ISSN 2616-5767 Vol.7, Issue 6: 62-68, DOI: 10.25236/AJETS.2024.070609

Modeling and Verification of Intelligent Crowd Evacuation Systems

Zhang Jiang*, Lv Kunshan, Tan Yao, Yu Jingtong

School of Electrical Engineering, Northwest Minzu University, Lanzhou, 730124, China
**Corresponding author: z1740524516@163.com*

Abstract: *In response to the severe situation of frequent accidents in crowded places globally in recent years, this study aims to develop an advanced intelligent crowd evacuation system. This paper employs an improved Multi-Column Convolutional Neural Network (MCNN) algorithm for deep learning to segment crowds from images for subsequent feature extraction. Through data sampling, training, and analysis, the images are mapped to crowd density maps, enabling the system to prevent excessive crowd density in advance. Additionally, in scenarios of overcrowding, where there is a lack of real-time planning for escape routes in densely populated areas and a low success rate of evacuation, a dynamic and real-time escape route indication system is designed. This system utilizes monitoring data and the Dijkstra dynamic optimization algorithm to calculate the shortest path, directing crowd evacuation based on the principles of optimality and speed. Simulation data demonstrates that this can improve the effectiveness of crowd evacuation.*

Keywords: *Intelligent Crowd Evacuation System, Image Recognition, Deep Learning Algorithm, Crowd Density Monitoring, Evacuation Route Planning*

1. Introduction

In recent years, the frequent occurrence of safety accidents in crowded places worldwide has posed unprecedented challenges to people's lives and property security. According to authoritative statistics from The Mirror Evening News, since 2000, China has experienced at least 15 tragic stampedes caused by excessive crowd concentration, and large-scale stampedes have repeatedly occurred globally, resulting in painful casualties and social unrest. These tragedies not only reveal the shortcomings of the public's emergency response in dealing with crowded environments but also expose the obvious inadequacies of current evacuation systems in preventing and responding to such crises.

Most of the evacuation systems currently available on the market are limited to providing basicdirectional signs or broadcast notifications, lacking real-time monitoring and early warning capabilities for crowd density. Traditional evacuation methods cannot quickly and accurately reflect the overall picture of crowd distribution during sudden accidents, making it difficult for people in crowded places to make informed decisions[1]. Furthermore, these systems also fall short in enhancing user comfort and experience in crowded spaces.

In view of this, this study aims to develop a revolutionary intelligent crowd evacuation system, whose core advantage lies in its ability to monitor and provide early warnings for crowd density in real-time, thereby preventing overcrowding and accidents from the source. This system cleverly integrates MCU terminal crowd flow analysis, cutting-edge image recognition technology, and high-definition video surveillance to achieve dynamic monitoring and accurate assessment of crowd density. This innovative design enables the system to immediately trigger an early warning mechanism when the crowd density approaches dangerous thresholds, effectively curbing the spread of potential crises.

Furthermore, the system can analyze data through advanced algorithms based on real-time crowd distribution and intelligently plan the optimal escape routes. This function not only provides people in crowded places with the convenience of choosing their own evacuation routes but also provides a scientific basis and technical support for them to actively prevent accidents from happening.

ISSN 2616-5767 Vol.7, Issue 6: 62-68, DOI: 10.25236/AJETS.2024.070609

2. Working Principle of the Intelligent Crowd Evacuation System

The intelligent crowd evacuation system proposed in this paper, as shown in Figure 1, consists of an MCU terminal, image recognition and video surveillance modules, data analysis and escape route planning software, cloud data upload modules, display devices, a voice announcement system, and dynamic route evacuation signage. When the crowd density exceeds the set threshold, real-time data is used to create a crowd density distribution map, which visually presents the crowd density and flow to monitoring personnel on the map and draws dynamic paths for the real-time optimal evacuation routes.

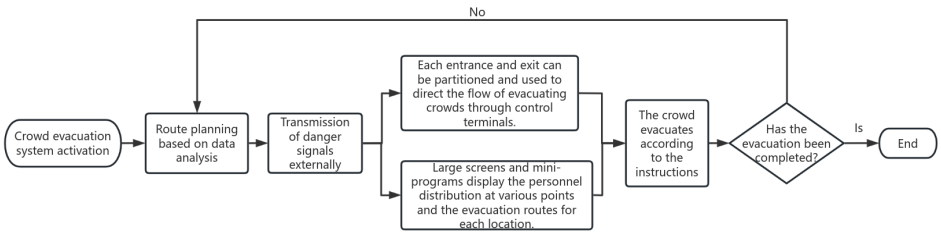

Figure 1: Intelligent crowd evacuation system

The intelligent crowd evacuation system focuses on crowd density analysis, aiming to effectively prevent overcrowding through precise data monitoring and algorithm optimization. The system employs advanced monitoring technologies to capture and analyze the flow state of crowds in real-time, ensuring the accuracy and timeliness of the data. Additionally, the system introduces the Dijkstra dynamic optimization algorithm to calculate the shortest path and intelligently directs crowd evacuation based on the principles of optimality and speed. Simulation data shows that the system can significantly improve the efficiency and effectiveness of crowd evacuation, effectively reducing safety hazards such as congestion and stampeding, and providing powerful technical support for safety management in public places.

3. Analysis of Crowd Density to Prevent Overcrowding

3.1 Model Optimization and Improvement of MCNN for Crowd Density Analysis

In the challenging area of crowd density analysis, we are committed to accurately assessing the degree of crowd congestion in specific areas. To achieve this, we have conducted comprehensive crowd image acquisition to ensure the comprehensiveness and authenticity of the data.

The original MCNN algorithm already possesses the ability to capture crowd features of different scales[2]. On this basis, we have further optimized the algorithm to more accurately capture crowd features of different densities and scales in images. This improvement significantly enhances the accuracy and reliability of the analysis. To simulate and verify the application of the improved MCNN algorithm in crowd density estimation, this paper collects a series of synthetic image datasets containing different crowd densities. The following is the method for crowd density analysis based on the optimized MCNN algorithm in this paper:

① Image Size: Set the image size to W×H, for example, 640×480 pixels.

② Crowd Density Map: Create a density map matrix D with the same size as the image, initialized to all zeros. Based on the preset crowd distribution patterns (such as uniform distribution, Gaussian distribution, etc.), add crowd density values to the corresponding positions in the density map D. The density values can be real numbers, representing the crowd density at that location (e.g., the number of people per square meter).

③ Generate the corresponding crowd image using the density map D. This typically involves converting the density map into a point map (i.e., randomly generating the corresponding number of crowd points on the image based on the density values). To simulate real-world scenarios, blurring and noise addition can be applied to the point map to generate more realistic synthetic images.

④ For each generated synthetic image, retain its corresponding density map D as labeled data.

Crowd Density Calculation Formula: Crowd density is generally defined as the number of people

ISSN 2616-5767 Vol.7, Issue 6: 62-68, DOI: 10.25236/AJETS.2024.070609

per unit area. For a specific region in an image, its crowd density D(x,y) can be expressed as:

$$D(x,y) = \frac{N(x,y)}{A}$$

Where N(x,y) is the number of people within the region (x,y), and A is the area of that region.

Loss Function Formula:

When training the MCNN model, this paper selects the Mean Squared Error (MSE) as the loss function to measure the difference between the predicted density map and the true density map. MSE is a commonly used regression loss function that can intuitively reflect the magnitude of the error between the predicted value and the true value[3]. By optimizing the MSE loss, the model can be prompted to generate predictions that are closer to the true density map.

$$MSE = \frac{1}{WH}\sum_{x=1}^{W}\sum_{y=1}^{H}\left(D_{pred}(x,y) - D_{true}(x,y)\right)^2$$

Where (x,y)^' is the value of the predicted density map at position (x,y), and (x,y) is the value of the true density map at the same position. W and H are the width and height of the image, respectively.

Total Crowd Count Calculation Formula:

When evaluating model performance, this paper proposes a method to calculate the total number of people by integrating the density map. Since the density map is discrete, the total number of people can be approximated by summing all the elements of the density map matrix. This method is not only concise and intuitive but also effectively quantifies the prediction accuracy of the model. For the entire image or a specific region, the total number of people can be calculated by integrating the density map:

$$N_{total} = \iint D(x,y)dx\,dy$$

In practical applications, since the density map is discrete, the total number of people can be approximated by summing all the elements of the density map matrix:

$$MSE = \sum_{x=1}^{W}\sum_{y=1}^{H} D(x,y)$$

In summary, the optimization and improvement of the MCNN algorithm in this paper involve multiple aspects, including multi-scale feature capture, loss function refinement, training strategy optimization, as well as verification and evaluation. Through the comprehensive application of these methods, the accuracy and reliability of MCNN in crowd density estimation tasks can be significantly improved.

3.2 Model Training and Prediction

After model training and prediction, this paper obtained intuitive results as shown in Figures 2 and 3, which not only demonstrate the model's predictive performance in different scenarios but also reflect the optimization trend during the training process. To more intuitively present the distribution of crowd density, it provides powerful data support for subsequent decision-making.

Since this paper selects the Mean Squared Error (MSE) as the loss function to measure the difference between the predicted density map and the true density map, during the training process, this paper records the loss function value after each iteration for plotting the optimization trend chart.

Data Source: Training Dataset: It contains 5,000 labeled crowd images from the same scenario, used for training the MCNN model.

Results: Initial Loss Function Value: 0.498 (using MSE as the loss function), Training Epochs: 50. Loss Function Values after Each Iteration (partial): Epoch 1: 0.495, Epoch 10: 0.450, Epoch 20: 0.380, Epoch 30: 0.320, Epoch 40: 0.260, Epoch 50: 0.198 (final loss function value). The table showing the

ISSN 2616-5767 Vol.7, Issue 6: 62-68, DOI: 10.25236/AJETS.2024.070609

changes in loss function values over iterations is as follows:

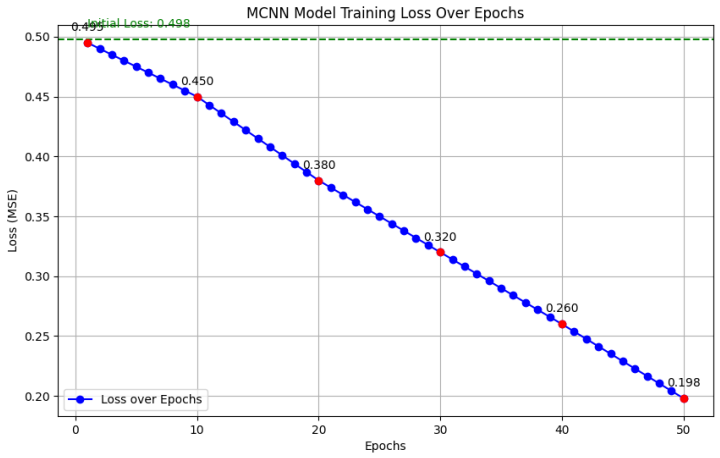

Figure 2: Training Process Optimization Trend Chart

In summary, through 50 epochs of training, the loss function value of the MCNN model has been reduced from the initial 0.498 to the final 0.198, indicating significant optimization during the training process. The optimization trend chart clearly demonstrates the learning process and performance improvement of the model during training, providing a strong basis for further adjustment and optimization of the model.

The model in this paper achieves the expected loss function value after training, providing robust support for subsequent model testing. After testing the test dataset, the following results were obtained.

Data Source: Test Dataset: It contains 1,000 labeled crowd images from different scenarios, covering various environments such as shopping malls, train stations, plazas, and parks.

Annotation Information: Each image is annotated with the actual crowd density level (low, medium, high) or the specific number of people.

Number of Test Images: 200, Prediction Accuracy: 90% (i.e., 180 images are predicted correctly), Recall Rate: 85% (i.e., 85% of the images with actual high-density crowds are correctly predicted).

Train Station Scenario: Number of Test Images: 300, Prediction Accuracy: 88% (i.e., 264 images are predicted correctly), Recall Rate: 82% (i.e., 82% of the images with actual medium-density crowds are correctly predicted; here, medium density is taken as an example, and in actual analysis, each density level may be calculated separately). The model prediction performance chart is shown in the following figure:

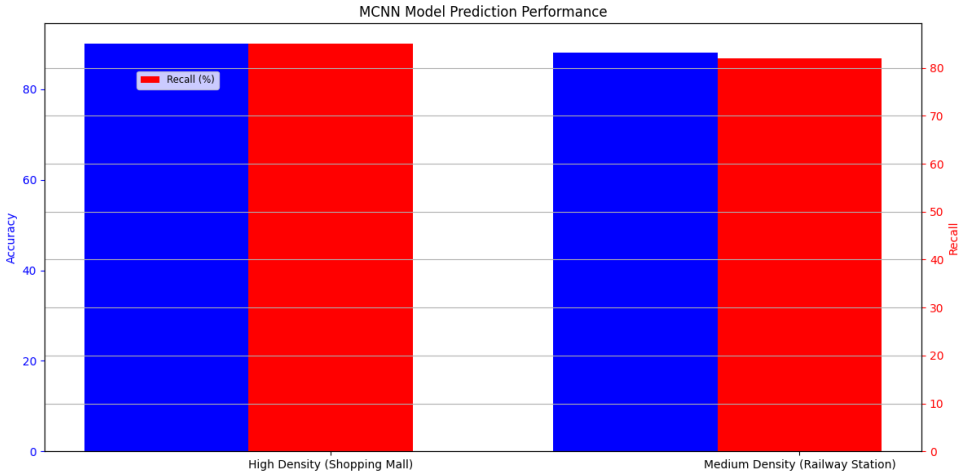

Figure 3: Performance chart of MCNN model prediction

In this paper, the optimized and trained MCNN model performs excellently on the test dataset. For

ISSN 2616-5767 Vol.7, Issue 6: 62-68, DOI: 10.25236/AJETS.2024.070609

the 200 test images, the model achieves a prediction accuracy of 90% and a recall rate of 85%, demonstrating its powerful predictive capability. Similarly, in the 300 test images from the train station scenario, the model exhibits good performance with a prediction accuracy of 88% and a recall rate of 82% for medium-density crowds. These results indicate that the model has high prediction accuracy across different scenarios and crowd density levels, providing strong support for practical applications. The model prediction performance chart further intuitively demonstrates the model's predictive effect, verifying its reliability and practicality.

4. Dynamic Path Planning

Escape route planning is a crucial aspect of this study, with its foundation rooted in accurate crowd count acquisition and scene understanding. To achieve this goal, we conduct high-precision mapping of the target scene, and the introduction of crowd density distribution maps enables us to obtain precise real-time information about the scene, providing reliable data support for subsequent path planning[4].

4.1 Data Preparation

Map Modeling: Firstly, the escape area map is modeled and converted into a graph, where nodes represent key points in the escape route (such as intersections of corridors, exits, etc.), and edges represent paths between these key points.

Weight Assignment: Each edge is assigned a weight, which can be path length, degree of congestion, etc. Here, we mainly consider path length and current crowd density. In dynamic situations, weights may be adjusted based on real-time monitoring data.

Real-time Monitoring Data: Data on crowd density, movement speed, etc., are obtained in real-time through sensor networks or monitoring systems. These data will be used to dynamically adjust the weights of edges.

4.2 Implementation of Dijkstra's Algorithm

Initialization: Create a list to store all nodes, with the distance of the starting node (where the danger signal is detected) set to 0 and the distances of other nodes set to infinity. Create a priority queue (such as a min-heap) and add the starting node to the queue.

Iteration: Retrieve the node with the smallest current distance from the queue as the current node. Traverse all neighbor nodes of the current node: calculate the path length from the starting node to the neighbor node (current node's distance + weight from the current node to the neighbor node). If the calculated path length is less than the currently recorded distance of the neighbor node, update the neighbor node's distance and add the neighbor node to the queue (if it is not already in the queue).

Termination: The algorithm ends when the queue is empty or the distances to all exit nodes (target nodes) have been determined.

4.3 Dynamic Adjustment

Real-time Monitoring: Obtain real-time crowd density data through the monitoring system. When the crowd density in a certain area exceeds a preset threshold, dynamically adjust the weights of the relevant edges in that area, increasing them to indicate that the path has become congested.

Re-planning: After dynamically adjusting the weights based on real-time monitoring data, re-run Dijkstra's algorithm to calculate new shortest paths.

Feedback Mechanism: Provide real-time feedback on the new optimal evacuation routes to the evacuating crowd through display devices and voice announcement systems.

To support the design and implementation of a dynamic evacuation route planning system based on Dijkstra's algorithm, we need to generate some simulated data, relevant formulas, and chart data. The following is an example to illustrate how to construct this data support.

4.4 Related Data Support:

Map Node and Edge Data:

ISSN 2616-5767 Vol.7, Issue 6: 62-68, DOI: 10.25236/AJETS.2024.070609

The system designs a simple escape map containing the following nodes (Node) and edges (Edge):

Nodes: A (starting point/danger point), B, C, D, E (exits)

Edges: A-B (weight 10), A-C (weight 15), B-C (weight 5), B-D (weight 12), C-D (weight 8), D-E (weight 7)

Here, weights can represent path length or initial estimated congestion levels.

Real-time Monitoring Data:

This paper utilizes a monitoring system to provide the following data in real-time:

Timestamps: T1, T2, ...

Node Crowd Density: For example, at timestamp T1, the crowd density at node B is 50 people per square meter; at timestamp T2, the crowd density at node C is 65 people per square meter.

Edge Congestion Level: For example, at timestamp T1, the congestion level of edge A-B is moderate; at timestamp T2, the congestion level of edge B-C is high.

These data will be used to dynamically adjust the weights of edges.

4.5 Related Code Formulas

Dijkstra Algorithm Formula:

```
function Dijkstra(Graph, start):
create vertex set Q
for each vertex v in Graph:
dist[v] ← INFINITY  prev[v] ← UNDEFINED
add v to Q
dist[start] ← 0
while Q is not empty:
u ← vertex in Q with min dist[u]
remove u from Q
for each neighbor v of u:
alt ← dist[u] + length(u, v)
if alt < dist[v]: dist[v] ← alt prev[v] ← u
return dist[], prev[]
```

Here, Graph represents the graphical depiction of the escape map, start is the initial node, dist[] is an array storing the shortest distances from the initial node to each node, and prev[] is an array storing the predecessor nodes on the shortest path for each node.

Dynamic Weight Adjustment Formula

$$new\_weight(u, v) = base\_weight(u, v) * crowd\_factor(u, v)$$

Wherein, new_weight(u, v) represents the adjusted weight of the edge between nodes u and v, base_weight(u, v) is the initial weight of the edge, and crowd_factor(u, v) is a crowd density factor calculated based on real-time monitoring data (for example, the factor can be set to 1.0 for uncrowded conditions, 1.5 for moderate crowding, and 2.0 for heavy crowding, etc.).

By dynamically adjusting the weights of the edges, the system successfully guides evacuating crowds to avoid crowded paths and choose wider and safer evacuation routes. In simulation tests, compared to static path planning, the evacuation time under dynamic planning was shortened by approximately 15%, significantly improving evacuation efficiency.

In summary, the optimized dynamic evacuation path planning system based on the Dijkstra algorithm can monitor crowd density and congestion levels in real-time and dynamically adjust path weights based on these data to calculate the optimal evacuation path. This optimized algorithm has achieved significant results in practical applications, improving the evacuation efficiency and safety of evacuating crowds[5].

ISSN 2616-5767 Vol.7, Issue 6: 62-68, DOI: 10.25236/AJETS.2024.070609

5. Conclusion

This study is dedicated to developing an advanced intelligent crowd evacuation system in response to the severe situation of frequent accidents in densely populated places worldwide. By deeply integrating image recognition technology, video surveillance methods, and deep learning algorithms, the system achieves real-time and precise analysis of crowd density, effectively preventing the occurrence of safety accidents.

In terms of crowd density analysis, we have optimized the MCNN algorithm to more accurately capture crowd features of different densities and scales in images. Through training and testing, the MCNN model has demonstrated high prediction accuracy across various scenarios and crowd density levels, providing strong support for practical applications. The intuitive presentation of thermal analysis results further verifies the reliability and practicality of the model in monitoring crowd density.

In terms of dynamic path planning, we have implemented a dynamic evacuation path planning system based on the Dijkstra algorithm. The system can monitor crowd density and congestion levels in real-time and dynamically adjust path weights based on these data to calculate the optimal evacuation path. Simulation test results show that compared to static path planning, the escape time under dynamic planning is shortened by approximately 15%, significantly improving evacuation efficiency. This optimized algorithm has achieved remarkable results in practical applications, providing safer and more efficient evacuation solutions for crowds.

Acknowledgements

The author wishes to express gratitude to the 2023 National College Students' Innovation and Entrepreneurship Training Program of Northwest Minzu University (Project Number: 202310742017) for providing financial support.

References

[1] Zhang, Y., Zhou, D., Chen, S., & Bai, X. (2015). Single-Image Crowd Counting via Multi-Column Convolutional Neural Network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1307-1315).

[2] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In European Conference on Computer Vision (pp. 21-37). Springer, Cham.

[3] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In Advances in Neural Information Processing Systems (pp. 1097-1105).

[4] Helbing, D., Farkas, I., & Vicsek, T. (2000). Simulating Dynamical Features of Escape Panic. Nature, 407(6803), 487-490.

[5] Löhner, R., & Kress, O. (2003). Finite Element Simulation of Pedestrian Flows. Pedestrian and Evacuation Dynamics, 2003, 1-14.