Multi-model Fine-tuning and Performance Optimization Based on the NTU-Fi-HumanID Dataset

Gengyang Xu

*SWJTU-Leeds Joint School, Southwest Jiaotong University, Chengdu, Sichuan, 611756, China*

el22gx@my.swjtu.edu.cn

**Abstract.** With the development of 6G network technology, human identification technology based on WiFi CSI data has also developed rapidly. This study aims to improve the recognition accuracy and reduce the computational cost of the NTU-Fi-HumanID dataset by adjusting the hyperparameters of the deep learning model without changing the model architecture. This paper optimizes 10 models and improves the performance of most models by adjusting the values of four hyperparameters. After parameter tuning, the performance of most models has been optimized, and the accuracy of the test set has been improved by up to 9.7%. This study found that when processing CSI data, the performance of time series models generally outperforms that of non-time series models. The performance improvement of non-time series models is mainly affected by the number of iterations. At the same time, the original results and the results after optimization verify the conclusion of previous studies that shallow models outperform deep models. This study provides a reference for hyperparameter selection for the NTU-Fi-HumanID dataset and improves the understanding of the key influencing factors of this dataset.

# Introduction

With the rapid development and increasing maturity of the sixth generation of mobile communication technology (6G), its high-speed transmission capability and high bandwidth provide a solid technical foundation for large-scale data collection and processing. In this context, human identification technology based on WiFi channel state information (CSI) has emerged and initial research results have been obtained. The NTU-Fi-HumanID dataset, which collects gait information of multiple subjects in different scenarios, provides a valuable data resource for research in this field.

Currently, WiFi sensing is mainly used in three categories: detection, recognition, and estimation. Specifically, it is used in application scenarios such as human detection, activity recognition, and positioning and tracking. However, this technology still faces challenges such as robustness, privacy and security, and network coexistence [1]. In terms of human identification, a WiFi identification system based on few-shot learning, CAUTION, has been proposed. It can use a small amount of CSI data to construct an accurate user identification model and effectively detect unknown intruders [2]. Meanwhile, a deep learning framework for WiFi sensing, SenseFi, has been proposed, which integrates multiple learning algorithms, hardware platforms, and datasets. It has been found that shallow models outperform deep models in WiFi sensing tasks and that the generalization ability of models can be further improved through transfer learning and unsupervised learning [3].

This study aims to optimize the performance of multiple models for the NTU-Fi-HumanID dataset by adjusting hyperparameters and to understand the impact of different hyperparameters and different models on this dataset. This paper introduces the selected dataset and the multiple models that have been optimized and draws conclusions by comparing and analyzing the hyperparameter combinations and performance of multiple models before and after optimization.

# Data and methods

## Data

The dataset used in this study is NTU-Fi-HumanID, a high-quality WiFi sensing dataset developed by a research team from Nanyang Technological University, focusing on human identification tasks. The dataset is part of the SenseFi framework [3] and is designed to provide rich data support for WiFi sensing research. The dataset was collected using the Atheros CSI tool, which can provide high-resolution subcarrier data. Each sample has dimensions (3,114,500), representing the number of antenna pairs, the number of subcarriers, and the length of the time series [2]. The dataset contains walking gait data from 14 different users, with 546 training samples and 294 test samples and has undergone preliminary preprocessing such as noise removal and data normalization [2]. The high resolution and diversity of this dataset make it ideal for studying WiFi sensing tasks such as human identification, activity recognition, and environmental perception and provide a valuable resource for exploring new WiFi sensing technologies and algorithms.

|  |
| --- |
| 14volunteer |
| **Figure 1.** 14 volunteers' gait in the same scene(Photo credit : Original ) |

|  |
| --- |
| 14volunteer |
| **Figure 2.** Gait of the same subject in three scenarios(Photo credit : Original ) |

Figure 1 shows the gait visualization heat maps of 14 subjects in the same scene. Each sub-map represents a subject, the horizontal axis is the time series, and the vertical axis is the sub-carrier index, which represents different frequency components. In the same scene, there are obvious differences between the sub-maps, which show that the gait characteristics (strength, speed, distance, etc.) of different people can be captured by CSI.

Figure 2 shows the gait visualization heat maps of the same subject in three scenarios. It is found that the environment also has a significant impact on CSI. In particular, Scenario C and Scenario A are both the same wearing, but the interference introduced in Scenario C (people moving and talking around) has a significant impact on the heat map.

## Method

### *Model introduction*

A total of 10 models have been optimized through parameter adjustment and can be roughly divided into two types based on their characteristics: non-sequential models and sequential models.

Non-sequential models include Multilayer Perceptron (MLP), LeNet (LeNet), Residual Network 18 (ResNet18), Residual Network 50 (ResNet50), ResNet101. MLP is a classical feed-forward neural network suitable for processing static data, and features are extracted through multiple fully connected layers. LeNet is an early convolutional neural network (CNN) mainly used for image classification tasks, and spatial features are extracted through convolutional and pooling layers. Neither of these models processes time series data. ResNet is a deep convolutional neural network that solves the problem of gradient vanishing in deep networks by introducing residual connections [4]. ResNet18, ResNet50 and ResNet101 are versions with different depths for image classification and object detection tasks. These models extract spatial features through convolutional layers but do not process time series data.

Sequential models include Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM), and Convolutional Neural Network + Gated Recurrent Unit (CNN+GRU). RNN is a type of neural network capable of processing sequential data. It is suitable for time series data because it remembers past input information through hidden states [5]. However, standard RNNs are susceptible to the problem of vanishing gradients, which limits their performance on long sequences. GRU and LSTM are two improved variants of RNNs that better handle long sequences by introducing gating mechanisms to avoid the vanishing gradients problem. GRU controls the flow of information through update gates and reset gates [6], whereas LSTM controls the storage and release of information through the input gate, forget gate and output gate [7]. BiLSTM is a bidirectional version of LSTM that can process forward and backward sequence information simultaneously and is suitable for scenarios where past and future information needs to be considered simultaneously [8]. CNN+GRU This combined model combines the spatial feature extraction capability of CNN with the temporal sequence processing capability of GRU, and is suitable for tasks that require the simultaneous processing of spatial and temporal features, such as video analysis and multimodal data processing.

### *Experimental process*

First, this paper performs data pre-processing on the dataset and uses the following indicators as the comparison indicators after subsequent parameter tuning: accuracy, loss rate, Flops, Params, and average iteration time. Since the model results are somewhat random [9], the original results and each subsequent adjustment are run three times independently and the average value is taken, so as to avoid the influence of chance as much as possible.

Then, this paper adjusts the four hyperparameters of Epoch, Batch size, Network complexity, and learning rate, four types of hyperparameters were adjusted, either increased or decreased from the original hyperparameters. The epoch step was 10, the batch size was 64, 32, or 16, the network complexity was adjusted by the size and number of layers, and the learning rate was 1e-2, 1e-3, 1e-4, or 1e-5. A single hyperparameter was adjusted until the result performance significantly deteriorated, and the single hyperparameter with the best performance was selected.

Finally, all the best combinations of hyperparameters for a model are combined and the results are compared with the results of all single best hyperparameters and the original results to ensure that the best combination of hyperparameters performs better than any other single combination.

## Evaluation metrics

The quantifiable evaluation metrics in this study can be mainly divided into recognition accuracy and computational complexity. Recognition accuracy includes the test set recognition accuracy and the loss rate. The test set accuracy is the ratio of the number of samples correctly classified (or predicted) by the model on the test data set to the total number of samples. The accuracy can reflect the generalization ability of the model. The test set loss rate is the average value of the loss function value calculated by the model on the test data set. The loss function measures the difference between the model's predicted value and the true value. The smaller the loss value, the closer the model's prediction is to the true value.

Computational complexity includes the number of floating-point operations, the number of model parameters, and the average iteration time. The number of floating-point operations refers to the total number of floating-point operations performed during model execution. This indicator can measure the computational requirements and running efficiency of the model. The number of model parameters refers to the number of trainable parameters in the model, including weights and biases. These parameters determine the complexity and expressiveness of the model. The average iteration time refers to the average time required for the model to complete an iteration (i.e., one forward propagation and one backward propagation) during training, which reflects the speed of model training.

# Results and analysis

This section will compare each model before and after performance optimization in detail and observe the impact of different hyperparameters.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table 1.** Original model data hyperparameters and results | | | | | | | | |
| **Model name** | **Epoch** | **Batch size** | **Network complexity** | **Learning rate** | **Accuracy** | **Loss rate** | **Flops (MFlops)** | **Params (MParams)** |
| MLP | 50 | 64 | 1024-128 | 1e-3 | 92.5% | 0.36 | 175.24 | 175.24 |
| LeNet | 50 | 64 | 128 | 1e-3 | 97.5% | 0.09 | 28.17 | 0.48 |
| ResNet18 | 50 | 64 | default | 1e-3 | 92.8% | 0.24 | 54.22 | 11.19 |
| ResNet50 | 50 | 64 | default | 1e-3 | 92.8% | 0.38 | 90.69 | 23.57 |
| ResNet101 | 50 | 64 | default | 1e-3 | 83.8% | 0.77 | 166.87 | 42.59 |
| RNN | 75 | 64 | 64 | 1e-3 | 89.8% | 0.30 | 13.09 | 0.03 |
| GRU | 50 | 64 | 64 | 1e-3 | 98.8% | 0.06 | 39.39 | 0.08 |
| LSTM | 50 | 64 | 64 | 1e-3 | 94.4% | 0.27 | 52.48 | 0.11 |
| BiLSTM | 50 | 64 | 128 | 1e-3 | 96.3% | 0.17 | 104.96 | 0.21 |
| CNN+GRU | 200 | 64 | 64 | 1e-3 | 85.6% | 1.91 | 46.66 | 0.06 |

Table 1 shows the original hyperparameter combinations and their performance. It can be seen that under the default hyperparameter combination, the performance of each model on the test set is already quite good, but the performance of some models is relatively low. For example, ResNet101 has a computational complexity second only to MLP, but its accuracy is only 83.8%, lower than the 89.9% accuracy of RNN, the model with the lowest complexity, and far lower than the 98.8% accuracy of GRU. This result may be attributed to the fact that the time series characteristics of the CSI data cannot be well captured by ResNet101, while RNN and GRU, as time series models, have certain advantages in processing this type of data. In addition, ResNet18 and ResNet50, which are similar models, also performed significantly better than ResNet101. This may be attributed to the fact that complex models are more susceptible to noise, resulting in overfitting or deviations in feature extraction. This result is consistent with the conclusion of the original paper, which states that shallow models outperform complex models in terms of processing CSI data [3].

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table 2.** Hyperparameters and results of the optimized model | | | | | | | | |
| **Model name** | **Epoch** | **Batch size** | **Network complexity** | **Learning rate** | **Accuracy** | **Loss rate** | **Flops (MFlops)** | **Params (MParams)** |
| MLP | 70 | 64 | 512-64 | 1e-4 | 94.3%  +1.9% | 0.19 | 87.59 | 87.59 |
| LeNet | 50 | 64 | 128 | 1e-3 | 97.5%  +0.0% | 0.09 | 28.17 | 0.48 |
| ResNet18 | 60 | 64 | default | 1e-3 | 97.9%  +5.1% | 0.08 | 54.22 | 11.19 |
| ResNet50 | 70 | 64 | default | 1e-3 | 95.0%  +2.2% | 0.20 | 90.69 | 23.57 |
| ResNet101 | 60 | 64 | default | 1e-3 | 93.5%  +9.7% | 0.25 | 166.87 | 42.59 |
| RNN | 75 | 64 | 64 | 1e-3 | 89.8%  +0.0% | 0.30 | 13.09 | 0.03 |
| GRU | 40 | 16 | 64 | 1e-3 | 99.0%  +0.2% | 0.03 | 39.39 | 0.08 |
| LSTM | 60 | 32 | 64 | 1e-3 | 97.0% +2.6% | 0.11 | 52.48 | 0.11 |
| BiLSTM | 60 | 32 | 256 | 1e-2 | 97.3% +1.0% | 0.11 | 104.96 | 0.21 |
| CNN+GRU | 200 | 64 | 64 | 1e-3 | 85.6%  +0.0% | 1.91 | 46.66 | 0.06 |

Table 2 shows the optimized hyperparameter combinations and their performance. As can be seen from the table, the performance of most models has been improved. Among them, ResNet101, as a more complex non-time series model, has achieved the largest improvement, with an accuracy rate of 93.5%. However, among the 10 models, its final accuracy rate is still relatively low. The accuracy rates of the LeNet, RNN, and CNN+GRU models did not improve after parameter adjustment. Among the models whose performance improved, the improvement of the non-time series model was significantly higher than that of the time series model, with an accuracy improvement of 1.9% to 9.7%, but the performance still generally lagged behind the time series model. All models with improved performance adjusted the Epoch, and except for GRU, the number of iterations was increased, indicating that most models still need to increase the number of iterations to learn the features in the CSI data.

For the MLP model, reducing the first fully connected layer (adjusting the input from 1024 to 512 and the output from 128 to 64) can reduce the iteration time to half the original time, while the accuracy only decreases by 0.3%. By adjusting other hyperparameters, the model performance can be improved while greatly reducing the computational complexity, which is an improvement of 1.9% compared to the original hyperparameters.

# Conclusion

This study improves the model's performance in terms of accuracy and loss on the test set and reduces the computational complexity by adjusting the hyperparameters one by one and ultimately finding the best combination of hyperparameters. After parameter optimization, the following findings can be obtained:

1) After parameter tuning, the improvement in recognition accuracy of non-time series models (MLP, LeNet, ResNet18, ResNet50, ResNet101) is significantly better than that of time series models (RNN, GRU, LSTM, BiLSTM, CNN+GRU). This phenomenon can be attributed to the fact that the CSI channel inherently contains time series information [10], so time series models have an advantage when processing this type of data.

2) The performance improvement of non-sequential models is mainly affected by the number of iterations. Since non-sequential models are not naturally suitable for processing time-series data, they require more iterations to learn data features, especially for network models with more complex structures.

3) The results of this study verify the conclusions of previous studies, that is, shallow models perform better than deep models when processing CSI data, and this conclusion is still valid even after parameter optimization.

This study explored the impact of different hyperparameters on model performance by finding the optimal hyperparameter combination for multiple models for the NTU-Fi-HumanID dataset, providing a reference for selecting hyperparameters for subsequent research based on this dataset. In the future, more advanced optimization algorithms, such as particle swarm optimization, can be considered to efficiently find more accurate hyperparameter combinations.

# REFERENCES

1. Y. Ma, G. Zhou, and S. Wang, “WiFi Sensing with Channel State Information: A Survey,” ACM Comput. Surv. 52, Article 46 (Jun. 2019).
2. D. Wang, J. Yang, W. Cui, L. Xie, and S. Sun, “CAUTION: A Robust WiFi-Based Human Authentication System via Few-Shot Open-Set Recognition,” IEEE Internet of Things Journal 9, 17323–17333 (Sep. 2022).
3. J. Yang, X. Chen, H. Zou, C. Xiaoxuan Lu, D. Wang, S. Sun, and L. Xie, “SenseFi: A library and benchmark on deep-learning-empowered WiFi human sensing,” Patterns 4, 100703 (Mar. 10, 2023).
4. N. Hnohom, S. Mekruksavanich, T. Theeramunkong, and A. Jitpattanakul, “Efficient Residual Neural Network for Human Activity Recognition using WiFi CSI Signals,” in 2024 9th International Conference on Information and Education Innovations (ICIEI), Verbania, Italy, Apr. 12–14, 2024, pp. 7.
5. A. M. Ali, P. Hendriks, N. Popping, S. Levi, and A. A. Naveed, “Comparison of Machine Learning Algorithms for Wi-Fi Sensing Using CSI Data,” Electronics 12, no. 3935 (2023).
6. K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation,” Digital Signal Processing 138, Art. no. 104056 (2023).
7. S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Computation 9, 1735–1780 (15 Nov. 1997).
8. S. Siami-Namini, N. Tavakoli, and A. S. Namin, “The Performance of LSTM and BiLSTM in Forecasting Time Series,” in 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 2019, pp. 3285–3292.
9. S. Scardapane and D. Wang, “Randomness in neural networks: an overview,” Electronics 12, no. 3935 (2023).
10. S. I. A. Showmik, T. F. Sanam, and H. Imtiaz, “Human Activity Recognition from Wi-Fi CSI Data Using Principal Component-based Wavelet CNN,” Digital Signal Processing 138, Art. no. 104056 (2023).