Proceedings of SEML 2025 Workshop: Machine Learning Theory and Applications
DOI: 10.54254/2755-2721/146/2025.TJ21927

A Time Series Data Classification Method for Gesture Recognition Based on LSTM and Attention Mechanism

Xinyu Mao

Institute of Computer Science and Engineering, University of Electronic Science and Technology of China, Xiyuan Ave, Chengdu, China
xinyumao311@gmail.com

Abstract: This paper addresses the issues of weak model transferability and poor environmental adaptability in cross-domain gesture recognition within wireless sensing technology. A time-series data classification model integrating Long Short-Term Memory (LSTM) and attention mechanism (W-LSTM+A) is proposed. By introducing a feature selection weight matrix to reconstruct the LSTM gating mechanism and combining a dynamic attention allocation strategy, the model's ability to capture key spatiotemporal features in channel state information is significantly enhanced. Experiments based on a WiFi signal dataset collected in a real office environment compared the performance of CNN, LSTM, and LSTM+A models. The results show that the LSTM+A model achieved a test accuracy of 87.3% after 200 training epochs, significantly outperforming CNN's 81.9%. Although the LSTM model had a higher final accuracy, its convergence speed was significantly slower than that of the LSTM+A model. Further analysis indicates that the attention mechanism, by strengthening key time-step features, enables the model to quickly capture effective patterns in the early stages of training. However, due to limited sample size, its potential has not been fully realized. This study provides new solutions for the cross-scene adaptability of wireless sensing technology and has application value in smart homes, health monitoring, and other fields.

Keywords: Wireless Sensing, LSTM, Attention Mechanism, Time-Series Data Analysis

1. Introduction

Intelligent wireless sensing systems leverage radio frequency signal propagation dynamics to achieve contactless environment-object interaction monitoring. By analyzing multipath distortion patterns and Doppler shift variations in electromagnetic radiation [1], these systems can infer spatial coordinates, motion trajectories, and biomechanical activity signatures of targets without physical sensors [2].The proliferation of IEEE 802.11-compliant devices has positioned WiFi channel state information (CSI) as a pivotal enabler for passive sensing architectures. CSI metrics, including amplitude attenuation, phase deviation, and multipath interference coefficients, encode sub-millimeter-scale displacement features and kinematic patterns correlated with target behaviors. This technological synergy drives device-free gesture recognition and continuous vital sign monitoring, particularly in energy-efficient smart spaces and tele-rehabilitation systems [3] where privacy-preserving sensing is paramount.

DOI: 10.54254/2755-2721/146/2025.TJ21927

Cross-domain recognition refers to the ability of a model trained in one scenario to be transferred to other scenarios while maintaining a high recognition accuracy rate, which is a branch of transfer learning. Factors such as target position, direction, and environmental changes affect perception accuracy [4]. Although there has been progress in cross-domain gesture recognition based on WiFi signals, existing technologies still face issues such as insufficient quality of training datasets, poor model transferability, and poor environmental adaptability, which urgently need improvement.

In temporal sequence analysis, conventional sensing frameworks frequently exhibit limitations in modeling inter-frame correlations of time-dependent signals. The gating architecture inherent in Long Short-Term Memory (LSTM) networks demonstrates superior capability in establishing temporal correlations across extended sequences, particularly suited for decoding non-stationary signal patterns in RF time-series analysis [5](@ref). Within wireless perception systems, this neural mechanism facilitates the extraction of Doppler-temporal features from channel state variations, significantly enhancing spatial resolution and gesture characterization accuracy, thereby emerging as a cornerstone in modern device-free sensing paradigms [6].

In recent years, research on cross-domain recognition has made a series of progress. Literature [7] proposed an adversarial training method based on domain adaptation, achieving an 87% cross-scene recognition accuracy rate in indoor positioning tasks by minimizing the feature distribution differences between the source and target domains. Furthermore, literature [8] introduced a meta-learning framework, using the MAML algorithm to optimize model initialization parameters, enabling WiFi gesture recognition models to maintain an 83.2% accuracy rate even with limited target-domain samples. To address the issue of dynamic environmental changes [9], developed a cross-domain adaptive model based on residual networks, preserving signal spatiotemporal features through skip connections, and reducing the misrecognition rate to 6.7% in device-heterogeneous scenarios. However, existing methods still face two major bottlenecks: one is sensitivity to hardware differences of transmission and reception devices, making feature alignment difficult; the other is reliance on partially labeled data in the target domain, limiting the application in purely unsupervised scenarios.

This paper attempts to classify time-series data from a wireless sensor network deployed in a real office environment. The task aims to serve as a real benchmark in the field of environment-assisted living. This paper focuses on the recognition of two gestures, which is a binary classification task consisting of predictions on time-series data of user movements in a real office environment. By combining deep learning models, the cross-domain gesture recognition problem is studied. Through feature analysis of CSI data in WiFi signals, algorithms such as LSTM, combined with attention mechanisms, are used to improve the robustness and recognition accuracy of wireless perception models under cross-scenario conditions, and to address issues such as feature extraction, dataset quality, and model transferability, providing new ideas for the development of wireless sensing technology.

2. Basic Principles

2.1. Introduction to LSTM (Long Short-Term Memory)

Long Short-Term Memory (LSTM) networks, initially conceptualized by Hochreiter and Schmidhuber in 1997, represent a paradigm shift in sequential data processing through enhanced recurrent architectures [10]. These networks fundamentally resolve the exponential gradient decay issues prevalent in conventional Recurrent Neural Networks (RNNs) during backpropagation through time (BPTT), particularly when modeling extended temporal dependencies.

While standard RNNs demonstrate intrinsic suitability for sequential pattern analysis, their practical implementation suffers from two critical constraints [11]: progressive information loss in

DOI: 10.54254/2755-2721/146/2025.TJ21927

temporal context propagation, and numerical instability during parameter updates. LSTM's architectural innovation lies in its differentiated control units,a system of adaptive information filters that dynamically regulate temporal feature retention.

The core of LSTM is its internal structure—the gating mechanism [10]. Compared to traditional RNNs, LSTM uses different gates at each time step to control the flow of information. The structure of LSTM can be divided into four main parts:

Forget Gate: Determines which information needs to be forgotten. The forget gate reads the current input and the hidden state from the previous time step, outputting a value between 0 and 1, indicating whether each unit's memory should be forgotten. For example, a value of 0 means complete forgetting, while a value of 1 means complete retention.

$$\mathrm{f_t} = \sigma(\mathrm{W_f} \cdot [h_{t-1}, \mathrm{x_t}] + \mathrm{b_f}) \tag{1}$$

where $\mathrm{f_t}$ is the output of the forget gate, σ is the sigmoid activation function, $\mathrm{W_f}$ and $\mathrm{b_f}$ are the weights and biases, h_{t-1} is the hidden state from the previous time step, and $\mathrm{x_t}$ is the input at the current time step.

Input Gate: Determines which new information needs to be stored in the memory unit. The input gate consists of two parts:

Selection of input signals: Decides which parts of the input should be updated.

Candidate memory unit: Calculates the candidate memory values to be updated.

$$\mathrm{i_t} = \sigma(\mathrm{W_i} \cdot [h_{t-1}, \mathrm{x_t}] + \mathrm{b_i}) \tag{2}$$

$$\tilde{\mathrm{C}}_\mathrm{t} = \tanh(\mathrm{W_C} \cdot [h_{t-1}, \mathrm{x_t}] + \mathrm{b_C}) \tag{3}$$

where $\mathrm{i_t}$ is the output of the input gate, $\tilde{\mathrm{C}}_\mathrm{t}$ is the output of the candidate memory unit, and $\tanh$ is the tanh activation function.

Cell State: The core of LSTM, where long-term memory is stored. By combining the forget gate and input gate, it decides which information to retain and which to forget. The update of the cell state is as follows:

$$\mathrm{C_t} = \mathrm{f_t} * \mathrm{C_{t-1}} + \mathrm{i_t} * \tilde{\mathrm{C}}_\mathrm{t} \tag{4}$$

where $\mathrm{C_t}$ is the cell state at the current time step, and $\mathrm{C_{t-1}}$ is the cell state from the previous time step.

Output Gate: Determines how the hidden state (output) at the current time step is calculated. The output gate decides the output of the hidden state based on the current cell state and input:

$$\mathrm{o_t} = \sigma(\mathrm{W_o} \cdot [h_{t-1}, \mathrm{x_t}] + \mathrm{b_o}) \tag{5}$$

$$h_\mathrm{t} = \mathrm{o_t} * \tanh(\mathrm{C_t}) \tag{6}$$

where $\mathrm{o_t}$ is the output of the output gate, and h_t is the hidden state at the current time step.

Compared to traditional RNNs, LSTM effectively solves the vanishing gradient problem, enabling it to capture dependencies over longer time ranges.

2.2. Attention Mechanism

The attention mechanism originates from biomimetic studies of human visual cognition [12] — humans selectively focus on key regions when perceiving complex scenes to optimize information processing efficiency. In 2014, Bahdanau et al. first introduced the attention mechanism into machine translation tasks [13] , and since then, it has gradually become a core component for deep learning models processing sequential data. In wireless sensing time-series data analysis, due to the influence of environmental noise and signal multipath effects, features at different time steps often have differential contributions [14]. The equal-weight memory mechanism of traditional LSTM struggles

DOI: 10.54254/2755-2721/146/2025.TJ21927

to effectively focus on key temporal features, while the attention mechanism, through dynamic weight allocation, enables the model to adaptively strengthen important moment signal features and suppress irrelevant noise interference [15].

Let the input sequence be $X = \{x_1, x_2, \dots, x_n\} \in \mathbb{R}^{n \times d}$ (where d is the feature dimension), and the model generates query (Query), key (Key), and value (Value) vectors through triple mapping:

$$\begin{aligned} Q &= XW_Q, \quad W_Q \in \mathbb{R}^{d \times d_k} \\ K &= XW_K, \quad W_K \in \mathbb{R}^{d \times d_k} \\ V &= XW_V, \quad W_V \in \mathbb{R}^{d \times d_v} \end{aligned} \tag{7}$$

The attention weight matrix is calculated through scaled dot-product:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{8}$$

where the scaling factor $\sqrt{d_k}$ prevents large dot-product values from causing gradient vanishing.

Let the input sequence be $X = \{x_1, x_2, \dots, x_n\}$, and the target sequence be $Y = \{y_1, y_2, \dots, y_m\}$. The purpose of the attention mechanism is to dynamically generate weighted representations by calculating the contribution of each element in the input sequence to the target elements.

3. Our Work: W-LSTM+A Model

This paper proposes a time-series data classification method based on deep learning models, aiming to improve accuracy and robustness in time-series data analysis. By combining Long Short-Term Memory (LSTM) with a custom attention mechanism and comparing it with Convolutional Neural Networks (CNN) and simple LSTM models, this paper designs an efficient time-series data classification framework, focusing on solving the efficiency and robustness issues in cross-scenario time-series data classification.

In the fields of wireless sensing and intelligent monitoring, time-series data often has complex spatiotemporal characteristics, which traditional single models struggle to capture. Therefore, this paper introduces an attention mechanism to enhance the model's ability to capture key time-step features. Specifically, the output h_t of LSTM will be passed as input to the attention layer. LSTM, through its unique gating mechanism, can effectively capture long-term dependencies in time-series data, while the attention mechanism further strengthens the feature extraction ability for important time steps.

To address the issue of traditional LSTM treating all CSI features equally in wireless sensing, this work designs a feature-weighted LSTM structure, introducing a feature selection weight matrix $\Phi \in \mathbb{R}^{d \times d}$ into its gating mechanism. The main computational process is as follows:

Dynamic Gating Calculation: Calculate four gates based on weighted features

$$G_t^f = \sigma(W_f \cdot [H_{t-1}, Z_t] + b_f)(\text{Forget Gate}) \tag{9}$$

The forget gate determines how much information from the previous time step the LSTM unit should retain. It calculates through matrix W_f, input $[H_{t-1}, Z_t]$ (concatenation of the previous hidden state and the current weighted input), and bias term b_f, and generates a value between 0 and 1 through the sigmoid function σ. A value closer to 1 indicates more retained information, while a value closer to 0 indicates more forgotten information. By dynamically adjusting the proportion of retaining information from the previous moment, the forget gate helps the network "forget" irrelevant or redundant information, thus focusing more on current useful information.

$$G_t^i = \sigma(W_i \cdot [H_{t-1}, Z_t] + b_i)(\text{Input Gate}) \tag{10}$$

DOI: 10.54254/2755-2721/146/2025.TJ21927

The input gate determines how much information from the current time step should be stored in the LSTM memory unit. Similar to the forget gate, the input gate calculates in a similar way but specifically controls the injection of new information. The input gate works in conjunction with the forget gate to decide which information to discard, which to retain, and how to introduce new information. This allows the network to flexibly balance information retention and updating, adapting to different input patterns and time-series characteristics.

$$G_t^o = \sigma(W_o \cdot [H_{t-1}, Z_t] + b_o)(\text{Output Gate}) \tag{11}$$

The output gate determines how much information from the current LSTM memory unit should be passed to the next time step. It controls access to the memory unit through the value generated by the sigmoid function σ. The output gate works with the candidate memory to decide how the current memory unit's state is transmitted to the next layer or time step. By adjusting the weights of the output gate, the network can control the propagation of information, preventing information overload or excessive consumption of computational resources.

$$\tilde{C}_t = \tanh(W_c \cdot [H_{t-1}, Z_t] + b_c) \quad (\text{Candidate Memory}) \tag{12}$$

The candidate memory generates potential new memories that will be partially or fully stored in the LSTM memory unit. Through the tanh activation function, the generated values range between -1 and 1. The candidate memory unit provides candidate content for new memories. In subsequent memory unit updates, these candidate contents will work with the input gate and forget gate to update the memory unit's state. This helps the network capture new features and patterns while maintaining associations with historical information.

Memory Unit Update:

$$C_t = G_t^f \circ C_{t-1} + G_t^i \circ \tilde{C}_t \tag{13}$$

The memory unit update combines the memory state from the previous moment with the current candidate memory to generate a new memory unit state. Through the Hadamard product of the forget gate and input gate, the contributions of the previous memory state and the current candidate memory are dynamically adjusted. The memory unit is the core component of the LSTM model, responsible for storing and transmitting long-term information. By dynamically updating the memory unit, the network can better adapt to long-term dependencies in time-series data. The memory unit update formula integrates information from the forget gate and input gate, allowing the network to retain historical information while introducing new important information.

Hidden State Output:

$$h_t = G_t^o \circ \tanh(C_t) \tag{14}$$

The hidden state output determines the hidden state at the current time step, which will be passed to the next time step or used as output. The output gate and tanh function work together to generate the hidden state. The hidden state is the output of the LSTM model at the current time step, carrying the model's understanding and memory of the input sequence. By controlling the output gate, the hidden state can avoid information overload or excessive consumption of computational resources during transmission. At the same time, the hidden state provides a foundation for subsequent prediction and classification tasks. By introducing the feature selection weight matrix and redesigning the gating mechanism, the feature-weighted LSTM structure can more effectively capture key features in wireless sensing, improving the model's accuracy and robustness. These formulas together form the basic framework of the feature-weighted LSTM model, providing a powerful tool for processing complex time-series data.

DOI: 10.54254/2755-2721/146/2025.TJ21927

The specific calculation process of the attention layer is as follows: The output sequence h_t of LSTM is weighted and summed through the attention mechanism to generate the final attention output. The formula can be written as:

$$\text{Attention Output} = \sum_{t=1}^{n} \alpha_t \cdot h_t \tag{15}$$

where α_t is the attention weight, representing the importance of each time step t. The attention weights are calculated through the following formula:

$$\alpha_t = \frac{\exp\left(u^T \tanh(W_a h_t + b_a)\right)}{\sum_{i=1}^{n} \exp(u^T \tanh(W_a h_i + b_a))} \tag{16}$$

where W_a and b_a are learnable parameters, and u is the attention vector.

The final attention output is passed through a fully connected layer (such as a Dense layer) for binary or multi-classification tasks. The formula is:

$$\hat{y} = \sigma\left(W_{output} \cdot \text{Attention Output} + b_{output}\right) \tag{17}$$

where W_{output} and b_{output} are the weights and biases of the fully connected layer, σ is the activation function (such as softmax or sigmoid), and $\hat{y}$ is the model's predicted output.

By combining LSTM and the attention mechanism, our model can more effectively capture long-term dependencies and key time-step features in time-series data. Compared to traditional CNN models and simple LSTM models, our method has significant advantages in the following aspects: enhanced robustness, improved classification accuracy, and cross-scene adaptability.

These methods can be widely applied to smart homes, health monitoring, and human-computer interaction, especially having important practical significance in wireless sensing and intelligent monitoring. Next, this paper will introduce in detail the experimental design and result analysis of this study, further verifying the effectiveness and superiority of the proposed method.

4. Results and Analysis

The length of the time-series data ranges from 19 to 103, with an average length of approximately 37.4. Each line of data is a four-dimensional RSS vector composed of (Sensor 1, Sensor 2, Sensor 3, Sensor 4). The dataset includes 166 samples in the training set and 159 samples in the test set.

The experimental results of this study show that the model combining LSTM and a custom attention mechanism outperforms traditional Convolutional Neural Networks (CNN) and basic LSTM models in time-series data classification tasks. The following are the specific analysis results:

This experiment trained and tested the dataset from a real office environment using CNN models, LSTM models, and LSTM models with a custom attention mechanism. The hyperparameter settings for each model are as follows:

CNN Model: Includes one convolutional layer with 64 3x3 convolutional kernels, using the ReLU activation function and L2 regularization (coefficient 0.01). The pooling layer uses a 2x2 pooling window. The output layer of the fully connected layer has one neuron, with the sigmoid activation function. The optimizer is Adam with a learning rate of 0.0001, the loss function is binary_crossentropy, the batch size is 128, and the number of training epochs is 200.

LSTM Model: Includes one LSTM layer with 128 hidden units, with an input shape of (seq_len, 4). The optimizer is Adam with a learning rate of 0.0001, the loss function is binary_crossentropy, the batch size is 128, and the number of training epochs is 200.

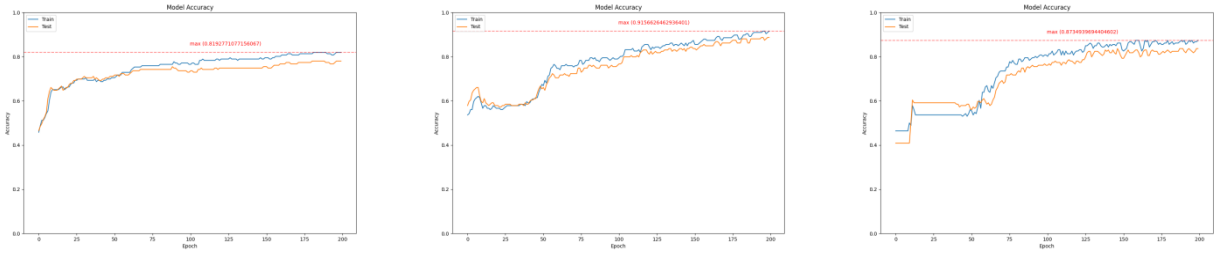
LSTM+A Model: Adds a custom attention layer on the basis of LSTM. The LSTM layer has 128 hidden units, and the attention layer includes learnable weight matrix W and bias b. The optimizer is Adam with a learning rate of 0.0001, the loss function is binary_crossentropy, the batch size is 128, and the number of training epochs is 200.

DOI: 10.54254/2755-2721/146/2025.TJ21927

4.1. Training and Testing Accuracy Plots

This experiment trained and tested the dataset from a real office environment using CNN models, LSTM models, and LSTM models with a custom attention mechanism.

The experimental results show that the LSTM model has higher accuracy on both the training and test sets than the CNN model. When trained to 200 epochs, the LSTM with attention mechanism (LSTM+A) has slightly lower accuracy on both the training and test sets than the basic LSTM model. However, it converges faster in the first 100 epochs and reaches an accuracy of over 80% earlier.



CNN LSTM LSTM+A

Figure 1: Accuracy Comparison

According to Figure a (CNN), the CNN model has a slow convergence speed during training, with the training loss stabilizing around the 150th epoch. The accuracy reached 0.819.

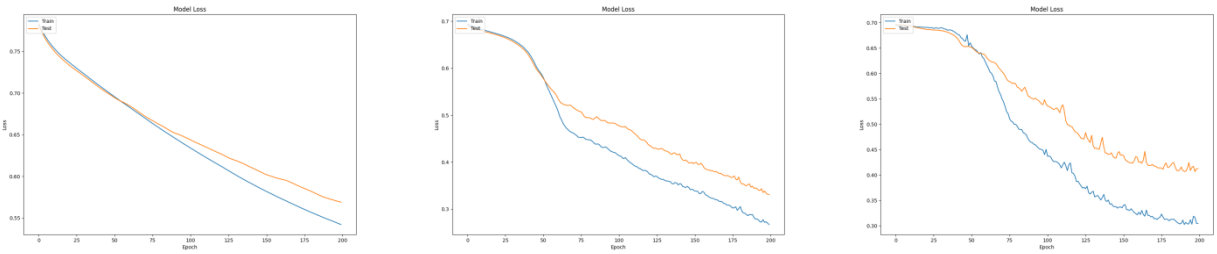
According to Figure b (LSTM), due to its ability to process sequential data, the LSTM model has a slower increase in accuracy in the early stages of training. However, as training progresses, the model gradually learns the temporal dependencies in the data. The training loss begins to stabilize at the 180th epoch, with an accuracy of 0.916.

According to Figure c (LSTM+A), after introducing the attention mechanism, the model can focus more on important parts of the sequence. In the early stages, similar to LSTM, the accuracy increases slowly at the beginning of training. In the later stages, the accuracy rises rapidly, and the training process is relatively stable. The loss converges around the 125th epoch. The attention mechanism allows the model to adaptively allocate weights during training, improving the model's convergence efficiency. The accuracy reached 0.873, which is better than the CNN model but slightly weaker than the LSTM model. It is preliminarily judged that the training data is insufficient, and the increased number of parameters due to the introduction of the attention mechanism has not fully realized the model's potential.

4.2. Training and Testing Loss Plots

From the training and testing loss plots, it can be seen that the LSTM+A model has a faster decrease in loss during training, but the increase in test loss is larger, indicating a lack of generalization ability. It is preliminarily judged that this may be due to the small number of training samples.

DOI: 10.54254/2755-2721/146/2025.TJ21927



CNN LSTM LSTM+A

Figure 2: Test Loss Change Plot

According to Figure a (CNN), the CNN model has a slow convergence speed during training, with the loss still being relatively large after 200 epochs of training, finally stabilizing around 0.55.

According to Figure b (LSTM), due to its ability to process sequential data, the LSTM model has a slower decrease in loss in the early stages of training. However, as training progresses, the model gradually learns the temporal dependencies in the data. The training loss begins to decrease more rapidly at the 30th epoch, and by the 60th epoch, the training loss tends to decrease linearly. By the end of 200 epochs, the final loss drops below 0.3.

According to Figure c (LSTM+A), after introducing the attention mechanism, the model can focus more on important parts of the sequence. In the early stages, the loss decreases slowly at the beginning of training, but as training progresses, the model's loss converges around the 150th epoch. The attention mechanism allows the model to adaptively allocate weights during training, improving the model's convergence efficiency. However, after converging, the loss fluctuates near the minimum value, and dynamically adjusting the learning rate may solve this problem. The loss also drops below 0.3 eventually.

5. Conclusion

This paper proposes a time-series data classification model based on LSTM and attention mechanism, which is innovative and practical in the field of wireless sensing. By comparing it with traditional CNN models and basic LSTM models, the significant advantages of this model in processing cross-scenario time-series data are verified. Experimental results show that the LSTM+A model outperforms the CNN model in terms of accuracy, loss, and has strong generalization ability. However, more data samples are needed for training to fully realize the model's potential. The excellent performance of the LSTM+A model is mainly attributed to its unique structural design. First, the LSTM layer can effectively capture long-term dependencies in time-series data, providing the model with strong sequence modeling capabilities. Second, the introduction of the attention mechanism enables the model to automatically focus on key time-step information, further enhancing the ability to capture important features in time-series data. This mechanism allows the model to adaptively allocate weights based on the dynamic characteristics of the input data, thus performing outstandingly in complex time-series data processing. However, the LSTM+A model also faces some challenges. Due to the complexity of the model structure, it requires more data samples for training to ensure that the model can fully learn the patterns in the data. In practical applications, data acquisition may be limited by various factors such as cost, time, and resources. Therefore, how to improve the model's performance and generalization ability under limited data samples is an important direction for future research. In addition, the experimental results of this paper also show that dynamically adjusting the learning rate can effectively improve the model's convergence. In future optimization work, more advanced optimization algorithms and learning rate adjustment

DOI: 10.54254/2755-2721/146/2025.TJ21927

strategies can be explored to further improve the model's training efficiency and performance. In summary, the LSTM+A model proposed in this paper provides an effective solution for time-series data classification tasks, especially with broad application prospects in the field of wireless sensing. In the future, we will continue to optimize the model structure and training strategies to improve its performance and adaptability in more complex scenarios.

References

[1] *Yu T C, Lin C C. An intelligent wireless sensing and control system to improve indoor air quality: Monitoring, prediction, and preaction[J]. International Journal of Distributed Sensor Networks, 2015, 11(8): 140978.*

[2] *Li J, He S, Ming Z, et al. An intelligent wireless sensor networks system with multiple servers communication[J]. International Journal of Distributed Sensor Networks, 2015, 11(8): 960173.*

[3] *Yang Z, Zhou Z, Liu Y. From RSSI to CSI: Indoor localization via channel response[J]. ACM Computing Surveys (CSUR), 2013, 46(2): 1-32.*

[4] *Lin Y, Chen J, Cao Y, et al. Cross-domain recognition by identifying joint subspaces of source domain and target domain[J]. IEEE transactions on cybernetics, 2016, 47(4): 1090-1101.*

[5] *Yu Y, Si X, Hu C, et al. A review of recurrent neural networks: LSTM cells and network architectures[J]. Neural computation, 2019, 31(7): 1235-1270.*

[6] *Singh U, Determe J F, Horlin F, et al. Crowd forecasting based on wifi sensors and lstm neural networks[J]. IEEE transactions on instrumentation and measurement, 2020, 69(9): 6121-6131.*

[7] *Wang F, Liu J, Gong W. WiCAR: WiFi-based in-car activity recognition with multi-adversarial domain adaptation[C]//Proceedings of the International Symposium on Quality of Service. 2019: 1-10.*

[8] *Zhang J, Chen Z, Luo C, et al. MetaGanFi: Cross-domain unseen individual identification using WiFi signals[J]. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2022, 6(3): 1-21.*

[9] *Li F, Liang Y, Liu G, et al. Wi-TCG: a WiFi gesture recognition method based on transfer learning and conditional generative adversarial networks[J]. Engineering Research Express, 2024, 6(4): 045253.*

[10] *Sherstinsky A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network[J]. Physica D: Nonlinear Phenomena, 2020, 404: 132306.*

[11] *Al-Selwi S M, Hassan M F, Abdulkadir S J, et al. RNN-LSTM: From applications to modeling techniques and beyond—Systematic review[J]. Journal of King Saud University-Computer and Information Sciences, 2024: 102068.*

[12] *Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.*

[13] *Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[J]. arXiv preprint arXiv:1409.0473, 2014.*

[14] *Vijay G, Bdira E B A, Ibnkahla M. Cognition in wireless sensor networks: A perspective[J]. IEEE sensors journal, 2010, 11(3): 582-592.*

[15] *Niu Z, Zhong G, Yu H. A review on the attention mechanism of deep learning[J]. Neurocomputing, 2021, 452: 48-62.*