Two-Stream Convolution Augmented Transformer for Human Activity Recognition [*]

Bing Li[1], **Wei Cui**[2], **Wei Wang**[1,3], **Le Zhang**[2], **Zhenghua Chen**[2], **Min Wu**[2]
[1]School of Computer Science and Engineering, University of New South Wales, Australia
[2]Institute for Infocomm Research, Agency for Science, Technology and Research (ASTAR), Singapore
[3]Dongguan University of Technology, China
{bing.li, weiw}@unsw.edu.au, {cuiw, zhang_le, chen_zhenghua, wumin}@i2r.a-star.edu.sg

Abstract

Recognition of human activities is an important task due to its far-reaching applications such as healthcare system, context-aware applications, and security monitoring. Recently, WiFi based human activity recognition (HAR) is becoming ubiquitous due to its non-invasiveness. Existing WiFi-based HAR methods regard WiFi signals as a temporal sequence of channel state information (CSI), and employ deep sequential models (*e.g.*, RNN, LSTM) to automatically capture channel-over-time features. Although being remarkably effective, they suffer from two major drawbacks. Firstly, the granularity of a single temporal point is blindly elementary for representing meaningful CSI patterns. Secondly, the time-over-channel features are also important, and could be a natural data augmentation. To address the drawbacks, we propose a novel *Two-stream Convolution Augmented Human Activity Transformer* (THAT) model. Our model proposes to utilize a two-stream structure to capture both time-over-channel and channel-over-time features, and use the multi-scale convolution augmented transformer to capture range-based patterns. Extensive experiments on four real experiment datasets demonstrate that our model outperforms state-of-the-art models in terms of both effectiveness and efficiency.

Introduction

Human activity recognition (HAR) aims to recognize various human activities such as walking, sitting, falling down, etc. It is widely used in many fields such as healthcare system (Zheng et al. 2017; Jobanputra, Bavishi, and Doshi 2019), smart building solutions (Pu et al. 2013), and Internet of Things (IoT) applications. For examples, elderly people monitoring (Jobanputra, Bavishi, and Doshi 2019), energy-efficiency system for smart buildings (Pu et al. 2013), daily activity recognition for robot-assisted living (Zhu and Sheng 2011). Other applications include fall detection (Wang, Wu, and Ni 2017), security monitoring (Zheng et al. 2017), rescue systems (Grzonka et al. 2010), etc.

Due to its importance, numerous research efforts are devoted to HAR in recent years, mostly rely on various sensors or dedicated devices, such as cameras (Aggarwal and Ryoo 2011), wearable sensors (Ertin et al. 2011), and radars (Lien et al. 2016). However, these methods commonly require limited working conditions and special treatments on deployment and commissioning. E.g., cameras have limited viewing angles, illumination requirement, and needs line-of-sight (LOS) condition (no fog, smoke, or other obstacles). The reliance on dedicated devices and limited working conditions generally leads to high costs yet low efficacy, which hinders their applications from daily-life scenes.

Recently, the research community has sought using off-the-shelf devices to recognize human activities in a “device-free” manner, primarily based on the WiFi, an ubiquitously available device. WiFi-based solutions build upon the principle that human actions between WiFi transmitters and receivers (as shown in Fig. 1) will incur subtle yet unique variations (a.k.a. multi-path and the fading effect) on WiFi signals (Wang et al. 2015). There are two kinds of commonly used WiFi signal data: received signal strength (RSS) and channel state information (CSI). Due to being finer-grained and containing abundant environment information that would benefit HAR task, CSI becomes the de-facto standard. However, different from RSS where the patterns are simple and explicit, in CSI, the patterns are often implicit because they underneath the large amount of raw data (90 times more than RSS data), leading to a high noise-feature ratio. Thus, it is hard to obtain satisfactory results solely relying on extracting simple yet effective patterns.

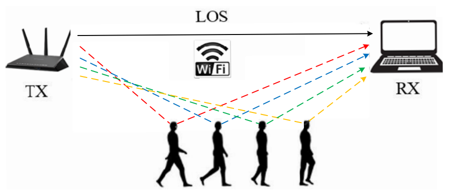

Figure 1: A human with different activities brings unique multi-path reflections and refractions in wireless signals from a transmitter (TX) to a receiver (RX).

Observing this, pioneer research efforts have sought for various deep learning (DL) techniques, such as CNN (Chowdhury 2018), RNN (Yousefi et al. 2017), and LSTM (Chen et al. 2018), to automatically extract informative features. Despite being effective, existing DL-based solutions share some common drawbacks, which hinder them from further performance improvement, as listed below:

[*]W. Cui and W. Wang are the co-corresponding authors.

i) *Order-unawareness.* Spatial models such as CNN simply treat CSI data as an image and use two-dimensional kernels to extract spatial patterns. However, different from images where semantics of both dimensions are identical (*i.e.*, refers to spatial positions), for CSI data, the meanings of the two dimensions are completely different (*i.e.*, refers to temporal time-stamp and channel state, respectively). Simply treating them equally will yield inferior features. Moreover, CNN does not preserve order information, so it is unable to effectively distinguishing order-sensitive activities such as sit-down and stand up.

ii) *Neglect the temporal features extracted along the channel dimension (time-over-channel).* Sequential models such as RNN and LSTM regard CSI data as a temporal sequence. They are aware of orders and able to capture channel features extracted along the temporal dimension (channel-over-time). Unfortunately, they neglect the time-over-channel features, which are shown to be effective in distinguishing actions with similar motions but different body postures, such as sit down and lie down, since the time-over-channel features are more sensitive to density changes as different channel frequency has different penetration on human bodies.

iii) *Granularity is too small.* Existing models take every point as the meaningful unit to generate features. The point-level granularity is too small to see the whole picture, since a meaningful pattern usually appears in a continuous *range* rather than a single point due to the continuity[1] of CSI data.

To address the above issues, we propose a sophisticated yet highly effective *Two-stream Convolution Augmented Human Activity Transformer* (THAT) model to exert the advances of deep learning techniques on HAR tasks. Our model presents a two-tower structure, each tower is in charge of the channel stream or the temporal stream to extract both time-over-channel and channel-over-time features. The core component of our model is the *Multi-scale Convolution Augmented Transformer* (MCAT), which adopts a residual-connected multi-head self-attention to effectively generate hidden representations and a multi-scale convolution block to capture range-based patterns. To make the model be order-sensitive, we propose a Gaussian range encoding to preserve the positional information of the CSI data. Besides, owing to the parallel nature of self-attention and convolution block, our model is also efficient enough to support real-time recognition. The experimental results show that the proposed model not only achieves a very high accuracy (above 98.7%), but also is very time-efficient (1.83$\sim$3.37$\times$ faster than state-of-the-art sequential models).

Related Work

CSI based human sensing methods mainly rely on data-driven approaches to learn the complex relationship between wireless signals and human action (Wang et al. 2019). Traditional schemes for CSI-based activity recognition firstly extract discriminative and representative features from multiple domains and then feed the extracted features into machine learning models to predict certain activities as a multi-class classification problem. Some of the popular techniques are support vector machines (SVMs) (Wang et al. 2016), hidden markov model (HMM) (Wang et al. 2015), and k-nearest neighbor algorithm (kNN) (Ali et al. 2015).

[1]Not only the temporal dimension, but the channel dimension is also continuous due to the continuity of channel frequency.

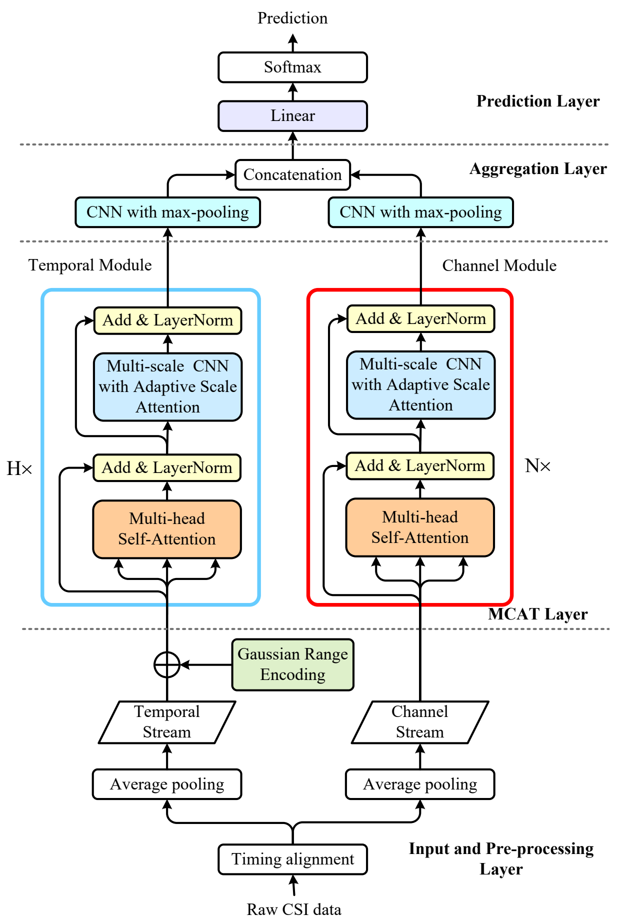

Figure 2: Architecture of THAT Model.

However, handcrafted solutions in previous works require export-knowledge and are labor-intensive and time-consuming. Recently, deep architectures contribute to the significant advancement in the fields of computer vision, natural language processing, data analysis and so on. Some researchers propose to use deep learning approaches, which are able to save that labor-intensive and time-consuming feature exploration procedure. Yousefi *et al.*proposed a deep learning approach, *i.e.*, LSTM, which could hold temporal state information of activities and help in extraction of implicit features for similar activities (Yousefi et al. 2017). Chen *et al.*argued that the conventional LSTM can only process the CSI measurements in the forward direction, which may lead to some informative features loss (Chen et al. 2018). Based on that, they proposed an attention based bidirectional long short-term memory (ABLSTM) approach for human activity recognition using WiFi CSI. In (Shi et al. 2019), discriminative features for different human activities were extracted by LSTM with RNN and then were inputted to a softmax classifier for activity recognition. Gao *et al.*developed a CSI-based sparse auto-encoder (SAE) net-

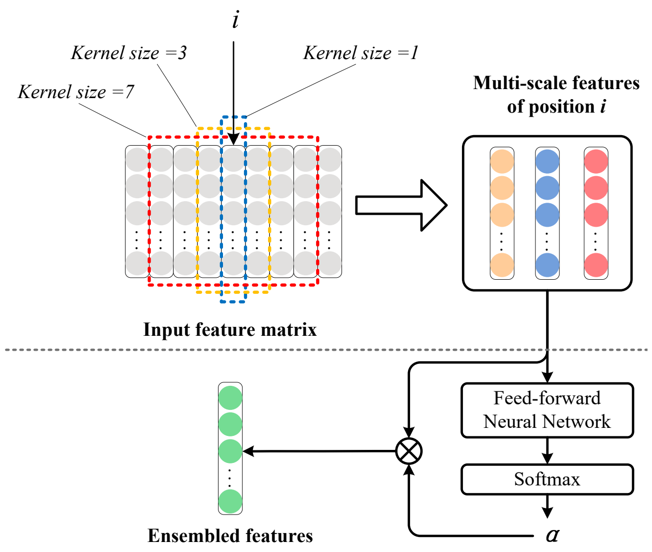

Figure 3: An example of generating ensembled features for position i via multi-scale convolutional blocks. The top part shows convolutions with kernel size 1, 3, and 7, and the bottom part shows an adaptive scale attention.

work for device free wireless localization and activity recognition (Gao et al. 2017). Ma *et al.*(Ma et al. 2018) proposed a deep learning framework based on convolutional neural network (CNN) for sign gesture recognition using CSI, which achieved higher accuracy and efficiency than the non-deep-learning baselines.

Model

Model Overview

The architecture of our THAT model is summarized in Figure 2. From bottom to top, the first layer inputs the raw CSI data and pre-processes it into temporal and channel streams. The MCAT layer extracts discriminative features from the two-stream. Finally, the two-stream features are aggregated and fed it into the prediction layer for the final output.

Input and Pre-processing Layer receives the raw CSI data as inputs and resizes it into temporal stream and channel stream to be fed as the input of MCAT layer. The raw CSI data is a set of records $\{r_1, r_2, ..., r_n\}$. Each record r_i is a two-dimensional matrix, in which a cell $v_{tc} \in r_i$ is a real value indicating the state value of channel c at time t. To facilitate batch-processing, we perform a timing alignment[2] to make the size of each record identical, *i.e.*, r_i has the same dimensionality as $T \times C$. To be less memory-intensive, we shrink the size of temporal dimension T by preforming a mean-pooling on adjacent time slots. The temporal dimension first (with dimensionality $T \times C$) data is used as temporal stream. The channel dimension first (with dimensionality $C \times T$) data is used as channel stream, which could be readily fetched by a simple transpose operation.

[2]We evenly split a time range into time slots and map each row into their slot according to their time-stamps.

MCAT Layer extracts discriminative features from the two stream data. This layer manifests a two-tower structure. Each tower is a stack of H-layer (for temporal stream, and N-layer for channel stream) multi-scale convolution augmented transformers. The two towers separately extract channel-over-time features and time-over-channel features. Notably, if $H = 0$ or $N = 0$, the model is degenerated to a single-stream model.

Aggregation Layer receives features of the temporal stream and channel stream and aggregates them into fixed-length vectors via two separate Convolutional blocks (CNN). Then, temporal and channel vectors are concatenated to be fed as the input of the prediction layer.

Prediction Layer is a linear layer with a softmax operation to compute the probabilities over activity categories.

Multi-scale Convolution Augmented Transformer (MCAT)

MCAT consists of two sequentially stacked sub-layers: (1) a multi-head self-attention, and (2) a multi-scale CNN with an adaptive scale. Each of the two sub-layers is encompassed by a residual connection (He et al. 2016) (Add) and a layer normalization (Ba, Kiros, and Hinton 2016) (LayerNorm).

Multi-head Self-attention Module We employ a multi-head self-attention mechanism (Vaswani et al. 2017) to generate hidden representations from inputs. Self-attention could directly integrate the information within the entire sequence; thus could fully overcome the long-range dependency problem of RNN and LSTM.

The self-attention outputs a weighted sum of the values, where weight assigned to each value is computed by the dot-product of the query with the corresponding key. Formally, it is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^\mathsf{T}}{\sqrt{d_k}})V, \quad (1)$$

where d_k is the queries and keys dimension, and $\sqrt{d_k}$ is a scaling factor that enables smoother gradients. $Q \in \mathbb{R}^{L\times d_k}$, $K \in \mathbb{R}^{L\times d_k}$, and $V \in \mathbb{R}^{L\times d_v}$ is queries, keys, and values, respectively. They are generated by letting the input $X \in \mathbb{R}^{L\times d_{in}}$ go through three linear projections, namely,

$$Q = X\mathcal{W}_Q,\ K = X\mathcal{W}_K,\ V = X\mathcal{W}_V \quad (2)$$

where $\mathcal{W}_Q \in \mathbb{R}^{d_{in}\times d_k}$, $\mathcal{W}_K \in \mathbb{R}^{d_{in}\times d_k}$, and $\mathcal{W}_V \in \mathbb{R}^{d_{in}\times d_v}$ are projection parameters.

In order to jointly attend to information from different representation subspaces, the queries, keys and values are independently projected h times (called h-head) with different projection parameters, and the outputs of projections are concatenated and projected again to result the final output:

$$\text{MultiHead}(Q, K, V) = [head_1; ...; head_h]\mathcal{W}_O, \quad (3)$$

where $head_i = \text{Attention}(X\mathcal{W}_Q^i, X\mathcal{W}_K^i, X\mathcal{W}_V^i)$, and $\mathcal{W}_O \in \mathbb{R}^{hd_v\times d_o}$ is the final projection matrix.

To facilitate residual connections, all sub-layers in the MCAT, as well as the input layer, share identical dimension, *i.e.*, $d_{in} = h * d_v = d_o$.

Multi-scale Convolutional Block with Adaptive Scale Attention For HAR tasks, a single temporal point is too small to be informative. To capture features in a lager scope, we propose multi-scale convolutional blocks that use different scale kernel sizes and adaptively adjust between different scale by an adaptive scale attention mechanism.

Given a set of kernel sizes $J = j_1, j_2, ..., j_{|J|}$, each kernel size is related to a specific scale, *e.g.*, a 30 kernel size enables the model to capture the pattern within a temporal range of 30 ms. The output of multiple convolutional blocks is $P = P_1, P_2, ..., P_{|J|}$, where $P_i \in \mathbb{R}^{L \times d_o}$ is the feature matrix regarding the i-th kernel size j_i. P_i is defined as:

$$P_i = \text{ReLU}(\text{Dropout}(\text{BN}(\text{Conv}(\mathcal{W}^{j_i}; X)))) \quad (4)$$

Eq. 4 consists of four cascaded operations: a convolution $(\text{Conv}(\cdot))$, a batch normalization $(\text{BN}(\cdot))$ (Ioffe and Szegedy 2015), a Dropout operation $(\text{Dropout}(\cdot))$ (Srivastava et al. 2014), and a ReLU unit. The core operation is $\text{Conv}(\mathcal{W}^{j_i}; X)$, which receives the output features X of the multi-head self-attention module along with learnable weights $\mathcal{W}^{j_i} \in \mathbb{R}^{d_o \times j_i \times d_o}$. The $\mathcal{W}^{j_i}$ consists of d_o filters, and each filter has the size of $j_i \times d_o$, convolving j_i adjacent positions. Notice that we use zero-padding to the two sides of X by $\lfloor (j_i - 1)/2 \rfloor$ to make the resulting feature map have the same size as $L \times d_o$. Fig. 3 shows an example of using scales $1, 3, 7$ to capture pattern within different ranges to yield three features vectors.

Notably, it is unwise to use all those features since some of them are either redundant or less informative. In order to adaptively ensemble features at different scales, we propose an adaptive scale attention mechanism, as shown in Fig. 3. Formally, the ensembled features P^{ens} are defined as:

$$P^{ens} = \boldsymbol{\alpha} P = \sum_i \alpha_i P_i, \quad (5)$$

where α_i is an attention score of P_i at scale j_i. $\boldsymbol{\alpha} =< \alpha_1, \alpha_2, ..., \alpha_{|J|} >$ is the vector of attention scores, which can be computed by:

$$\boldsymbol{\alpha} = \text{softmax}(\text{FFN}(P)), \quad (6)$$

where $\text{FFN}(\cdot)$ is a two-layer fully connected feed-forward neural network:

$$\text{FFN}(P_i) = \text{ReLU}(P_i \mathcal{W}_1^\mathsf{T} + b_1)\mathcal{W}_2^\mathsf{T} + b_2, \quad (7)$$

where $\mathcal{W}_1 \in d_h \times d_o$ and $\mathcal{W}_2 \in 1 \times d_h$ are parameter matrices with a hidden dimension d_h, and b_1 and b_2 are biases.

Preserve Order Information by Gaussian Range Encoding The order information is vitally important in identifying reverse actions such as sit down and stand up.Unfortunately, existing positional encodings methods, *e.g.*, absolute encodings (Vaswani et al. 2017) or relative encodings (Shaw, Uszkoreit, and Vaswani 2018) are all defined on a single point: they assign a unique and highly-discriminative encoding for each single point. As we argued, a meaningful unit should be a range rather than a single point. Simply encoding each point will turn out to be a kind of noise, and lead to generally worse results. Thus, we wish to use *range-based* encodings instead.

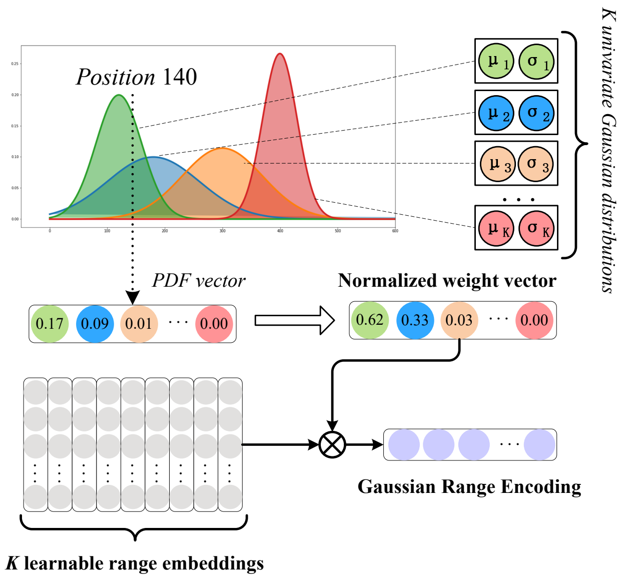

Figure 4: An example of generating Gaussian range encoding for one of positions. The final Gaussian range encoding is the multiplication of the K learnable range embeddings and normalized PDF vector of K Gaussian distributions.

To make the encoding be flexible to different start/end blank periods or subtle action speed changes, we propose a Gaussian range encoding that allows a position belonging to multiple ranges at the same time, and dynamically adjust proportions of different ranges during the training.

Without loss of generality, we assume there are K different ranges, and use random variable z_{ij} to denote the occurrence of position i belonging to the j-th range. We assume z_{ij} is drawn from a Gaussian distribution $\mathcal{N}(\mu^j, \sigma^j)$ (*i.e.*, $z_{ij} \sim \mathcal{N}(\mu^j, \sigma^j)$) with probability $p^j(i)$. As such, a position i is a mixture of all K ranges, and the distribution over ranges are $p_i =< \frac{p^1(i)}{\zeta}, \frac{p^2(i)}{\zeta}, ..., \frac{p^K(i)}{\zeta} >$ where ζ is a normalization factor. Given the values vector $v =< v_1, v_2, ..., v_K >$, v_j is the value of the j-th range. The expectation of position i is $p_i v^\mathsf{T}$.

All μ, σ, and v are unobserved hidden variables, and thus they cannot be directly estimated through maximum likelihood estimation (MLE). Fortunately, we can implicitly perform an expectation-maximization by neural networks' back-propagation. We set μ, σ, and v as randomly initialized free parameters, and dynamically update them with the training of the whole THAT model.

Formally, let $\boldsymbol{\beta} \in \mathbb{R}^{L \times K}$ be the normalized weights over K Gaussian distributions, which is defined as

$$\boldsymbol{\beta} = \text{softmax}(B), \quad (8)$$

where $B \in \mathbb{R}^{L \times K}$ is a weight matrix, in which each cell b_{ij} indicates the weight of the j-th Gaussian distributions for position i. b_{ij} is formally defined as:

$$b_{ij} = -\frac{(i - \mu^{(j)})^2}{2\sigma^{(j)2}} - log(\sigma^{(j)}), \quad (9)$$

where $\mu^{(j)}$ and $\sigma^{(j)}$ are learnable parameters that indicate the mean and standard deviation for j-th Gaussian distributions, respectively.

Finally, the range-biased stream X' is generated by adding the range encodings to the original stream X:

$$X' = X + \boldsymbol{\beta} E, \tag{10}$$

where $E \in \mathbb{R}^{K \times d_{in}}$ is a set of learnable range encodings. We can see that E corresponds to v. $\boldsymbol{\beta}$ corresponds to p_i, which can be proved that it is equivalent to computing L_1-normalized K univariate Gaussian probability density functions (PDFs).

Fig. 4 shows an illustrative examples of the range encoding. For position 140, we can get the normalized PDF vector for K different Gaussian distributions, and each element in the vector indicates the proportion the corresponding range. The final encoding is the multiplication of the range embeddings and their corresponding proportions.

Aggregation Layer

Assume that the features extracted from temporal and channel streams have the size of $T \times d_t$ and $C \times d_c$, respectively. In order to be amenable for the prediction layer, it is necessary to aggregate the features into a small fixed length vector. The aggregation layer aggregates each feature matrix using a CNN with a max-pooling operation:

$$\begin{aligned} u^X &= v(\mathcal{W}^X, X) \\ &= \mathrm{ReLU}(\mathrm{Dropout}(\mathrm{Pooling}(\mathrm{Conv}(\mathcal{W}^X; X)))), \end{aligned} \tag{11}$$

where $v(\cdot)$ consists of four cascaded operations: a convolution, a 1-max-over-time pooling operation (Kim 2014), a Dropout operation, and a ReLU unit. The convolutional layer has w kernel sizes and l filters, and each filter has the size of $h \times d$ (h denotes the kernel size). Further, we use a 1-max-pooling operation to select the largest value over the feature map of a particular kernel to capture the most important feature, and it also helps to shrink the feature size to the kernel number. The output vector u^X has a fixed size $1 \times wl$.

The final feature vector $U \in \mathbb{R}^{1 \times (w^t + w^c)}$ is generated by concatenating the two output vectors:

$$U = [u^T; u^C]. \tag{12}$$

The final feature vector U is fed to the prediction layer to identify different activities.

Loss Function

The loss function $\mathcal{L}$ is a standard cross-entropy loss (CE-loss), which is defined as:

$$\mathcal{L} = -\sum_{i}^{|A|} y_i log(f(\boldsymbol{\theta}; < X_T; X_C >)), \tag{13}$$

where y_i is the ground-truth label, and $f(\cdot)$ denotes the predicted distribution of the final prediction layer of THAT model. X_T and X_C are input temporal and channel streams. $|A|$ is the number of different activity categories.

Experimental evaluation

In this section, we evaluate the performance of the proposed THAT model on human activity recognition tasks, and demonstrate its superiority over state-of-the-art models.

Table 1: Statistics of the four evaluation datasets.

Datasets	# Rec.	# Activ.	# Channels.	Freq.	TX-RX Dist.
Office Room	140	7	30×3	1K Hz	3m
Activity Room	600	6	30×3	500 Hz	4.5m
Meeting Room	600	6	30×3	500 Hz	2m
Activity+Meeting	1200	6	30×3	500 Hz	4.5m/2m

Evaluation Datasets

We used four datasets for evaluation, *i.e.*, Office Room[3], Activity Room, Meeting Room, and Activity+Meeting. The first dataset is publicly available, and the last three are collected by our prototype system. Each dataset is a set of CSI matrices along with their corresponding ground-truth labels. Their brief statistics are summarized in Table 1.

Experimental Settings and Evaluation Metrics

The strides of average pooling were set to 4 for temporal stream and 3 for channel stream. The number of Gaussian distributions was $K = 10$. The μs were evenly distributed among temporal dimension, namely, beginning from 25 and ending with 475 with the step 50. All σs were set to 8. The number of stacks for temporal module was $H = 5$ and channel module was $N = 1$; The dimensionality $d_{in} = d_k = d_o$ was set to 90 and 500, the number of heads h were set to 9 and 200, and $d_v = d_o/h$. The dropout rate was 0.1. For temporal module and channel module, the kernel sizes were $\{1, 3, 5\}$ and $\{1, 2, 3\}$, the hidden dimensions d_h were 360 and 4000. For temporal and channel module, the kernel numbers w were set to 128 and 16, and the kernel sizes l were set to $\{10, 40\}$ and $\{2, 4\}$. The dropout rate for this layer was set to 0.5.

The model was implemented using Pytorch 1.4 with Python 3.6, and trained on a Nvidia 1080Ti GPU. For optimization, we used Adam (Kingma and Ba 2014) with an initial learning rate 0.001. All weight parameters were initialized using Xavier (Glorot and Bengio 2010). All datasets adopt 8:1:1 train/dev/test split. The batch size was 16. We ran the model for a maximum of 50 epochs and selected the best on validation set for testing.

Evaluation Metrics Following the common practice in human activity recognition efforts (Chen et al. 2018), we adopt recognition accuracy (*i.e.*, the proportion of correctly recognized activities among all predictions) as the metric.

Baselines

We compared with five state-of-the-art models, including two feature-based models S-RF (Yousefi et al. 2017) and S-HMM (Wang et al. 2017), and three DL-based models CNN, LSTM (Yousefi et al. 2017), and ABLSTM (Chen et al.

[3]https://github.com/ermongroup/Wifi_Activity_Recognition

2018). The detailed features extraction process of feature-based methods can be found in (Yousefi et al. 2017).

Table 2: The recognition accuracy (%) comparison of THAT and baselines on the four evaluation datasets.

Datasets	S-RF	S-HMM	LSTM	CNN	ABLSTM	THAT
Office Room	75.3	79.7	91.4	96.4	97.1	**98.2**
Activity Room	80.0	75.0	89.7	94.3	95.6	**98.4**
Meeting Room	84.7	83.4	90.6	96.2	96.8	**99.0**
Activity+Meeting	82.6	80.5	90.1	95.4	95.9	**98.6**

Main Results

Table 2 lists the recognition accuracy of the proposed THAT model compared with baselines on the four evaluation datasets. From Table 2, we can see that:

i) Our model THAT excels all baselines significantly, achieving new state-of-the-art results on these datasets. Our model outperforms the best previous model ABLSTM by 1.1 pts (percentage points), 2.8 pts, 2.2 pts, and 2.7 pts on the four datasets, respectively. This demonstrates our convolution augmented transformer and the two-stream structure do truly achieve a better performance on HAR tasks.

ii) The performance gaps between feature-based models (S-RF and S-HMM), and deep-neural models (LSTM, CNN, ABLSTM, and THAT) are huge. Among all the models, S-HMM has the worst performance (on average 79.65% accuracy), and S-RF is only slightly better than S-HMM (80.65%). For DL-baseed models, LSTM performs the worst since it suffers greatly from the long-range dependency problem. Being able to adaptively integrate hidden states by a attention mechanism, ABLSTM achieved a better results (around 6 pts) than traditional LSTM model and became the best among baselines. All DL-based models achieved at least 10 pts average improvements over feature-based models. Compared with the best feature-based model, our model achieved 18.6 pts, 18.4 pts, 14.3 pts, and 16 pts improvements. This is because deep-neural models are of high expressiveness, and can automatically and adaptively extract useful features from raw data.

iii) Our model performed equally well on different working scenarios. On the first three single-scenario datasets, the difference is only 0.6 pt. For hybrid-scenario dataset Activity+Meeting, the performance of THAT model is commensurately good compared to that under single-scenario and excels other models significantly. This demonstrates the effectiveness of THAT on hybrid-scenario. Another observation is that other DL-based models (*i.e.*, LSTM, CNN, and ABLSTM) perform slightly worse on Activity Room than other two. We believe it mainly caused by the TX-RX distance, which is larger than others in Activity Room (4.5m v.s. 2m and 3m). It may weaken the received strength and entail more noise. Thus, our model is of high robustness against different working scenarios.

Categorical Performance Comparison

Table 3 shows the categorical performance comparison between THAT and baselines on the four evaluation datasets. We can see that our model outperformed almost all baselines in identifying any activity on all the four datasets. This demonstrates our model is effective in capturing general patterns rather than just taking advantages of the "biases" of some specific activities.

Another observation is that, LSTM and CNN suffer more difficulties in recognizing reverse activities sit down and stand up because they need position or order information. Owing to involving an attention mechanism, ABLSTM has an improvement over LSTM. Our model performs significantly better than both scope-based model CNN (5.5 pts) which does not preserve positional information and sequential model ABLSTM (1.5 pts) that focuses on a fixed scale. This demonstrates our proposed Gaussian range encoding and multi-scale CNN block could preserve order information and are able to capture multi-scale features.

Ablation Test

we conducted an ablation study to evaluate the contributions of the proposed Gaussian range encoding, multi-scale CNN, and the two-stream structure. Table 4 represents the results. In each test, we ablated a specific component from the full model. Notably, we made two further testes by filling the ablated component with an alternative existing method: the first used positional encoding proposed in (Vaswani et al. 2017) to replace our Gaussian range encoding (*i.e.*, - Gaussian Range Encoding (+ PE)), the second used a position-wise feed-forward neural network to replace our multi-scale CNN (*i.e.*, - Multi-scale CNN (+ PFFN)).

From Table 4, we can see that position/order information does truly help a better recognition. The ablation on Gaussian range encoding incurs roughly 0.5 pt accuracy decline. Interestingly, the results of using positional encoding (PE) are even worse than without any positional encoding. This is caused by the over-positionality that the encodings assigned to each single position turn out to be an interference.

The ablation on multi-scale CNN incurs an average 1.83 pts accuracy decline, compared with using PFFN, still 1 pt performance gap. This demonstrates multi-scale CNN can capture multi-scale patterns, which is critical for model performances, especially on datasets (*e.g.*, Office Room) that have a long time range.

Another observation is, compared with single-stream model, the two-stream structure could effectively improve the model performance. This indicates the two-stream could be complementary to improve the recognition accuracy. Among the two streams, the ablation on temporal module incurs a bigger decline, which is in accord with common senses that temporal features are more intuitive and important. From ablation test, we can conclude that all these components contribute significantly to the model performance.

Empirical Efficiency

To validate the time-efficiency of our model, we show the empirical execution time in Table 5. We take Activity Room as an example, the efficiencies for others are similar.

Table 3: Categorical performance comparison on Office Room, Activity Room, Meeting Room, and Activity+Meeting datasets.

Environment	Methods	Lie down	Fall	Pick up	Run	Sit down	Stand up	Walk	Average
Office Room	S-RF	65.1	82.2	85.5	83.8	59.8	60.7	89.8	75.3
	S-HMM	62.9	86.7	89.2	94.2	73.4	59.9	92.0	79.8
	LSTM	95.0	92.8	98.1	96.9	82.4	82.2	93.4	91.6
	CNN	**97.4**	97.5	98.0	98.5	90.2	93.8	**99.0**	96.3
	ABLSTM	96.4	98.9	97.6	97.8	94.9	96.5	96.6	97.0
	THAT	96.4	**99.0**	**98.9**	**98.7**	**97.4**	**99.9**	98.4	**98.4**

Environment	Methods	Jump	Bow	Run	Sit down	Wave hand	Walk	Average
Activity Room	S-RF	66.1	71.2	88.3	78.2	87.1	88.8	80.0
	S-HMM	36.8	47.9	92.7	90.8	89.8	91.7	75.0
	LSTM	88.1	81.8	90.6	90.9	94.7	91.9	89.7
	CNN	94.1	90.4	93.8	96.5	97.2	93.8	94.3
	ABLSTM	93.4	93.9	95.5	96.6	96.5	97.9	95.6
	THAT	**98.8**	**98.7**	**96.7**	**98.7**	**98.7**	**98.9**	**98.4**
Meeting Room	S-RF	84.6	88.7	81.1	82.1	81.4	90.5	84.7
	S-HMM	67.1	77.2	89.1	82.8	90.6	93.6	83.4
	LSTM	89.6	87.2	90.5	91.6	93.7	90.7	90.6
	CNN	96.2	92.2	95.8	98.4	98.3	96.2	96.2
	ABLSTM	96.5	98.2	97.1	98.5	92.8	97.6	96.8
	THAT	**99.3**	**99.8**	**97.3**	**99.7**	**99.2**	**98.4**	**99.0**
Activity+Meetinm	S-RF	70.9	82.8	88.7	85.6	81.7	85.9	82.6
	S-HMM	50.9	66.8	88.7	88.8	89.7	97.8	80.5
	LSTM	78.8	91.9	96.9	92.1	89.2	91.7	90.1
	CNN	93.1	95.2	97.3	95.9	97.1	93.6	95.4
	ABLSTM	93.6	94.5	97.1	97.4	95.8	97.1	95.9
	THAT	**99.0**	**99.2**	**97.4**	**99.6**	**97.6**	**98.7**	**98.6**

Table 4: Ablation study results compared with the full THAT model.

Model	Office Room		Activity Room		Meeting Room		Activity+Meeting	
	Accuracy (%)	Δ	Accuracy (%)	Δ	Accuracy (%)	Δ	Accuracy (%)	Δ
THAT	**98.2**	-	**98.4**	-	**99.0**	-	**98.6**	-
- Gaussian Range Encoding	97.9	-0.4	97.9	-0.5	98.5	-0.5	98.2	-0.4
- Gaussian Range Encoding (+ PE (Vaswani et al. 2017))	91.1	-7.2	84.3	-14.1	90.3	-8.7	87.8	-10.8
- Multi-scale CNN	95.3	-3.0	97.4	-1.0	97.0	-2.0	97.3	-1.3
- Multi-scale CNN (+ PFFN)	97.2	-1.1	97.2	-1.2	98.1	-0.9	97.7	-0.9
- Temporal Module	92.0	-6.3	95.2	-3.2	97.5	-1.5	95.9	-2.7
- Channel Module	93.8	-4.5	97.7	-0.7	98.3	-0.7	98.0	-0.6

Table 5: Empirical execution time of all models on Activity Room dataset.

Models	Training(Sec)	Testing(Sec)	Throughput(Recs/Sec)
S-RF	6.09	0.016	31250
S-HMM	0.029	0.22	2272.72
LSTM	5168.86	4.39	113.9
CNN	1474.76	1.12	446.43
ABLSTM	11352.82	6.77	73.86
THAT	2996.75	1.55	332.58

From Table 5, we can see that all DL-based models (*i.e.*, LSTM, CNN, ABLSTM, and THAT) are more time-consuming than traditional machine learning models (*i.e.*, S-RF, S-HMM). Our THAT model (and CNN) has a better time-efficiency than sequential models LSTM and ABLSTM, since most of the computations (*e.g.*, multi-head self-attention module, all convolution modules) can be computed in parallel. The high time costs in training phase would not be a big issue since it can be computed offline. In testing phase, our model is 3.37× faster than ABLSTM, 1.83× faster than LSTM, and competitive to CNN. The throughput rate of THAT is 332.58 Recs/Sec, indicating each record can be inferred within 4ms. Considering the window size of each record is 4 seconds, the time cost of our model could be fully ignored. This demonstrates our model is not only of high effectiveness, but also efficient enough for real-time WiFi-based human activity recognition.

Conclusion

In this paper, we propose a novel network structure for device-free HAR tasks. Our model uses a two-stream layout to extract both time-over-channel and channel-over-time features, and uses Gaussian range encoding and a multi-scale convolution block to capture range-based patterns. The experimental results show that, compared with the best previous model ABLSTM, our model delivers an average 2.2 pts accuracy improvement, while being 1.83∼3.37× faster.

References

Aggarwal, J. K.; and Ryoo, M. S. 2011. Human activity analysis: A review. *ACM Computing Surveys (CSUR)* 43(3): 16.

Ali, K.; Liu, A. X.; Wang, W.; and Shahzad, M. 2015. Keystroke recognition using wifi signals. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, 90–102.

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *NIPS* .

Chen, Z.; Zhang, L.; Jiang, C.; Cao, Z.; and Cui, W. 2018. WiFi CSI based passive human activity recognition using attention based BLSTM. *IEEE Transactions on Mobile Computing* 18(11): 2714–2724.

Chowdhury, T. Z. 2018. *Using Wi-Fi channel state information (CSI) for human activity recognition and fall detection*. Ph.D. thesis, University of British Columbia.

Ertin, E.; Stohs, N.; Kumar, S.; Raij, A.; Al'Absi, M.; and Shah, S. 2011. AutoSense: unobtrusively wearable sensor suite for inferring the onset, causality, and consequences of stress in the field. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, 274–287. ACM.

Gao, Q.; Wang, J.; Ma, X.; Feng, X.; and Wang, H. 2017. CSI-based device-free wireless localization and activity recognition using radio image features. *IEEE Transactions on Vehicular Technology* 66(11): 10346–10356.

Glorot, X.; and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 249–256.

Grzonka, S.; Dijoux, F.; Karwath, A.; and Burgard, W. 2010. Mapping indoor environments based on human activity. In *2010 IEEE International Conference on Robotics and Automation*, 476–481. IEEE.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Ioffe, S.; and Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *International Conference on Machine Learning*, 448–456.

Jobanputra, C.; Bavishi, J.; and Doshi, N. 2019. Human activity recognition: a survey. *Procedia Computer Science* 155: 698–703.

Kim, Y. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* .

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Lien, J.; Gillian, N.; Karagozler, M. E.; Amihood, P.; Schwesig, C.; Olson, E.; Raja, H.; and Poupyrev, I. 2016. Soli: Ubiquitous gesture sensing with millimeter wave radar. *ACM Transactions on Graphics (TOG)* 35(4): 142.

Ma, Y.; Zhou, G.; Wang, S.; Zhao, H.; and Jung, W. 2018. Signfi: Sign language recognition using wifi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2(1): 1–21.

Pu, Q.; Gupta, S.; Gollakota, S.; and Patel, S. 2013. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th annual international conference on Mobile computing & networking*, 27–38. ACM.

Shaw, P.; Uszkoreit, J.; and Vaswani, A. 2018. Self-Attention with Relative Position Representations. In *NAACL-HLT*.

Shi, Z.; Zhang, J. A.; Xu, R.; and Cheng, Q. 2019. Deep learning networks for human activity recognition with csi correlation feature extraction. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, 1–6. IEEE.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15(1): 1929–1958.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wang, H.; Zhang, D.; Wang, Y.; Ma, J.; Wang, Y.; and Li, S. 2016. RT-Fall: A real-time and contactless fall detection system with commodity WiFi devices. *IEEE Transactions on Mobile Computing* 16(2): 511–526.

Wang, W.; Liu, A. X.; Shahzad, M.; Ling, K.; and Lu, S. 2015. Understanding and modeling of wifi signal based human activity recognition. In *Proceedings of the 21st annual international conference on mobile computing and networking*, 65–76.

Wang, W.; Liu, A. X.; Shahzad, M.; Ling, K.; and Lu, S. 2017. Device-free human activity recognition using commercial WiFi devices. *IEEE Journal on Selected Areas in Communications* 35(5): 1118–1131.

Wang, Y.; Wu, K.; and Ni, L. M. 2017. Wifall: Device-free fall detection by wireless networks. *IEEE Transactions on Mobile Computing* 16(2): 581–594.

Wang, Z.; Jiang, K.; Hou, Y.; Dou, W.; Zhang, C.; Huang, Z.; and Guo, Y. 2019. A Survey on Human Behavior Recognition Using Channel State Information. *IEEE Access* 7: 155986–156024.

Yousefi, S.; Narui, H.; Dayal, S.; Ermon, S.; and Valaee, S. 2017. A survey on behavior recognition using wifi channel state information. *IEEE Communications Magazine* 55(10): 98–104.

Zheng, X.; Wang, J.; Shangguan, L.; Zhou, Z.; and Liu, Y. 2017. Design and implementation of a CSI-based ubiquitous smoking detection system. *IEEE/ACM Transactions on Networking* .

Zhu, C.; and Sheng, W. 2011. Wearable sensor-based hand gesture and daily activity recognition for robot-assisted living. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 41(3): 569–573.