Deep Feature Extraction and Optimal Neighbor Selection Method for RFID-Based Localization

Cui Lizhi[1,2,a], **Jiang Zhen**[1,2,b,*]

[1]Country School of Electrical Engineering and Automation, Henan Polytechnic University, Jiaozuo, China
[2]Henan Key Laboratory of Intelligent Detection and Control for Coal Mine Equipment, Jiaozuo, China
[a]clzh0308@126.com, [b]1224113498@qq.com
**Corresponding author*

Abstract: *Aiming at the issues of inadequate positioning accuracy in RFID indoor positioning technology, a localization method based on Convolutional Neural Network (CNN) feature extraction and Weighted K-Nearest Neighbors (WKNN) is proposed. This method integrates the fingerprint features of Received Signal Strength (RSSI) and Phase. Firstly, the CNN is trained for feature extraction and weight learning using the offline RSSI/Phase dataset. Then, during online matching, the RSSI/Phase data collected at the test points are processed by the trained CNN for feature extraction. Finally, the WKNN algorithm is employed to predict the labeled location of the test points. In the WKNN prediction stage, a Genetic Algorithm (GA) is adopted to optimize the weights of RSSI and Phase, respectively. The experimental results of the proposed method are compared with those of CNN, WKNN, and LANDMARC localization methods, and the impact of different CNN architectures and fingerprint datasets on positioning accuracy is analyzed. The experimental results show that the proposed method achieves significant improvement in both positioning accuracy and adaptability.*

Keywords: *Radio Frequency Identification; Convolutional Neural Network; Weighted k-Nearest Neighbors; Genetic Algorithm*

1. Introduction

With the rapid development of industrial technology, the importance of radio frequency identification (RFID) technology has become increasingly prominent across various industries[1-2]. As an efficient non-contact automatic identification technology, RFID has been widely adopted in indoor positioning due to its advantages such as low power consumption, non-line-of-sight communication, strong environmental adaptability, and precise positioning. Compared to traditional positioning technologies like GPS[3] and Wi-Fi[4], RFID enables higher-precision real-time positioning in indoor environments, demonstrating significant advantages in complex application scenarios such as warehouse management and asset tracking. However, in practical deployment, RFID[5] positioning still faces multiple challenges posed by complex environments, including multipath propagation, signal delay, human interference, and signal attenuation, all of which can affect the final positioning accuracy.

In RFID indoor positioning technology, distance-based methods primarily include Received Signal Strength Indicator (RSSI), Angle of Arrival (AOA)[6], Time of Arrival (TOA), and Time Difference of Arrival (TDOA)[7]. However, these methods are susceptible to interference from factors such as multipath effects and non-line-of-sight propagation in practical complex environments, making it difficult to meet high-precision positioning requirements. In response, researchers have proposed various improvement solutions. Reference[8] proposes a hybrid similarity-weighted KNN (MWKNN) localization algorithm based on Mean Shift clustering for adaptive K-value selection and geometric position optimization, which holds significant importance in enhancing indoor positioning accuracy. The algorithm employs Mean Shift clustering technology to adaptively screen reference points, effectively identifying and eliminating outliers. Subsequently, it dynamically optimizes the K-value by integrating the geometric distribution characteristics of neighboring points, significantly reducing the interference of high-error neighbors on the positioning results. Experimental results demonstrate that, compared to traditional WKNN algorithms, this method achieves notable improvements in both positioning accuracy and stability. Xie. Y[9] et al. expanded the system's application scope by deploying multiple readers and implementing multi-tag positioning based on RSSI within the LANDMARC framework and its improved algorithms. Peng. C[10] proposed an innovative passive RFID tag localization method based on a deep convolutional neural network (CNN). This method designs a network architecture comprising three convolutional and pooling layers, which constructs normalized RSSI and PDoA data into a two-dimensional image format as input, fully leveraging CNN's advantages in image feature extraction. During the offline training phase, the network learns the complex mapping relationships between position and signal features from a large amount of data. In the online localization phase, the trained model is directly utilized for end-to-end coordinate prediction. This approach not only reduces the workload of manual feature engineering but also effectively mitigates the impact of environmental noise on localization data through CNN's powerful feature learning capability, demonstrating good robustness in complex indoor environments.

In summary, in RFID positioning technology, the received signal strength (RSSI) and the received phase (Phase) are susceptible to on-site environmental influences, thereby reducing positioning accuracy. Therefore, a hybrid localization method combining convolutional neural networks and weighted KNN is proposed to improve positioning accuracy. Deep learning, with its capabilities for big data analysis and learning, has become a new hotspot, offering multiple advantages in data processing. In the method studied in this paper, first, a CNN model is trained using offline RSSI/Phase datasets, and the trained convolutional neural network model is employed to extract high-dimensional features from the offline database. Subsequently, the trained CNN model is used to extract features from the online test dataset (RSSI/Phase). Finally, weighted KNN predicts the coordinates of the online test data based on the extracted features, with a genetic algorithm optimizing the weights of RSSI and Phase, respectively, during the WKNN localization stage. Experimental results demonstrate that this method significantly enhances both the accuracy and robustness of RFID positioning.

2. Research Methodology

2.1. Feature Extraction Based on Convolutional Neural Network Model

First, an offline database and an online test database are constructed, and the collected data undergo transformation and preprocessing. Then, the offline database is used to train the convolutional neural network, which subsequently performs feature extraction on the data.

As shown in Figure 1, during the construction of the offline database, sampling points are first arranged in advance in the experimental area, and the label coordinates (x, y) cm of each sampling point are recorded. Then, the corresponding received signal strength indication (RSSI) and signal phase (Phase) data for the coordinate positions of the sampling points are collected, and finally, they are associated and stored in the offline database. During the construction of the test database, test points are selected in the experimental area, and the RSSI and Phase data of the labels at the test point locations are collected and stored in the test database.

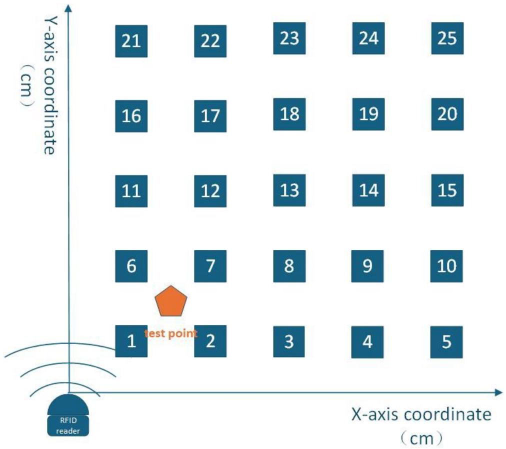


Figure 1: CNN Feature Extraction Overview Diagram

2.1.1. Data Collection and Processing

Based on the multiple measurements of RSSI and Phase data for the i-th sampling point obtained from the reader, the RSSI and Phase fingerprint features corresponding to the label location of that sampling point are constructed. The collected data for the label at this sampling point is organized in a grid structure, where the RSSI data and Phase data for the i-th sampling point can be expressed as:

$$r^i = (r_1^i, r_2^i, r_3^i, \cdots, r_n^i)$$

$$p^i = (p_1^i, p_2^i, p_3^i, \cdots, p_n^i)$$

Here, *n* represents the number of times the reader collects data at the i-th sampling point. Received Signal Strength Indicator (RSSI) is one of the fundamental features of RFID positioning systems, reflecting the attenuation of propagation loss between the reader and the tag. Phase data, on the other hand, compensates for the shortcomings of RSSI, such as low positioning accuracy and susceptibility to environmental changes, enabling high-precision indoor positioning. When using fingerprint database data for positioning, both RSSI and Phase data can serve as fingerprint database information for the sampling point coordinates (x, y).

To accelerate the convergence speed during model training and prevent training imbalance caused by excessively large numerical ranges of certain features, as well as to improve training accuracy and avoid gradient explosion or vanishing, data normalization is required. The expression is as follows:

$$\tilde{r}_k^i = \frac{r_k^i - min(r^i)}{max(r^i) - min(r^i)}, \qquad k = ,1,2,\cdots,n \tag{1}$$

$$\tilde{p}_k^i = \frac{p_k^i - min(p^i)}{max(p^i) - min(p^i)}, k = ,1,2,\cdots,n \tag{2}$$

Here, r_k^i and p_k^i represent the raw RSSI and Phase values received by the reader, respectively. $min(r^i), max(r^i), min(p^i), max(p^i)$ denote the minimum and maximum values of RSSI and Phase in the r^i, p^i datasets, respectively. Normalization maps the received data to a common scale, ensuring balanced weights for RSSI and Phase. This allows the model to learn each signal feature equally, reduces the impact of noise on the signals, and enhances the robustness of the system.

The normalized RSSI/Phase dataset for the i-th sampling point can be expressed as:

$$r^i = (\tilde{r}_1^i, \tilde{r}_2^i, \tilde{r}_3^i, \cdots, \tilde{r}_n^i)$$

$$p^i = (\tilde{p}_1^i, \tilde{p}_2^i, \tilde{p}_3^i, \cdots, \tilde{p}_n^i)$$

In the actual indoor measurement environment, when collecting offline data, the offline database should include the sampling point coordinates (x, y), received signal strength (RSSI), and signal phase (Phase) information. At M different sampling points, the RSSI data and Phase data can be expressed as follows:

$$\mathrm{R} = \{(r^1, l^1), (r^2, l^2), \cdots, (r^M, l^M)\}$$

$$\mathrm{P} = \{(p^1, l^1), (p^2, l^2), \cdots, (p^M, l^M)\}$$

Here, r^M and p^M represent the RSSI and Phase values corresponding to the M-th sampling point, respectively; l^M denotes the position coordinate information corresponding to the M-th sampling point.

By transforming and processing the collected data, the raw data is converted into a two-dimensional grid structure that can be recognized by the convolutional neural network. This enables the CNN to train on the combined RSSI and Phase dataset and extract fingerprint features, making the fingerprint features of labels at different locations more discriminative.

2.1.2. Method for Improving Convolutional Neural Network Feature Extraction

In RFID positioning technology, convolutional neural networks can automatically identify information useful for positioning tasks from the dataset, reducing the impact of noise on positioning. Compared to other methods, convolutional neural networks offer the following advantages for feature extraction:

1) CNN utilizes convolutional layers to capture local information and pooling layers to reduce the dimensionality of the feature space, enabling automatic learning of complex patterns in the input data;

2) CNN exhibits strong robustness, enabling it to effectively extract useful features even in the presence of noise and interference;

3) By sliding convolutional kernels over the data, CNN achieves parameter sharing, which reduces model complexity and improves training efficiency.

The feature extraction framework of the convolutional neural network is illustrated in Figure 2. During offline dataset feature extraction, the RFID reader collects raw RSSI and Phase data from M sampling points. The collected dataset undergoes normalization to form datasets R and P. The convolutional neural network is trained using the offline fingerprint database constructed from the M sampling points. After training, the offline fingerprint database is re-input into the trained convolutional neural network to extract features from the offline database. For online test dataset feature extraction, the RSSI and Phase data of the target tag are collected to generate a test dataset, which is then processed by the trained convolutional neural network to extract features from the test data.

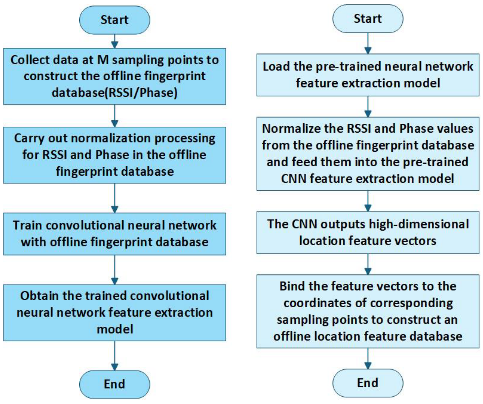


Figure 2: Feature Extraction Flowchart of Convolutional Neural Network

Figure 3 illustrates the training process and parameters of a designed convolutional neural network model. CNN is a network structure with weight sharing, which helps reduce the number of weights and the complexity of the network. Consisting of convolutional layers, pooling layers, and fully connected layers, CNN is a type of multilayer perceptron designed to recognize two-dimensional shapes.

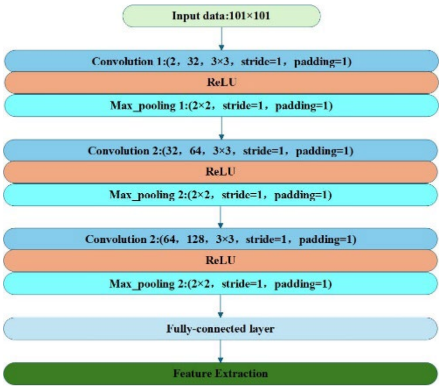


Figure 3: Convolutional Neural Network Model Design Diagram

The core of CNN is the convolution operation, which aims to slide a convolution kernel over the input data and perform weighted calculations to generate a feature map. For a two-dimensional convolution operation, the value of the output pixel (i , j) is calculated using the following formula:

$$O_k(\mathrm{i},\mathrm{j}) = b_k + \sum_{c=0}^{C_{in}-1} \sum_{m=0}^{K_h-1} \sum_{n=0}^{K_w-1} I_c(\mathrm{i} + \mathrm{m} - P_h, \mathrm{j} + \mathrm{n} - P_w) \cdot W_{k,c}(\mathrm{m}, \mathrm{n}) \tag{3}$$

Here, b_k is the bias of the k-th output channel. C_{in} is the number of channels in the input feature map. K_h, K_w are the height and width of the convolution kernel, respectively. $I_c\ (i,j)$ is the pixel at position *(i, j)* in the c-th channel of the input feature map. P_h, P_w are the padding sizes, typically set to 1. $W_{k,c}(m,n)$ is the value of the convolution kernel corresponding to the k-th output channel and the c-th input channel at position (m, n). The indices i and j denote the coordinate indices of the output image.

In convolutional neural networks, the ReLU activation function is used to mitigate issues such as gradient explosion and vanishing gradients. Its mathematical definition is ReLU(x) = max(0, x), where x represents a single pixel value in the output feature map of a convolutional layer. After passing the convolution result (i, j) through the ReLU activation function, the nonlinear output is obtained as:

$$A_k(\mathrm{i,j}) = \mathrm{ReLU}\big(O_k(\mathrm{i,j})\big) \tag{4}$$

The activation function enhances the model's nonlinear fitting capability and improves the overall expressive performance of the network. Subsequently, the feature map output by the convolutional layer is fed into the pooling layer. Max pooling is employed in the pooling layer to extract the maximum value from the local regions of the feature map, expressed as:

$$P(i,j) = \max_{0\le m<f}\max_{0\le n<f} A(s\cdot i+m, s\cdot j+n) \tag{5}$$

Here, $A(s\cdot i+m, s\cdot j+n)$ in the input feature map, centered at position *(i, j)*, where i and j are the row and column coordinates of the feature map, a region with a pooling window size of f×f is processed. After the pooling operation, the maximum value within this region is selected. s is the pooling stride, meaning the window moves s pixels across the input image for the next pooling step.

To map the high-level semantically abstract feature maps extracted by the neural network to the output space, fully connected layers are employed to transform feature representations into task decisions. The output $h^{(g)}$ after the g-th fully connected layer can be expressed as:

$$h^{(g)} = ReLU\big(W^{(g)}x^{(g-1)} + b^{(g)}\big) \tag{6}$$

Here, $x^{(g-1)} \in \mathbb{R}^n$ is the flattened input vector from the (g-1)-th layer. $W^{(g)} \in \mathbb{R}^{m\times n}$ is the weight matrix of the g-th layer. $b^{(g)} \in \mathbb{R}^m$ is the bias vector. ReLU is the activation function.

During the training of the convolutional neural network, the design of the loss function is crucial for improving model performance. In this paper, a comprehensive loss function is designed, with coordinate prediction as the main task and RSSI and Phase prediction as auxiliary tasks. The adaptive multi-task comprehensive loss function based on homoscedastic uncertainty is defined as:

$$Loss = \frac{1}{2\sigma_1^2}\cdot MSE_{coords} + \frac{1}{2\sigma_2^2}\cdot MSE_{RSSI} + \frac{1}{2\sigma_3^2}\cdot MSE_{Phase} + \ln(\sigma_1\sigma_2\sigma_3) \tag{7}$$

Here, σ_1, σ_2, and σ_3 are the adaptive uncertainty parameters for the coordinate prediction task, the RSSI prediction task, and the Phase prediction task, respectively. These are learnable parameters of the network, updated in real-time during iterative training. $\frac{1}{2\sigma_i^2}$ presents the adaptive loss weight for each task, and $\ln(\sigma_1\sigma_2\sigma_3)$ is the uncertainty regularization term, which constrains parameter updates, prevents weights from approaching zero indefinitely, and ensures training stability.

During the training iteration process, if RSSI or Phase prediction is subject to environmental interference and high noise levels, the corresponding uncertainty parameters σ_2 and σ_3 adaptively increase, Consequently, their loss weights $\frac{1}{2\sigma_2^2}$ and $\frac{1}{2\sigma_3^2}$ automatically decrease, thereby weakening the interference of high-noise auxiliary tasks on the main task. If the auxiliary tasks exhibit high prediction stability and low noise, the uncertainty parameters decrease, leading to an adaptive increase in the corresponding weights, which fully leverages the feature constraints of the auxiliary tasks. For the core coordinate prediction main task, the model adaptively learns a smaller uncertainty, ensuring that it consistently occupies the dominant weight in the loss function. This naturally fulfills the training requirement of prioritizing the main task.

To further standardize weight values and enhance the iterative stability of the loss function, the task weights derived from uncertainty are normalized. The original adaptive weights for the three tasks are defined as follows $\alpha = \frac{1}{2\sigma_1^2}, \beta = \frac{1}{2\sigma_2^2}, \gamma = \frac{1}{2\sigma_3^2}$,calculate the global weight sum and perform normalization:

$$\text{Total Weight} = \alpha + \beta + \gamma \tag{8}$$

$$\alpha' = \frac{\alpha}{\text{Total Weight}}, \beta' = \frac{\beta}{\text{Total Weight}}, \gamma' = \frac{\gamma}{\text{Total Weight}} \tag{9}$$

After normalization, the sum of the weights for all tasks is always equal to 1, which further strengthens the dominance of the main task weight. Simultaneously, the adaptive nature of the uncertainty weights is preserved, effectively avoiding issues such as imbalance in the magnitude of multi-task weights and chaotic gradient updates.

The adaptive multi-task loss function improved based on homoscedastic uncertainty in this paper completely overcomes the limitations of traditional manual parameter tuning. By enabling the model to autonomously learn the training uncertainty of each task and dynamically optimize the loss weights of the main and auxiliary tasks, combined with global normalization constraints, it achieves intelligent and adaptive multi-task collaborative optimization throughout the entire training process. Compared to the manual k-value adjustment strategy, this method exhibits stronger robustness and superior generalization performance. It can adapt to varying complex environments and different training stages of the model, effectively enhancing the localization accuracy and environmental adaptability of the network.

CNN-based feature extraction can automatically extract features, capture complex signals, and model nonlinear relationships. This approach not only extracts useful features but also reduces feature dimensionality, enhancing the model’s robustness and accuracy.

2.2. Research on GA-Weighted KNN Localization Algorithm

2.2.1. Improved Weighted KNN Localization Algorithm

During the online testing phase, the RSSI and Phase data of the test tags are collected and processed by the trained CNN for feature extraction, resulting in the feature datasets R_{online}, P_{online}. A total of G sets of data are collected for each test tag, generating the corresponding feature datasets as $R_{online} = [R^1, R^2, \cdots, R^G], P_{online} = [P^1, P^2, \cdots, P^G]$. To achieve high-precision tag localization, a localization method based on the Quantum Particle Swarm Optimization algorithm is proposed, which separately optimizes the weights of RSSI and Phase in the weighted KNN localization process.

The weighted KNN localization algorithm involves matching and computing the test dataset (after CNN feature extraction) with the offline dataset. The computational steps are as follows:

(1)The Euclidean distance between the RSSI/Phase database of the test tag and the RSSI/Phase database of the reference tag is calculated. The formula for the Euclidean distance of the i-th sampling point label is as follows:

$$D_i = \sqrt{w_{RSSI}^2 \cdot (R_{online} - R_{offline})^2} + \sqrt{w_{Phase}^2 \cdot (P_{online} - P_{offline})^2}, 0 < i \le M \tag{10}$$

Here, w_{RSSI}^2, w_{Phase}^2 represent the weights of RSSI and Phase, respectively; $R_{online}, P_{online}, R_{offline}, P_{offline}$ denote the test data and offline data after CNN feature extraction, respectively.

(2)Based on the Euclidean distance between the online test RSSI/Phase database and the offline RSSI/Phase data of the sampling point labels, the k reference labels with the smallest Euclidean distances are selected as neighboring reference labels. The weight calculation formula for each neighboring reference label is as follows:

$$w_i = \frac{\frac{1}{D_i^2+\epsilon}}{\sum_{i=1}^{k} \frac{1}{D_i^2+\epsilon}}, 0 < i \le k \tag{11}$$

Here, ϵ represents a minimal constant, such as 10^{-6}, a very small positive number, to prevent division by zero in the numerator.

(3)The coordinates of the test tag are estimated based on the coordinates and weights of the neighboring reference labels:

$$(\mathrm{x}, \mathrm{y}) = \sum_{i}^{k} w_i \, (x_i, y_i) \tag{12}$$

Formula (12) represents the weighted average formula for coordinate calculation. The right-hand side denotes the weighted sum of the coordinates of the k neighboring reference labels. The coordinates $(x_i, y_i)$of each reference label are multiplied by its corresponding weight w_i. Finally, the products of the coordinates and weights for all k neighboring reference labels are summed to obtain the position coordinates of the test label.

(4)Based on the WKNN calculation described above for estimating the coordinates of the test tags at the test points, the computational steps (10) to (12) can be repeated to estimate the position coordinates of all test tags.

2.2.2. Genetic Algorithm optimizes RSSI and Phase weights

In the proposed RFID indoor positioning system, the genetic algorithm optimizes the feature fusion weights by initializing a population of chromosomes (encoded as weight vectors for RSSI and phase) with the constraint that their sum equals 1. Through an iterative evolutionary process, the fitness of each chromosome is evaluated by calculating the reciprocal of the positioning error. Based on fitness, operations such as selection, crossover, and mutation are performed to continuously explore the weight space and escape local optima. Ultimately, the optimal weight combination that minimizes positioning error is decoded, achieving an adaptive balance between the contributions of the two features.

The genetic algorithm completes population iterative updates through three operators: selection, crossover, and mutation, gradually screening the optimal weight individuals. The specific calculation formulas and rules are as follows:

(1)Selection operator: The roulette wheel selection method is used to screen high-quality individuals. The probability of an individual being selected is positively correlated with its fitness value. The probability calculation formula is as follows:

$$P_i = \frac{Fitness(x_i)}{\sum_{j=1}^{N} Fitness(x_j)} \tag{13}$$

In the formula, P_i is the selection probability of the i-th individual. Through probability-based screening, weight combinations with high fitness are retained, while inferior individuals are eliminated, ensuring the overall optimization direction of the population. $Fitness(x)$ is the fitness function, used to evaluate the quality of individuals and select the optimal weight combination, where $x = (\alpha, \beta)$ denotes the weights of RSSI and Phase, respectively.

(2)Crossover operator: A single-point real-number crossover method is used to perform crossover operations on the population individuals. Two parent individuals, $Ind_1 = [\alpha_1, \beta_1]$ and B $Ind_2 = [\alpha_2, \beta_2]$, are randomly selected, and after crossover, offspring individuals are generated. The calculation formula is as follows:

$$\begin{cases} \alpha_1' = \omega\alpha_1 + (1-\omega)\alpha_2 \\ \beta_1' = \omega\beta_1 + (1-\omega)\beta_2 \end{cases} \tag{14}$$

In the formula, $\omega = (0,1)$ is the random crossover weight coefficient, used to achieve random fusion of weight parameters and enrich population diversity.

(3)Mutation operator: To prevent the algorithm from falling into local optima, a Gaussian mutation operator is introduced to apply random perturbations to the individual weight parameters. The mutation update formula is as follows:

$$\begin{aligned} \alpha^* &= \alpha + \delta \cdot \xi \\ \beta^* &= \beta + \delta \cdot \xi \end{aligned} \tag{15}$$

In the formula, δ is the mutation step size, and ξ is a random variable that follows a standard Gaussian distribution. By introducing small random perturbations, the weight search space is expanded, thereby enhancing the global optimization capability.

The genetic algorithm-based weight optimization strategy proposed in this paper addresses the drawbacks of RSSI and Phase weights being prone to local optima and poor adaptability to high-noise environments. Through global iterative optimization via the genetic algorithm, the optimal baseline weights for RSSI and Phase can be determined. This method effectively suppresses environmental noise interference on the weight allocation for positioning tasks, accurately identifies the optimal weight proportion for RSSI and Phase, and further enhances the model's localization accuracy, convergence stability, and generalization robustness in complex scenarios.

In indoor experimental environments, the reliability of RSSI or Phase is often compromised due to indoor environmental changes. The Genetic Algorithm (GA) algorithm is capable of automatically adjusting the weight coefficients of RSSI or Phase, thereby mitigating the impact of environmental factors on localization and enhancing the adaptability of the positioning system.

3. Experimental Verification

3.1. Experimental Equipment and Scenario

As shown in Figure 4, the experimental setup in this paper uses an RFID reader based on the E710 four-channel development board. The reader operates at a frequency of 920 MHz, with communication interfaces including RS232, TCP/IP, and GPIO. The reader is equipped with a circularly polarized antenna, which has a gain of 9 dB. The tags used are of the Ucode8 chip type, with an operating frequency range of 860–960 MHz and dimensions of 85.5×54×0.8 mm.

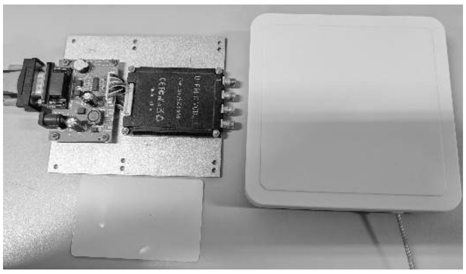


Figure 4: Laboratory Equipment Diagram

As shown in Figure 5, the experimental scenario in this paper involves establishing an XOY coordinate system measuring 1.4m × 1m. The experimental area is divided into multiple square grids with a specification of 0.2m × 0.2m. The origin O of the coordinate system is the antenna location, and the vertices of the square grids represent sampling positions. Test points can be located at any position within the coordinate system.

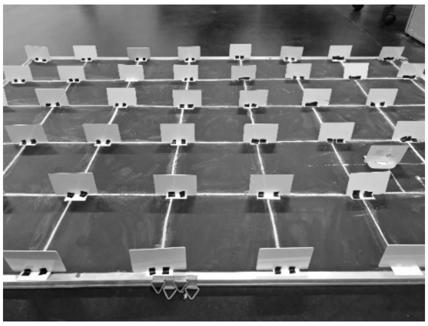
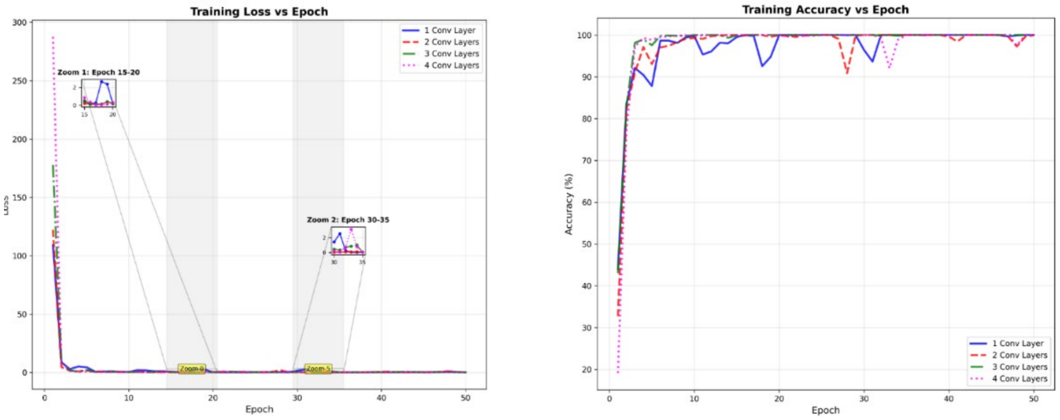


Figure 5: Experimental Scenario Diagram

3.2. Experimental Results and Analysis

Figure 6 illustrates the training loss and accuracy under the RSSI and Phase datasets for different convolutional layers. As shown in (a), the three-layer network demonstrates the most stable convergence characteristics during the 15th to 20th and 30th to 35th iterations, while other networks exhibit significant fluctuations. As shown in (b), the three-layer network achieves an accuracy of over 80% in the early training stages and stabilizes after the 10th epoch, whereas the accuracy of other networks continues to show noticeable fluctuations beyond the 10th epoch.

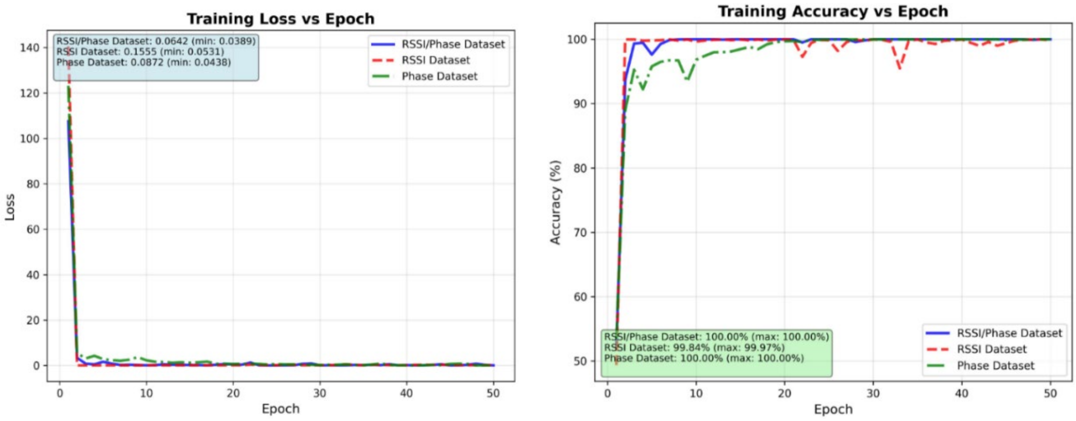


(a)Training Loss Graphs   *(b)Training Accuracy Graphs*

Figure 6: Training Loss and Training Accuracy Graphs for Different Convolutional Layers

Based on comprehensive analysis, the three-layer network demonstrates the best performance in terms of feature representation, training efficiency, stability, and generalization capability. It effectively avoids the limited expressive power of a single-layer network and mitigates the training instability issues encountered in four-layer networks.

Figure 7 presents the training loss and training accuracy for different datasets in a three-layer network. As shown in (a), the average training loss of the RSSI/Phase dataset is lower than that of using RSSI or Phase datasets alone, indicating that the model learns more effectively from the combined RSSI/Phase dataset. As shown in (b), the training accuracy of the RSSI/Phase dataset reaches its peak earlier and remains stable throughout the training process without significant fluctuations. This demonstrates that RSSI and Phase data effectively complement each other, providing a more comprehensive representation of signal characteristics.



(a)Training Loss Graphs *(b)Training Accuracy Graphs*

Figure 7: Training Loss and Training Accuracy Graphs for Different Datasets in a Three-Layer Convolutional Network

In this experiment, 10 sets of data were collected at the test points for coordinate prediction. The localization error of the test point coordinates was calculated using the improved CNN and weighted KNN algorithm, and a comparison was made with the localization errors calculated by other algorithms. As shown in Figure 8, the localization errors of four different methods at the test points are compared. Table 1 presents the localization accuracy values of the different algorithms.

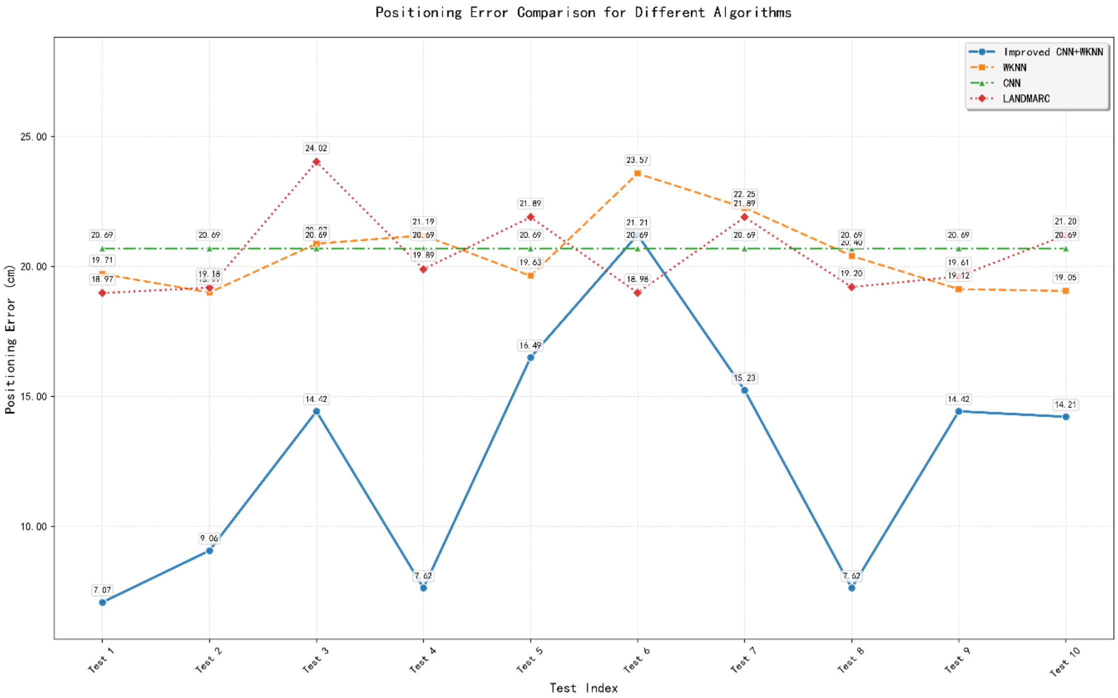


Figure 8: Positioning Error Comparison Chart of Four Different Methods at Test Points

Table 1: Positioning Performance Comparison of Different Algorithms

Localization algorithm	Minimum error/cm	Maximum error/cm	Average error/cm
Improved CNN+WKNN	7.06	21.11	12.75
WKNN	18.99	23.57	20.48
CNN	20.69	20.69	20.69
LANDMARC	18.97	24.02	20.48

As shown in Figure 8, the localization error of Improved CNN+WKNN is generally lower than that of the other algorithms, with the exception of Test6 , where it exceeds the error of WKNN. From the comparison of localization performance among different algorithms in Table 1, it can be seen that the proposed improved CNN and weighted KNN-based RFID localization method can maintain errors at around 12 cm. In contrast, the error ranges for WKNN and CNN are between 18 and 22 cm, while LANDMARC can reach a maximum error of 24.02 cm. It is evident that the RFID localization method combining improved CNN and weighted KNN achieves a significant improvement in localization accuracy, and the experiments validate the effectiveness of the proposed improvement.

4. Conclusions

In the process of indoor multi-label localization, the accuracy of RFID positioning is affected by on-site environmental factors. Therefore, an improved RFID indoor localization method combining CNN and WKNN is proposed to mitigate the impact of noise. The fingerprint features of this method integrate both RSSI and Phase data. By leveraging a convolutional neural network to learn the characteristics of the data, interference from noise and environmental variations can be reduced. WKNN is used for coordinate prediction based on the extracted features, and GA is integrated to enhance localization accuracy. Comparative experiments demonstrate that this method improves the precision of RFID indoor localization. In future work, further consideration will be given to incorporating different environmental interferences, as well as exploring localization research with antennas moving at a constant speed.

References

[1] Wu, L., Ren, J., Li, Y., et al. RFID data-driven product processing cycle prediction for flexible manufacturing workshops[J]. Mechanical Design and Manufacturing, 2025, (01): 265-271.

[2] Wang, S., Shi, C., Zeng, D. Y., & Zou, Y. S. Research on real-time positioning method of AGV based on low-cost IMU and RFID technology. Mechanical Design and Manufacture, 2021,(05): 269-272.

[3] Y. Shao and Q. Xiang, "Intelligent Vehicle Positioning Method Based on UWB and GPS Fusion," 2024 4th International Conference on Electronic Information Engineering and Computer Communication (EIECC), Wuhan, China, 2024: 447-450.

[4] Z. Turgut and A. G. Kakisim, "An explainable hybrid deep learning architecture for WIFI-based indoor localization in Internet of Things environment", Future Gener Comput Syst,2024,7:50-63.

[5] M. D. Jovanovic and S. M. Djosic, "Analysis of Indoor Localization Techniques", Proc. ICEST, 2023:219-222.

[6] S. Böller, A. Bödder, J. Riegler, "SHF RFID based Localization: Combining Time-of-Flight and Angle-of-Arrival," 2024 IEEE International Conference on RFID Technology and Applications (RFID-TA), Daytona Beach, FL, USA, 2024:70-73.

[7] R. Shu, J. Feng, Y. Xu, "Indoor Following Model Based on TDOA Positioning and Q-Learning Algorithm," 2023 2nd International Conference on Automation, Robotics and Computer Engineering (ICARCE), Wuhan, China, 2023: 1-5.

[8] Shang, L., Guan, W., & Gong, R. An improved localization algorithm based on clustering optimization and adaptive KNN[J]. Transducer and Microsystem Technologies, 2023, 42(3): 136-139.

[9] Y. Xie, Z. Wang, and S. Zheng, ''On RFID positioning base on LANDMARC and improved algorithm,'' in Proc. Chin. Control Conf., Beijing, China, 2010:4831–4836.

[10] C. Peng, H. Jiang and L. Qu, "Deep Convolutional Neural Network for Passive RFID Tag Localization Via Joint RSSI and PDOA Fingerprint Features," in IEEE Access, 2021,9: 15441-15451.