2016; pp. 1533–1540.
22. Neverova, N.; Wolf, C.; Lacey, G.; Fridman, L.; Chandra, D.; Barbello, B.; Taylor, G.W. Learning Human Identity From Motion Patterns. *IEEE Access* **2016**, *4*, 1810–1820. [[CrossRef](#)]
23. Yang, J.; Nguyen, M.N.; San, P.P.; Li, X.; Krishnaswamy, S. Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015; pp. 3995–4001.
24. Yang, Z.; Raymond, O.I.; Zhang, C.; Wan, Y.; Long, J. DFTerNet: Towards 2-bit Dynamic Fusion Networks for Accurate Human Activity Recognition. *IEEE Access* **2018**, *6*, 56750–56764. [[CrossRef](#)]

25. Ronao, C.A.; Cho, S. Deep Convolutional Neural Networks for Human Activity Recognition with Smartphone Sensors. In Proceedings of the Neural Information Processing—22nd International Conference, Istanbul, Turkey, 9–12 November 2015; pp. 46–53. [\[CrossRef\]](#)
26. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [\[CrossRef\]](#)
27. Milenkoski, M.; Trivodaliev, K.; Kalajdziski, S.; Jovanov, M.; Stojkoska, B.R. Real time human activity recognition on smartphones using LSTM networks. In Proceedings of the 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, Opatija, Croatia, 21–25 May 2018; pp. 1126–1131. [\[CrossRef\]](#)
28. Meng, B.; Liu, X.; Wang, X. Human action recognition based on quaternion spatial-temporal convolutional neural network and LSTM in RGB videos. *Multimed. Tools Appl.* **2018**, *77*, 26901–26918. [\[CrossRef\]](#)
29. Han, J.; Kamber, M.; Pei, J. *Data Mining: Concepts and Techniques*, 3rd ed.; Morgan Kaufmann: Burlington, MA, USA, 2011.
30. Morales, F.J.O.; Roggen, D. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors* **2016**, *16*, 115. [\[CrossRef\]](#)
31. Baldi, P.; Sadowski, P.J. A theory of local learning, the learning channel, and the optimality of backpropagation. *Neural Netw.* **2016**, *83*, 51–74. [\[CrossRef\]](#)
32. Nair, V.; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel, 21–24 June 2010; pp. 807–814.
33. Glorot, X.; Bordes, A.; Bengio, Y. Deep Sparse Rectifier Neural Networks. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 315–323.
34. Pascanu, R.; Mikolov, T.; Bengio, Y. On the difficulty of training recurrent neural networks. In Proceedings of the 30th International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 1310–1318.
35. Carreira, J.; Zisserman, A. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4724–4733. [\[CrossRef\]](#)
36. Feichtenhofer, C.; Pinz, A.; Wildes, R.P. Spatiotemporal Residual Networks for Video Action Recognition. In Proceedings of the Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, Barcelona, Spain, 5–10 December 2016; pp. 3468–3476.
37. Wang, X.; Girshick, R.B.; Gupta, A.; He, K. Non-Local Neural Networks. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7794–7803. [\[CrossRef\]](#)
38. Li, F.; Shirahama, K.; Nisar, M.A.; Köping, L.; Grzegorzec, M. Comparison of Feature Learning Methods for Human Activity Recognition Using Wearable Sensors. *Sensors* **2018**, *18*, 679. [\[CrossRef\]](#)
39. Feichtenhofer, C.; Fan, H.; Malik, J.; He, K. SlowFast Networks for Video Recognition. *arXiv* **2018**, arXiv:1812.03982
40. Roggen, D.; Calatroni, A.; Rossi, M.; Holleczeck, T.; Förster, K.; Tröster, G.; Lukowicz, P.; Bannach, D.; Pirkel, G.; Ferscha, A.; et al. In Proceedings of the Collecting complex activity datasets in highly rich networked sensor environments. Seventh International Conference on Networked Sensing Systems, Kassel, Germany, 15–18 June 2010; pp. 233–240. [\[CrossRef\]](#)
41. Micucci, D.; Mobilio, M.; Napoletano, P. UniMiB SHAR: A new dataset for human activity recognition using acceleration data from smartphones. *arXiv* **2016**, arXiv:1611.07688.
42. Baños, O.; García, R.; Terriza, J.A.H.; Damas, M.; Pomares, H.; Ruiz, I.R.; Saez, A.; Villalonga, C. mHealthDroid: A Novel Framework for Agile Development of Mobile Health Applications. In Proceedings of the Ambient Assisted Living and Daily Activities—6th International Work-Conference, Belfast, UK, 2–5 December 2014; pp. 91–98. [\[CrossRef\]](#)
43. Reiss, A.; Stricker, D. Creating and benchmarking a new dataset for physical activity monitoring. In Proceedings of the 5th International Conference on Pervasive Technologies Related to Assistive Environments, Heraklion, Greece, 6–9 June 2012; p. 40. [\[CrossRef\]](#)

44. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv* **2016**, arXiv:1603.04467.
45. Yang, Z.; He, X.; Gao, J.; Deng, L.; Smola, A.J. Stacked Attention Networks for Image Question Answering. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 21–29. [[CrossRef](#)]



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).