Intelligent Classification Model for Interior Design Knowledge Graph based on Simulated Annealing Algorithm

Jie Liu[1], Feng Wang[2*], Bin Song[2], Xiangyun Wang[2]
[1]Faculty of Mathematics, Qilu Normal University, Jinan 250200, China
[2]School of Art Design, Shandong Youth University of Political Science, Jinan 250103, China
Email: ljqlnu1018@163.com
[*]Corresponding author

Keywords: simulated annealing algorithm, knowledge graph, genetic algorithm, interior design, classification models

Received: April 16, 2024

In real life, interior design is a complex and challenging job. Interior design solutions need to consider factors such as spatial layout, color matching, etc., and the emergence of knowledge graph provides a new method of summarizing design ideas for the interior design industry. However, in the face of a large number of knowledge graphs, how to achieve high-quality classification of knowledge graphs has become a hot topic of discussion in related industries. The work builds a knowledge graph intelligent classification model based on machine learning, simulated annealing, and genetic algorithms to accomplish effective knowledge graph classification. The global optimization of convolutional neural network parameters is accomplished by merging the model using the simulated annealing approach and the genetic algorithm. The experimental results indicated that the proposed model converged to an F1 score of about 95.03%, while the control model converged to an average F1 score of 94.37% and 94.26%. The average recall of the proposed model was 91.71% while the average recall of the control model was 87.06%. Based on the experimental findings, it can be said that the suggested model performs noticeably better than the control model, indicating that it is an improved knowledge graph classification method. In addition, the proposed model contributes to the development of interior design related industries.

Povzetek: V članku je opisan razvoj inteligentnega modela za klasifikacijo grafov znanja na področju notranjega oblikovanja. Predlagan model temelji na algoritmu simuliranega ohlajanja in dosega boljšo točnost ter izboljšano klasifikacijo.

1 Introduction

Interior design (InD) is a comprehensive field, which involves aesthetics, architecture and so on. Currently there are numerous InD styles of various types, such as minimalism, industrial style, Scandinavian style, etc. Each of these design styles has its own characteristics, which brings a wealth of choices for modern home life [1]. However, with the continuous development of the design field, the diversity and innovation of InD have become increasingly prominent. The rich and diverse design styles and creativity also create a considerable workload for InD solution summarization. But as different machine learning (ML) algorithms have evolved, the issue of InD solution summarization has been resolved with the introduction of knowledge graphs (KG). The term “knowledge domain visualization,” also known as “knowledge domain mapping map,” refers to a collection of several graphs that illustrate the structural links and knowledge production process [2-3]. Through KG, the association between design styles, design solutions and design objects can be visualized, thus assisting the planning and program construction of InD. KGs are usually constructed in a structured way, but the construction of KGs through this way will output a large number of KGs [4]. How to effectively categorize the generated KGs has become a difficult problem in KG research. Currently available optimization techniques include the genetic algorithm (GA) and the simulated annealing algorithm (SAA), which when combined with ML algorithms can increase the ML algorithm's convergence speed and computational efficiency [5]. In view of this, the study employs GA and SAA for combining and constructing a fusion optimization seeking algorithm to help convolutional neural networks (CNN) for KG classification.

To address the problem of KG generation, the study found through reviewing the literature that a number of researchers have innovated the methods of KG generation and application methods. For example, Xue et al. found that the current mainstream KG still contains inaccurate or outdated entries, so they proposed a KG construction method that can be used for quality assessment and error detection. In comparison to existing models of the same kind in the KG creation process, the suggested method had superior accuracy and sophistication, as confirmed by experimental verification in the final study [6]. Wang and other researchers used heterogeneous graphs to improve the KG generation process, and the KGs after the introduction of heterogeneous graphs can be represented in a low dimensional space. In addition, the study also analyzed the applicability of heterogeneous graph-based KGs in real industrial environments and concluded that the proposed KG construction method has achieved some success in real application scenarios [7]. Yuan et al. constructed an exponential atlas generation model based on depth model using ML method, which achieves more accurate data keyword extraction by graph convolutional network for data extraction and fusion. It was experimentally proved that the atlas generated by the proposed KG generation method outperforms other similar models in terms of keyword summarization and keyword relationship processing, and the results show the advanced nature of the proposed model [8]. Steenwinckel et al. tried to add ML algorithm into the process of KG generation. The relationship between the data was characterized by the instance neighborhood of the knowledge in the deep learning-based KG construction, while the representation of the nodes of interest in the KG was in a binary way for easy processing by the ML algorithm. The final experiment indicated that the proposed new KG construction method has higher efficiency and accuracy compared to other methods [9].

Numerous research teams have also made innovations in the optimization and implementation of SAA. For example, Li et al. suggested a broadband passive interference technique based on SAA to address the technological challenges associated with broadband passive interference. Using the least amount of each chosen sub-element, the method employs SAA to produce the newly integrated foil element's amplitude-frequency profile with the least amount of variation. Experimental results showed that the new integrated foil element optimized with SAA has high interference efficiency [10]. A new path planning method for robotic systems was proposed by researchers such as Shi to address the logical rationality of robot trajectories and final states. In this path planning method, the SAA was employed for the optimization of each robot's path trajectory, thereby enabling the derivation of the optimal path solution for each robot within the current region. The study's findings indicate that, in terms of both computing cost and the quality of the solutions produced, the strategy proposed in the research performs better than the current strategies in use [11]. Shin et al. found that the traditional SAA has obvious shortcomings in coping with large-scale problems, so they chose to add an efficient storage hardware for memory optimization based on stochastic SAA, and designed relevant experiments to verify the optimization effect of the improved SAA on memory. The experimental results demonstrated that the proposed method achieved an absolute advantage in the maximum cut combination optimization problem [12]. Cloud cover can provide a significant challenge to microcosmic data transmission since optical remote sensing sensors are unable to detect through clouds. In view of this, Han et al. introduced SAA to optimize the cloud coverage uncertainty. The final results demonstrated that the proposed technique represents a significant advancement in AEOS scheduling, outperforming existing state-of-the-art methods in various scenarios [13].

The summary of relevant work is shown in Table 1.

Table 1: Summary of related work

Research topic	Research methods and applications	Main indicators	Performance results	Limitations
Knowledge graph	Xue et al.'s knowledge graph construction method [6]	Quality assessment and error detection	Improve the accuracy and complexity of knowledge graphs	There may still be inaccurate or outdated entries
	Wang et al.'s heterogeneous graph method [7]	Low dimensional space representation and practical industrial applicability	Improve the practicality and applicability of knowledge graphs	Further verification is needed for its wide applicability to actual industrial environments
	Yuan et al.'s deep model-based exponential graph generation model [8]	Data keyword extraction and keyword summary	Superior keyword processing and summarization performance compared to other models	The complexity of deep models may increase computational costs
	Steenwinckel et al.'s deep learning method [9]	Efficiency and accuracy	Higher efficiency and accuracy	There may be a high computational complexity issue in constructing large-scale knowledge graphs

Simulated annealing algorithm	Application of Li et al.'s simulated annealing algorithm in broadband passive interference technology [10]	Interference efficiency	Improving the interference efficiency of broadband passive interference technology	Further validation is needed to evaluate the effectiveness in different environments
	Shi et al.'s path planning method based on simulated annealing algorithm [11]	Optimal path solution and cost calculation	Better path planning effect	There may be challenges in real-time path planning for complex environments
	Shin et al.'s memory optimization simulated annealing algorithm [12]	Memory optimization effect	Maximum cut combination optimization solution with absolute Advantage	Need to consider applicability in different hardware environments
	Han et al.'s simulated annealing algorithm for optimizing uncertainty in cloud coverage [13]	Optimize cloud coverage and improve performance	Better performance than current methods	Performance improvement only under specific tasks, generalization needs further verification

In conclusion, existing research has made notable advancements in knowledge graph generation and SSA optimization, with a particular focus on enhancing accuracy and efficiency. Nevertheless, existing methodologies present shortcomings, including suboptimal accuracy of KG and diminished algorithmic efficacy. The fusion optimization search algorithm proposed in the study combines a GA and an SSA, which can effectively address the classification problem in the process of knowledge graph generation, thereby improving accuracy and efficiency. This method can overcome the inaccuracies and obsolescence inherent in KG, while also exhibiting high algorithm convergence speed and computational efficiency. It thus represents a novel solution for the construction of KG in the field of InD.

The paper is organized primarily into four sections. The first section is the introduction, which introduces the current research status of the technology used. The approach, which carries out the intelligent categorization and creation of the KG of InD by SAA and GA, is covered in the second section. The third section is the model performance test, which verifies the advancement of the proposed model by designing controlled experiments. The fourth section is the discussion section, which mainly compares and analyzes the performance results of the proposed model. The last section is the conclusion, which mainly summarizes the research results and shortcomings.

2 Methods and materials

This subsection explores the keyword-based generation of KG for InD and its classification method. In order to realize the classification problem of KG of InD, the study introduces SAA and GA for ICM construction, and GA is utilized to improve SAA in order to solve the local convergence problem of SAA so as to improve the classification effect of ICM.

2.1 Intelligent classification model construction based on SAA

To carry out the intelligent classification of KG for InD, the study adopts a structured approach for KG construction with InD as the main body. The structured KG construction includes four steps: data entity extraction, data fusion, data constraints, and KG output [14]. Data entity extraction is to construct the relationship between InD data into a ternary group, which consists of a predicate and two formal parameters, and the data in the ternary group determine the existence of the relationship through the formal parameters. The study obtains the triad of data relationships before data fusion [15]. The feature data of different InDs exists in the form of triples, so the fusion process only needs to perform the merging of like terms of the triples. The data constraint process mainly carries out the normalization calculation and de-weighting of the data, and the constrained data is persisted into the KG through data persistence. Figure 1 depicts the precise flow of KG construction.

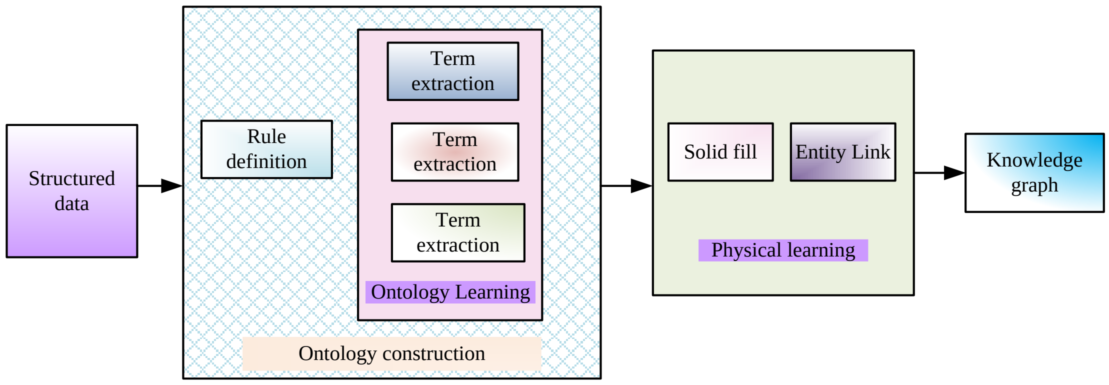


Figure 1: Schematic diagram of knowledge graph construction

After constructing the KG generation framework, iterative updating is also required to get the complete KG. In structural optimization methods, the iterative update of KG is usually carried out by incremental update [16]. Incremental updating can update only the changed data, so this updating method plays an important role in saving system resources and time. Although the traditional KG construction has formed a more complete system, the KG constructed by the above method still has obvious drawbacks, such as long construction period, low accuracy of data relationship processing, etc. In view of the above drawbacks of KG construction, the study adopts keyword entity extraction instead of traditional data entity extraction. In order to accurately extract keywords from data, the research also needs to use ML algorithms for assistance. By investigating and analyzing the application effects of current common ML algorithms, the study selects the simple, fast, and easy-to-understand term frequency-inverse document frequency (TF-IDF) method to extract keywords from InD data. This method mainly evaluates the importance of keywords in the input dataset by calculating term frequency (TF) and inverse document frequency (IDF) of the input InD data [17]. Where the expression for TF calculation is shown in Equation (1).

$$TF = \frac{f_{i,j}}{\sum_k f_{f,j}} \tag{1}$$

In Equation (1), $f_{i,j}$ denotes the times a word appears in the input data set. tf denotes the meaning of the proportion of a word in the input text. In addition, the expression for calculating IDF is shown in Equation (2).

$$IDF = \lg^{\frac{|D|}{|1+\{j:w_i \in d_j\}|}} \tag{2}$$

In Equation (2), D is the total number of keywords in the dataset related to the input InD. $\{j : w_i \in d_j\}$ is the occurrences of a word w_i among all the keywords. d_j denotes all words in the input dataset. The final output of the keywords can be obtained by synthesizing the output of TF and IDF, and the expression of the synthesized output is shown in Equation (3).

$$\begin{aligned} & TF(w_i) * IDF(w_i) \\ & = TF - IDF(w_i) = \frac{f_{i,j}}{\sum_k f_{f,j}} * \lg^{\frac{|D|}{|1+\{j:w_i \in d_j\}|}} \end{aligned} \tag{3}$$

The output result after processing by TF-IDF is a data sequence, and the elements in this data sequence are the keywords of KG [18]. To extract the keywords of the InD data and construct the KG, CNN is selected for the study to perform the keyword relationship extraction, while the powerful classification ability of CNN can also be used for the classification model (CM) construction of the KG. After extracting the relationships between keywords, the generated KG of InD is shown in Figure 2.

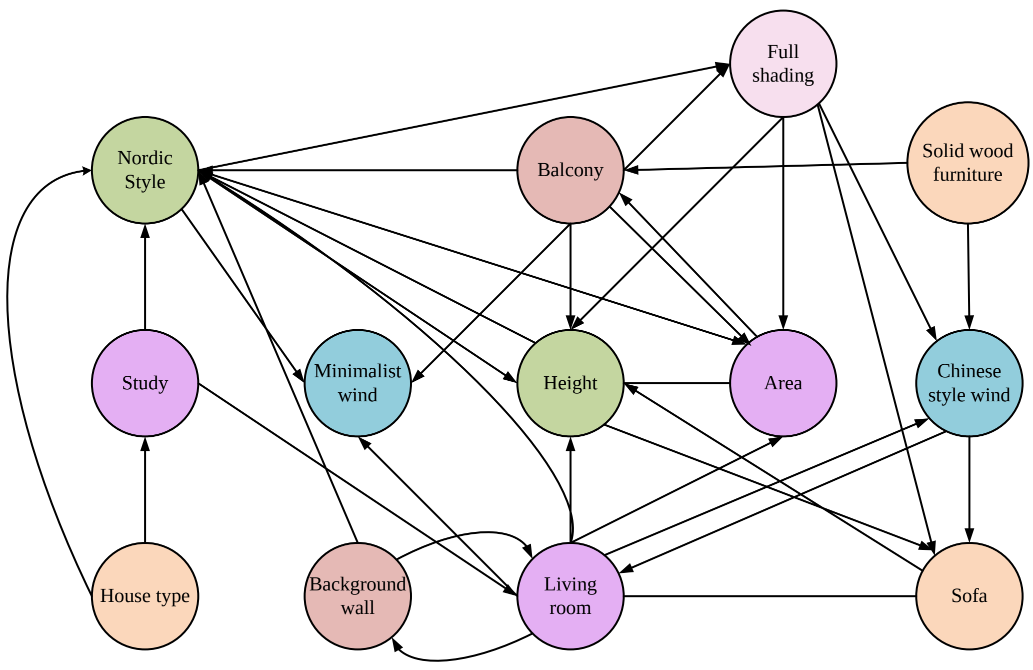


Figure 2: Interior design knowledge graph display

In this study, the CNN algorithm is not only used for the extraction of keyword relationships, but also involved in the classification of KG, but the traditional CNN algorithm also has the obvious defect of the difficulty of convergence of the objective function (OF) [19]. In order to solve this problem, the fully connected layer (FCL) of CNN needs to be made to find the globally optimal parameters during feature computation. Therefore, the study introduces SAA for OF optimization based on the traditional CNN algorithm. The fundamental goal of SAA, a universal optimization technique inspired by the solid matter annealing process, is to simulate the solid annealing process in order to obtain the global optimal solution for the OF [20-21]. The temperature decay function is one of the most important parameters in SAA, and the computational expression of this parameter is shown in Equation (4).

$$T_{k+1} = \alpha T_k \tag{4}$$

In Equation (4), α denotes the temperature decay coefficient also known as the cooling rate parameter, which is used to control the rate of temperature change. T_{k+1} denotes the temperature of the $k+1$th iteration and T_k denotes the temperature of the k th iteration. To adapt SAA to the EucD computation of CNN, the study adds new inversion parameters to the algorithm, and therefore the algorithm's temperature decay function computation is changed accordingly. The new temperature decay function calculation expression of SAA is shown in Equation (5).

$$T(k) = T_0 \cdot EXP\left(-Ck^{\frac{1}{N}}\right) = T_0 \alpha^{K^{\frac{1}{N}}} \tag{5}$$

In Equation (5), k denotes the iterations of the algorithm, which is consistent with the iterations of the CNN algorithm. T_0 denotes the initial temperature set before the start of the iteration, and $EXP(.)$ denotes the exponential operation. C denotes a random constant and N denotes the number of parameters to be inverted. After determining the temperature decay function and the initial temperature, the algorithm needs to generate the initial solution and set up a perturbation mechanism to generate a new solution [22]. The computational expression for the perturbation mechanism is shown in Equation (6).

$$\begin{cases} \tilde{m}_i = m_i + y_i\left(B_i - A_i\right) \\ y_i = T\operatorname{sgn}\left(u - 0.5\right) \cdot \left[\left(1 + \frac{1}{T}\right)^{|2U-1|} - 1\right] \end{cases} \tag{6}$$

In Equation (6), m_i is the i th variable of the solution in the current iteration number. $\tilde{m}_i$ denotes the i th variable of the new solution and T is the temperature in the current iteration number. $|.|$ denotes taking absolute values. u denotes a constant generated by a random number function that takes values in the range [0, 1]. $\operatorname{sgn}(.)$ denotes the sign function. A_i denotes the lower limit of the i th variable and B_i denotes the upper limit of the i th variable. y_i denotes the intermediate variable in the perturbation mechanism. In addition, Metropolis criterion also has an important role in SAA. By Metropolis criterion, it makes the algorithm to receive worse solutions, so the algorithm can better search the global solution [23-24]. The mathematical expression of Metropolis criterion is shown in Equation (7).

$$H = \begin{cases} 1, E_1 > E_2 \\ EXP\left(-\frac{E_2 - E_1}{T}\right), E_1 \le E_2 \end{cases} \quad (7)$$

In Equation (7), E_1 denotes the state energy of the algorithm at the current moment and E_2 denotes the state energy of the algorithm after the state update. The SAA calculation flow is shown in Figure 3.

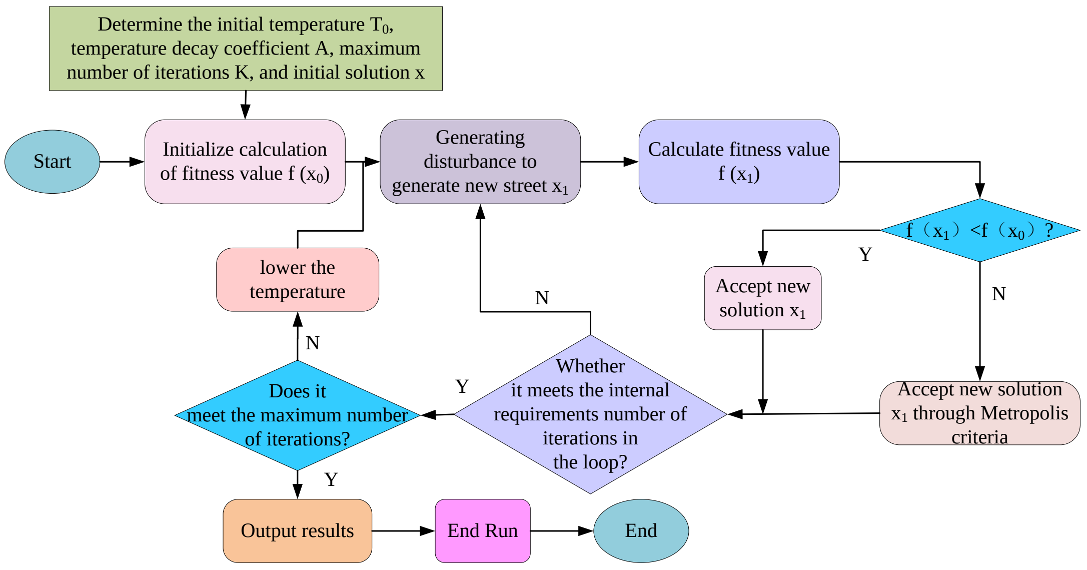

Determine the initial temperature T0, temperature decay coefficient A, maximum number of iterations K, and initial solution x
Start
Initialize calculation of fitness value f (x0)
Generating disturbance to generate new street x1
Calculate fitness value f (x1)
f（x1）<f（x0）?
Y
N
Accept new solution x1
Accept new solution x1 through Metropolis criteria
lower the temperature
N
Whether it meets the internal requirements number of iterations in the loop?
N
Y
Does it meet the maximum number of iterations?
Y
Output results
End Run
End

Figure 3: Simulated annealing algorithm optimization calculation process

Although SAA has the advantages of skipping the local optimal solution and reasonably dealing with the NP problem, the algorithm has obvious shortcomings in practical applications. The cooling rate parameter affects the SAA outcomes, thus if it is set excessively big or little while the algorithm is executing, it may bypass the global optimal solution [25]. Furthermore, the technique necessitates numerous Metropolis criterion operations, resulting in an extended execution time and sluggish convergence during the process. In view of this, although the use of SAA for optimization can help the fully-connected layer of the CNN algorithm to find the optimal parameters, it also leads to a reduction in the overall operating rate of KG's CM due to the addition of SAA, so the study needs to further optimize the model.

2.2 CM construction incorporating GA and SAA

GA is proposed by Prof. Holland of the University of Chicago, the algorithm is a meta-heuristic algorithm. A population in GA is made up of a particular number of chromosomes, and the process of natural selection is the iteration of the OF [26]. The rationale behind selecting GA as the optimization algorithm is based on the characteristics of its metaheuristic algorithm, which is capable of effectively searching the solution space and identifying the global optimal solution. The GA represents individuals in the solution space through chromosomes and simulates the process of biological evolution through natural selection, crossover, and mutation. It exhibits high search efficiency and global optimization capability. The combination of GA with the SSA results in enhanced robustness and adaptability when confronted with complex problems, and facilitates the convergence to the optimal solution in a more expeditious manner.

Each solution in GA can be represented by a chromosome. The method begins by creating a certain number of chromosomes based on a random function; the first chromosome to be created is the initial result provided by the algorithm, which may or may not represent the actual solution to the problem [27]. At the same time after generating the initial solution, the algorithm will also give the corresponding fitness according to the OF and the initial solution, at this time, each chromosome's respective fitness is f_i . The expression for the computation of the fitness in this study is shown in Equation (8).

$$f_i = \begin{cases} C_{\max} - f', f' < C_{\max} \\ 0, else \end{cases} \quad (8)$$

In Equation (8), $C_{\max}$ denotes the maximum value of the fitness value, also known as the fitness threshold. f' denotes the output value. Equation (9), which shows the computation expression for the fitness of the entire population, is the superposition of the fitness of each chromosome.

$$F = \sum_{i=1}^{m} f_i \quad (9)$$

In Equation (9), F denotes the fitness of the population as a whole and m is the all chromosomes in the current population. When the algorithm enters the iterative session the determination mechanism will judge the chromosomes according to the fitness value, and realize the inheritance of high-quality genes by retaining the chromosomes with high fitness and eliminating the chromosomes with low fitness [28]. Figure 4 illustrates the algorithm's precise workings.

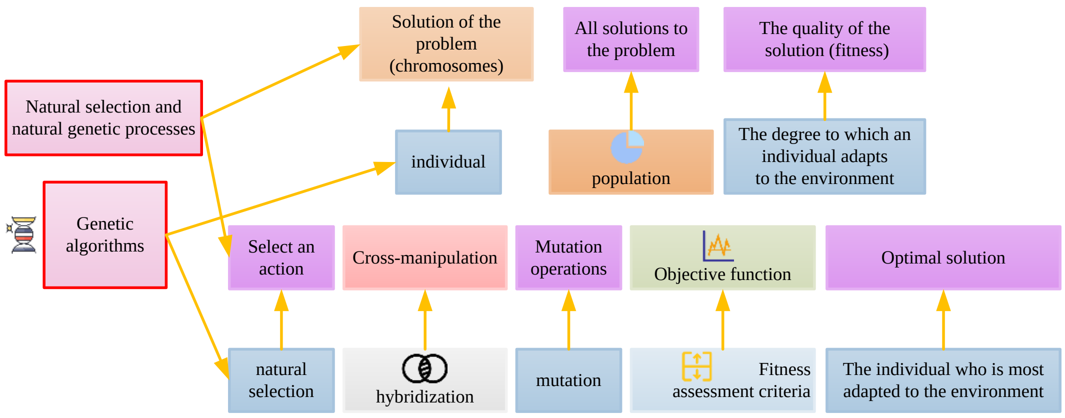

Natural selection and natural genetic processes
Genetic algorithms
Solution of the problem (chromosomes)
individual
All solutions to the problem
population
The quality of the solution (fitness)
The degree to which an individual adapts to the environment
Select an action
Cross-manipulation
Mutation operations
Objective function
Optimal solution
natural selection
hybridization
mutation
Fitness assessment criteria
The individual who is most adapted to the environment

Figure 4: Schematic diagram of the simulation relationship between genetic algorithm and natural selection and genetic processes

The coding of every variable in InD's KG, where the various variables are grouped based on the number of bits they are identified in the binary code, makes up the population's chromosome. The seven-bit binary code denotes the variables A, B, E, and F. The type of variables E is represented by 10-bit binary code. The types of variables C and D are represented by a 13-bit binary code. The collective's chromosomal coding is displayed in Figure 5.

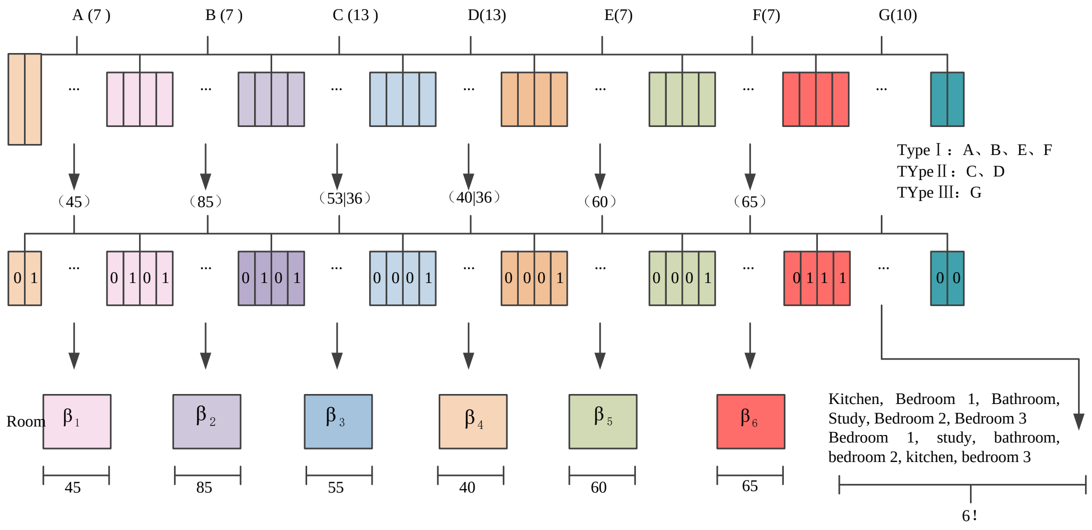


Figure 5: Schematic diagram of chromosome coding details

After encoding is complete, the formal selection phase can begin. During this phase, the likelihood of a certain chromosome being chosen is directly correlated with its fitness—the greater the fitness, the greater the likelihood of selection [29]. The expression for calculating the probability of a chromosome being selected is shown in Equation (10).

$$P_i = \frac{Z_i}{F} = \frac{Z_i}{\sum_{i=1}^{m} f_i} \tag{10}$$

P_i is the likelihood that the i th chromosome will be chosen in Equation (10). The fitness of the i th chromosome is represented by Z_i, while the population's overall fitness is represented by F. In addition, the process of chromosome inheritance is an iterative process, so different chromosomes will be selected or eliminated many times, so the study selects the final cumulative selection probability of each chromosome to judge the superiority of the solution [30]. The expression for calculating the cumulative selection probability is shown in Equation (11).

$$h_i = \sum_{j=1}^{i} P_i \tag{11}$$

In Equation (11), h_i denotes the cumulative selection probability of the i th chromosome. P_i denotes the selection probability of the i th chromosome. The coding process also needs to pay attention to the constraints generated by the initial state of the room, the design style, and the room type in different KGs. The constraints are mainly reflected in the OF of the algorithm, and in the study, the classification of KGs of InD is performed based on the CNN algorithm, so the expression of the classification OF after adding the constraints is shown in Equation (12).

$$w_j = \frac{\partial a_j}{\partial x_j} = \frac{\sum_{j=1}^{N} W * x_j}{\partial x_j} \tag{12}$$

In Equation (12), w_j denotes the OF of the j th iteration. N is the total iterations set, and ∂ is the bias sign. W is the weight matrix of the convolutional layer, and a_j denotes the parameter set of the EucD. To further improve the accuracy of the OF, the study integrates GA and SAA, where the construction of the initial solution in SAA is carried out by encoding and setting the initial population size in GA. Meanwhile the construction process of the perturbation mechanism can also be carried out by chromosome selection and crossover inheritance. The specific fusion scheme is shown in Figure 6.

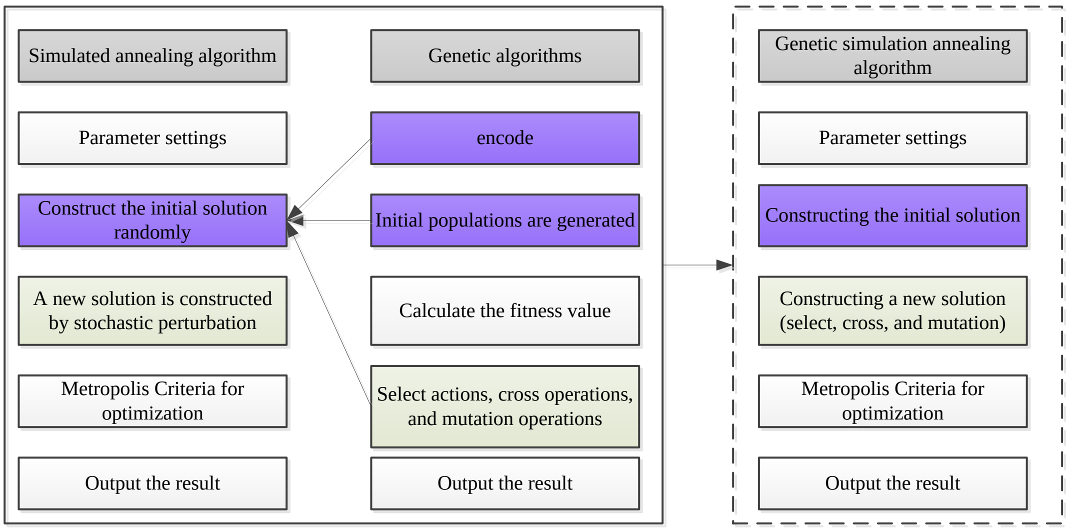
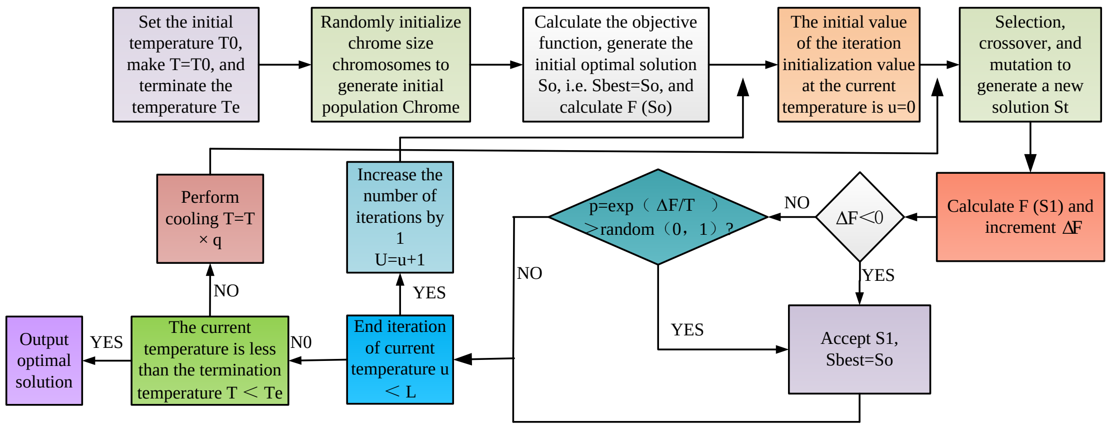

Simulated annealing algorithm
Genetic algorithms
Genetic simulation annealing algorithm
Parameter settings
encode
Parameter settings
Construct the initial solution randomly
Initial populations are generated
Constructing the initial solution
A new solution is constructed by stochastic perturbation
Calculate the fitness value
Constructing a new solution (select, cross, and mutation)
Metropolis Criteria for optimization
Select actions, cross operations, and mutation operations
Metropolis Criteria for optimization
Output the result
Output the result
Output the result

Figure 6: Schematic diagram of the fusion scheme of genetic algorithm and simulated annealing algorithm

Fusion algorithms Following the algorithm's lack of a unified standard of judgment for assessing the output solution's degree of merit, the study employs the Euclidean distance (EucD) between the output solution and the value of the OF for evaluation in order to solve the output solution's merit situation evaluation. Equation (13) displays the expression used to compute the EucD.

$$L_i = \sqrt{\sum_{i=1}^{n} \left(x_i - o_i\right)^2} \tag{13}$$

In Equation (13), o_i is the OF value at the i th iteration. x_i denotes the solution output by the fusion algorithm at the i th iteration and n denotes the total iterations. L_i denotes the EucD at the i th iteration. The study uses the results calculated by Equation (13) in the judgment of the merit of the solution, and the final flowchart of the computation of the fusion optimization algorithm based on GA and SAA is shown in Figure 7.



Set the initial temperature T0, make T=T0, and terminate the temperature Te
Randomly initialize chrome size chromosomes to generate initial population Chrome
Calculate the objective function, generate the initial optimal solution So, i.e. Sbest=So, and calculate F (So)
The initial value of the iteration initialization value at the current temperature is u=0
Selection, crossover, and mutation to generate a new solution St
Calculate F (S1) and increment ΔF
ΔF<0
p=exp（ΔF/T ）>random（0，1）?
Accept S1, Sbest=So
Perform cooling T=T × q
Increase the number of iterations by 1 U=u+1
End iteration of current temperature u < L
The current temperature is less than the termination temperature T < Te
Output optimal solution
YES
NO
N0

Figure 7: Calculation flowchart of simulated annealing algorithm with integrated genetic algorithm

The study uses the above fusion algorithm for parameter optimization of CNN algorithms to achieve ICM construction of KG. The fusion model mainly works on the EucD of the CNN algorithm. In the CNN algorithm, the parameters present in the EucD is high, so it is necessary to introduce the above fusion model for parameter optimization to help the model converge quickly. The output expression of the EucD after the introduction of the fusion model after the CNN model is shown in Equation (14).

$$H\left(x_j\right)=\sum_{j=1}^{N}\frac{\partial H\left(x_{j-1}\right)}{\partial a_j}\square w_j \tag{14}$$

In Equation (14), $H\left(x_j\right)$ denotes the output function of the EucD and ∂ denotes the bias sign. x_j denotes the j th element in the input sequence, a_j denotes the set of parameters in the EucD, and w_j denotes the OF. After fusing the improved optimization algorithm with the CNN EucD, the problems of convergence difficulty and excessive computation of the traditional CNN model are solved. By fusing the optimization algorithm for parameter optimization, the CNN model can quickly determine the computational parameters and reduce unnecessary computations, thus improving the output efficiency of the model. To facilitate the subsequent experiments, the study adopts the AS-GA-CNN model to refer to the CM of the proposed KG.

3 Results

3.1 Experimental environment and parameter settings

The hardware equipment used for the experiment is a computer with Intel Xeon w9-3495X CPU, RTX 2080 Ti graphics card and 16GB cache, and the system environment is Windows 10. Distributed load transfer model is constructed in JAVA language, and the compiler version used for the experiment is JAVA 3.7, and the JDK version is JDK 1.8. performance The control models chosen for the test experiments include CNNCM (K-CNN) improved based on K-means algorithm and CNNCM (A-CNN) optimized based on Ant Colony algorithm. The Scenes dataset is selected as the training dataset for the experiment. This dataset includes RGB-D images, real camera poses, and 3D models of seven indoor rooms. The images exhibit a diverse range of features, including textureless surfaces, motion blur, and repetitive structures, which provide a comprehensive assessment of the model's adaptability to different scenes. The test dataset comprises the Innoc and Aachen Day Night datasets, which exhibit disparate environmental and lighting conditions. These datasets serve to corroborate the model's capacity for generalization. The rationale for selecting these datasets is that they are frequently employed in the domain of InD and are representative and challenging.

Table 2 provides further comprehensive experimental information. In terms of experimental parameter settings, the model's operating parameters have been optimized in order to enable the model to fully learn data features and achieve good performance. The potential for bias may arise from an uneven distribution of samples in the dataset or an incomplete selection of the test dataset, which may affect the generalizability of the research results. Accordingly, these factors are taken into account during the interpretation and inference process of the experimental results.

Table 2: Experimental environment and details of experimental parameters

Experimental environment	Parameter
hardware environment	CPU: Intel Xeon w9-3495X; Graphics card: RTX 2080 Ti; Cache: 16GB
software environment	JAVA 3.7; JDK 1.8
System environment	Windows 10
Training dataset	Scene's dataset
Test dataset	Inloc dataset; Aachen Day Night dataset
Experimental model	AS-GA-CNN model
Comparison model	K-CNN model; A-CNN model
Test indicators	Output delay; Recall rate; Balance rate; Loss rate; F1 score; Error rate; P-R curve
Model running parameters	Learning rate: 0.1; Iterations:100

In Table 2, in addition to the given model operating parameters, other key operating parameters include population size, crossover rate, and mutation rate. In this study, the population size is set to 100, which ensures sufficient search space coverage and controls computational costs. Setting the crossover rate to 0.8 and the mutation rate to 0.1 can ensure the convergence speed and search efficiency of the algorithm while maintaining population diversity. Furthermore, the length of each chromosome is classified according to the type of variable, which are 7, 10, and 13 positions. This classification takes into account the characteristics of different variables in the knowledge graph, thereby making chromosome coding more in line with the actual situation of the problem. The selection of these parameters is intended to ensure the effectiveness of the algorithm while maximizing search efficiency and convergence speed.

3.2 CM Performance testing incorporating SAA and GA

In Figure 8, the study compares the equilibrium rates of the solutions obtained by the GA, SAA and fusion algorithms using the Aachen Day-Night dataset (ADND) as input to the model. The mean value of the equilibrium rate obtained by the AS-GA on the ADND is significantly larger than the mean value of the equilibrium rate obtained by the AS and GAs, so it is concluded that the AS-GA's performance is superior to that of the AS and GAs. Thus, it can be inferred from the experiment's results that the research on the merger of the AS algorithm and GA has produced some outcomes.

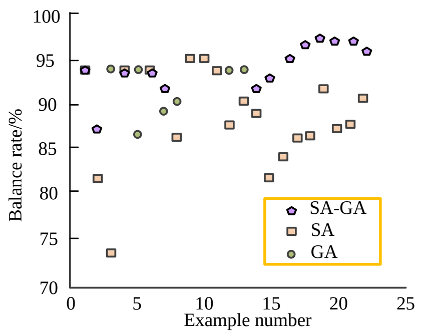
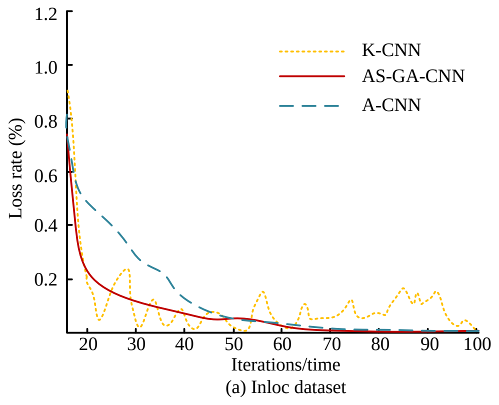


Figure 8: Schematic diagram of the comparison of GA, AS, and AS-GA balance rates

Figure 9 represents the InD keyword relation extraction loss rates for the three experimental models on the Inloc dataset and the ADND. Figure 9(a) represents the variation of the loss rate of the three models on the Inloc dataset with the iterative book publication. The AS-GA-CNN model has a smoother loss rate change curve and after convergence this curve is closer to 0 than the loss rate curves of the other control models, whereas the loss rate curve of the K-CNN model is obviously more fluctuating than that of the other two models and the model has not yet fully converged after 100 iterations. The A-CNN model has a higher initial loss rate although the loss rate is also close to 0 after convergence, but this model has a higher initial loss rate. Figure 9(b) represents the loss rate trends of the three models on the ADND. The lowest loss rate is still the AS-GA-CNN model, the K-CNN model and the A-CNN model both converge within 100 iterations, and the loss rate of the two models is much higher than that of the AS-GA-CNN model.



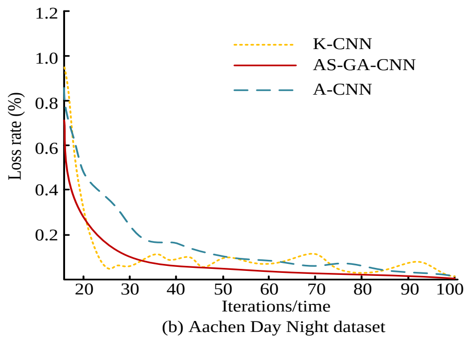
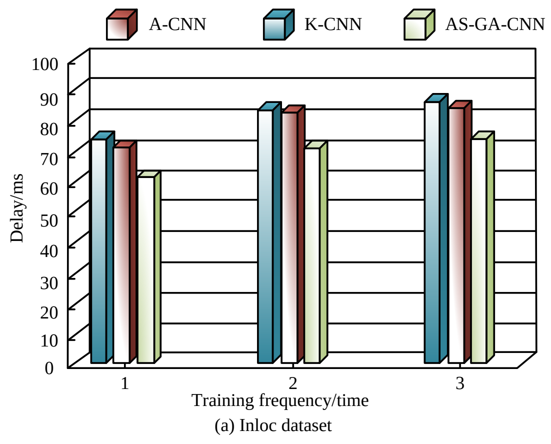


Figure 9: Comparison of loss rate and iteration times of different models

To exam the computational efficiency of the ICM proposed by the study, the study tests the output delay of the experimental model and the control model using the Inloc dataset and the ADND as inputs, respectively. The exam is administered three times in the study to eliminate chance mistakes, and the findings are displayed in Figure 10. Figure 10(a) represents the output delay of each model on the Inloc dataset. Figure 10(b) represents the output delay of each model on the ADND. From Figure 10(a), the average output delay of AS-GA-CNN model can be calculated as 64.90ms. In addition, the average output delay of K-CNN and A-CNN models are 77.16ms and 76.98ms. In Figure 10(b), the average delay of the three models of AS-GA-CNN, K-CNN, and A-CNN are 75.44ms, 83.91ms, and 82.54ms. The output delay control experiment results show that the proposed model performs better than the control model on two different datasets. This suggests that the proposed model has good output delay performance across a wide range of datasets. Based on these findings, it can be concluded that the AS-GA-CNN model has a strong computational mechanism.



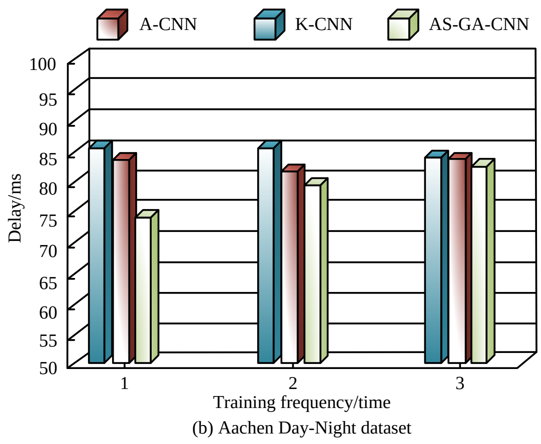


Figure 10: Comparison of classification latency among different models

Figure 11 represents the relationship between the number of pre-training times and the variation of model error rate for different models on the Inloc dataset and the ADND. Figure 11(a) represents the error rate of each model with different number of pre-training times when the Inloc dataset is used as the input. The lowest classification error rate on the Inloc dataset is found in the fourth pre-training for both models except for the K-CNN model and it is 5.11% for the AS-GA-CNN model, while the error rate for the A-CNN model is 6.23%. The error rate of K-CNN model is the lowest at the third pre-training and instead increases after the fourth pre-training. Figure 11(b) represents the error rate comparison of the three models on the ADND. All the experimental models have the lowest error rate at the third pre-training, and the error rate is 3.87% for the AS-GA-CNN model, 5.86% for the A-CNN model, and 6.02% for the K-CNN model. Comparing Figure 11(a) and Figure 11(b), the errors obtained by the proposed model of the study in several experiments are lower than the control model, so the AS-GA-CNN model has a more stable performance in the error test.

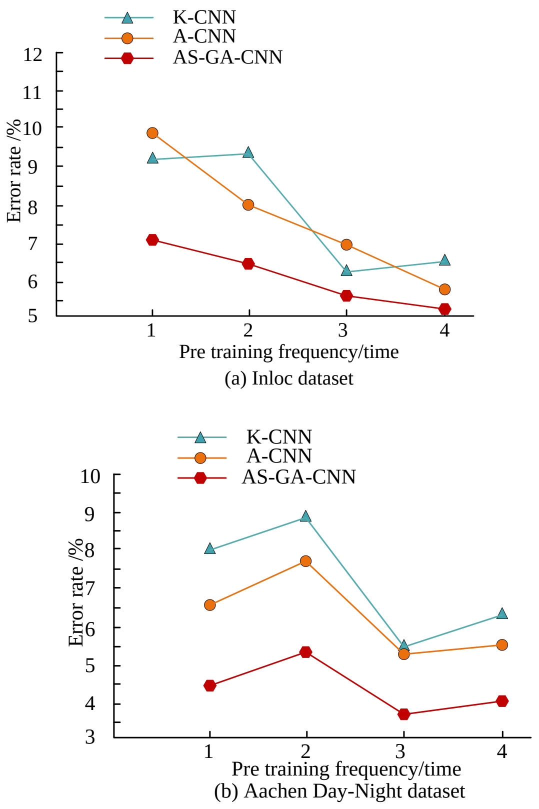


Figure 11: Schematic diagram of the relationship between pre training times and error rate of different models

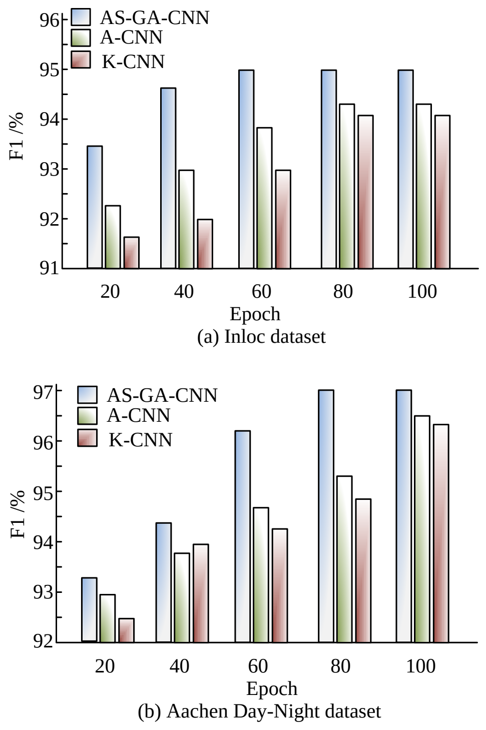
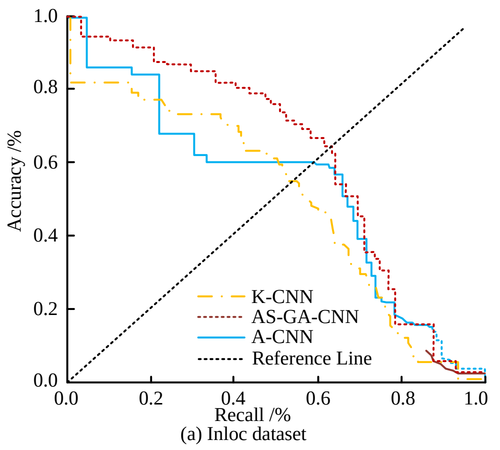


Figure 12: Schematic diagram of F1 score comparison between different models

The F1 score is a metric for comprehensive evaluation of models based on their accuracy and recall. In Figure 12, the study uses the Inloc dataset and the ADND as inputs for the three models and compares the average values of the F1 scores of each model on different iteration steps. Figure 12(a) represents the F1 scores of the models on the Inloc dataset, where the average F1 scores of the AS-GA-CNN model at 20, 40, 60, 80, and 100 iterations are 93.47%, 94.52%, 95.03%, 95.01%, and 95.03%, respectively. The AS-GA-CNN model converged with an F1 score of around 95.03%, while the A-CNN model converged with an average F1 score of 94.37% and the K-CNN model converged with an average F1 score of 94.26%. Figure 12(b) represents the comparison of the F1 scores of the three models on the ADND. The comparison of F1 scores of each model is consistent with Figure 12(a), but the convergence of each model differs from Figure 12(a). The AS-GA-CNN model has a convergence step of 80 on the ADND, while the model has a convergence step of 60 on the Inloc dataset, and the other models have a higher convergence step on the ADND than on the Inloc dataset.

Figure 13 represents the P-R curves of the experimental and control models on different datasets, where Figure 13(a) represents the P-R curves of each model on the Inloc dataset, and Figure 13(b) shows the P-R curves of the experimental model on the ADND. The P-R curve, which is typically used to represent the model, has a reference line that points toward the built position one month backward, indicating greater model performance. Furthermore, the AS-GA-CNN model performs better than the other two control models because it crosses the reference line on the two datasets at a higher position based on the P-R curves' intersection position with the line of the three models. The P-R curve is a thorough assessment indicator, and the AS-GA-CNN model's outstanding performance in this index further highlights the sophisticated character of the study from the side.



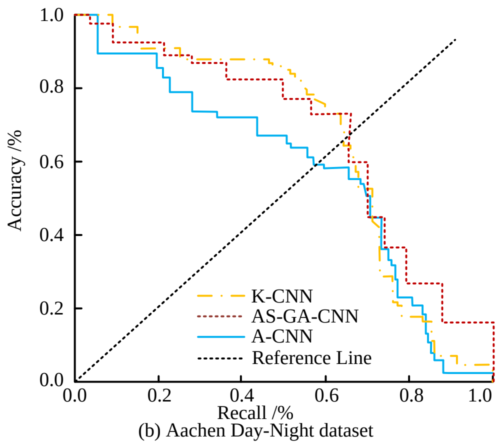
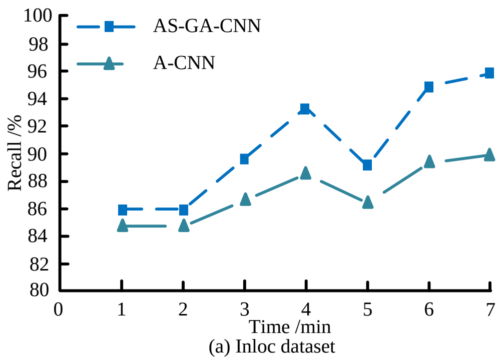


Figure 13: Comparison diagram of P-R curves of different models

producing negative samples leads to a substantially greater recall rate after calculation than the A-CNN model, based on the aforementioned experimental results. This outcome further confirms the good accuracy performance of the suggested model from the side.



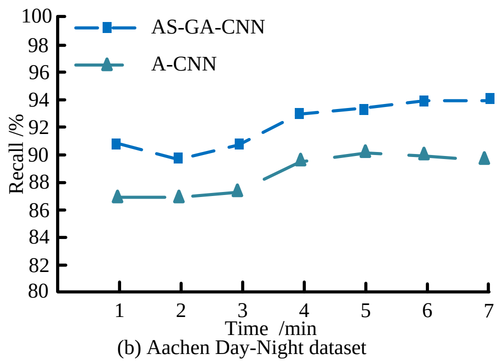


Figure 14: Comparison of recall rates of different models

Recall refers to the number of correctly classified KG samples as a proportion of the number of all KG samples. Using the Inloc dataset and the ADND as inputs, the study records the memory of the experimental and control models in order to test the recall of the proposed model. The findings are displayed in Figure 14. The recall comparison of the models using the Inloc dataset is shown in Figure 14(a). The average recall of AS-GA-CNN model is 91.71%, and the average recall of A-CNN model is 87.06%. In addition, in Figure 14(b), the change of recall of the two models is relatively flat. The average recall of AS-GA-CNN model is 92.11%, and the average recall of A-CNN model is 82.01%. The AS-GA-CNN model's exceptionally low chance of

Table 3 presents the statistical test results for three models. The data presented in Table 3 clearly demonstrates that the AS-GA-CNN model outperforms the A-CNN and K-CNN models in terms of balance rate, output delay, error rate, F1 score, and recall rate. The P-values are all less than 0.05, indicating that these differences are statistically significant. This further verifies the excellent performance and superior performance of the AS-GA-CNN model in intelligent classification tasks.

Table 3: Statistical test results of three models

Index	AS-GA-CNN	A-CNN	K-CNN	Statistical test results
Average equilibrium rate	0.78	0.65	0.62	P < 0.001
Average output delay	70.17ms	79.84ms	80.18ms	P < 0.01
Average error rate	4.99%	6.36%	6.47%	P < 0.05
Average F1 score	95.03%	94.37%	94.26%	P < 0.001
Average recall rate	91.91%	84.54%	87.56%	P < 0.001

4 Discussion

In order to achieve intelligent design in InD, an AS-GA-CNN model has been proposed as a method for summarizing design ideas for the InD industry. This research demonstrated that the average balance rate, output delay, error rate, F1 score, and recall rate of the AS-GA-CNN model are 0.78, 70.17 ms, 4.99%, 95.03%, and 91.91%, respectively. These values were significantly superior to those observed in models that employ solely SA or GA. The primary rationale for employing SA was its capacity to facilitate comprehensive exploration of the search space, whereas GA was adept at conducting in-depth search and optimization of solutions through genetic mechanisms. The combination of the two can more effectively balance the performance indicators of the model and achieve superior performance in intelligent classification tasks. The results of the comparison with other literature demonstrated that the AS-GA-CNN model also exhibits notable advantages. To illustrate, although reference [31] has enhanced the modular AS, it continues to apply the AS algorithm in isolation. In contrast, the algorithm incorporated the GA, which exhibits superior parameter optimization capabilities and a lower error rate. In Reference [32], the attention mechanism was introduced into AS. Although some improvements have been made to the model, its performance in terms of F1 score and recall is inferior to that of the AS-GA-CNN model. This is mainly due to the superior global optimization and deep search capabilities of the AS-GA-CNN model. Furthermore, the superior performance of the AS-GA-CNN model in intelligent classification tasks, when compared with the methods proposed in reference [33], can be fully demonstrated, thereby increasing its applicability. Nevertheless, while the AS-GA-CNN model has demonstrated notable performance advantages in various domains, it is also important to acknowledge its limitations. For instance, the introduction of the GA necessitates additional parameter tuning and experimental verification to guarantee the model's stability and reliability. Furthermore, its convergence may be affected to a certain extent. In conclusion, the AS-GA-CNN model, which integrates SA and GA, represents a significant advancement in intelligent CMs. It offers a promising avenue for further research and development, with the potential to enhance performance and practical application. It is recommended that this model be applied to a wider range of intelligent classification tasks and that more practical field validation and application be conducted.

5 Conclusion

KG can organize and present all kinds of knowledge, information and experience in the design field in a structured form. Therefore, the emergence of KG is of great significance to the InD industry. However, in the face of a large number of KGs, the challenge of efficient classification remains a significant issue. To effectively solve the classification problem of KG, the study tries to optimize the parameters of CNN by using optimization algorithm, so as to improve the classification speed and accuracy of CNN. In view of this, the study employs GA and SAA for parameter optimization of CNN, by which a high-performance CM for KG is constructed, and performance analysis experiments are conducted on the proposed model. The outcomes indicated that the average output latency of the proposed model on the Inloc dataset is 64.90 ms, which is 12.26 ms lower than the K-CNN model and 12.08 ms lower than the A-CNN model. Meanwhile, the error rate of the proposed model on the ADND was 3.87%, while the error rate of the A-CNN model was 5.86%, and that of the K-CNN model has an error rate of 6.02%. In addition, the study also conducted several experiments on the F1 scores, recall, P-R curves and other metrics of the models, and the findings all indicated that the proposed model has a superior performance than the control model, thus concluding that the proposed model is feasible and advanced in KG classification. The proposed CNNCM based on GA and SAA optimization achieves efficient classification of KG for InD and also promotes the development of InD industry from the side, so the research proposed model is of practical significance. Meanwhile, there are several issues with the study. While the introduction of GA corrects the SAA's local convergence issue, it does not deal with the algorithm's poor convergence speed. Therefore, in order to improve it, more research is required.

References

[1] A. Djeddai, and R. Khemaissia, "PrivyKG: security and privacy preservation of knowledge graphs using blockchain technology," Informatica, vol. 47, no. 5, pp. 137-152, 2023. https://doi.org/10.31449/inf.v47i5.4698

[2] K. Bhosle, and V. Musande, "Evaluation of deep learning CNN model for recognition of devanagari digit," Artificial Intelligence and Applications, vol. 1, no. 2, pp. 114-118, 2023. https://doi.org/10.47852/bonviewAIA3202441

[3] A. Santos, A. R. Colaço, A. B. Nielsen, L. Niu, M. Strauss, P. E. Geyer, F. Coscia, N. J. W. Albrechtsen, F. Mundt, L. J. Jensen, and M. Mann, "A knowledge graph to interpret clinical proteomics data," Nature Biotechnology, vol. 40, no. 5, pp. 692-702, 2022. https://doi.org/10.1038/s41587-021-01145-6

[4] L. Yang, "Feature extraction of english semantic translation relying on graph regular knowledge recognition algorithm," Informatica, vol. 47, no. 8, pp. 103-124, 2023. https://doi.org/10.31449/inf.v47i8.4901

[5] A. Sohail, "Genetic algorithms in the fields of artificial intelligence and data sciences," Annals of Data Science, vol. 10, no. 4, pp. 1007-1018, 2023. https://doi.org/10.1007/s40745-021-00354-9

[6] B. Xue, and L. Zou, "Knowledge graph quality management: A comprehensive survey," IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 5, pp. 4969-4988, 2023. https://doi.org/10.1109/TKDE.2022.3150080

[7] X. Wang, D. Bo, C. Shi, S. Fan, Y. Ye, and P. S. Yu, "A survey on heterogeneous graph embedding: methods, techniques, applications and sources," IEEE Transactions on Big Data, vol. 9, no. 2, pp. 415-436, 2023. https://doi.org/10.1109/TBDATA.2022.3177455

[8] H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in graph neural networks: A taxonomic survey," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 5, pp. 5782-5799, 2023. https://doi.org/10.1109/TPAMI.2022.3204236

[9] B. Steenwinckel, G. Vandewiele, M. Weyns, T. Agozzino, F. D. Turck, and F. Ongenae, "INK: knowledge graph embeddings for node classification," Data Mining and Knowledge Discovery, vol. 36, no. 2, pp. 620-667, 2022. https://doi.org/10.1007/s10618-021-00806-z

[10] J. Li, L. Guo, Y. Zuo, and W. Liu, "A design method for wideband chaff element using simulated annealing algorithm," IEEE Antennas and Wireless Propagation Letters, vol. 21, no. 6, pp. 1208-1212, 2022. https://doi.org/10.1109/LAWP.2022.3161762

[11] W. Shi, Z. He, W. Tang, W. Liu, and Z. Ma, "Path planning of multi-robot systems with boolean specifications based on simulated annealing," IEEE Robotics and Automation Letters, vol. 7, no. 3, pp. 6091-6098, 2022. https://doi.org/10.1109/LRA.2022.3165184

[12] D. Shin, N. Onizawa, W. J. Gross, and T. Hanyu, "Memory-efficient FPGA implementation of stochastic simulated annealing," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 13, no. 1, pp. 108-118, 2023. https://doi.org/10.1109/JETCAS.2023.3243260

[13] C. Han, Y. Gu, G. Wu, and X. Wang, "Simulated annealing-based heuristic for multiple agile satellites scheduling under cloud coverage uncertainty," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 53, no. 5, pp. 2863-2874, 2023. https://doi.org/10.1109/TSMC.2022.3220534

[14] S. Chen, Y. Gu, G. Wu, and X. Wang, "Differential evolution based simulated annealing method for vaccination optimization problem," IEEE Transactions on Network Science and Engineering, vol. 9, no. 6, pp. 4403-4415, 2022. https://doi.org/10.1109/TNSE.2022.3201079

[15] W. Xia, Q. Gao, Q. Wang, X. Gao, C. Ding, and D. Tao, "Tensorized bipartite graph learning for multi-view clustering," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 4, pp. 5187-5202, 2023. https://doi.org/10.1109/TPAMI.2022.3187976

[16] G. Bouritsas, F. Frasca, S. Zafeiriou, and M. M. Bronstein, "Improving graph neural network expressivity via subgraph isomorphism counting," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 1, pp. 657-668, 2023. https://doi.org/10.1109/TPAMI.2022.3154319

[17] M. Zhang, S. Wu, X. Yu, Q. Liu, and L. Wang, "Dynamic graph neural networks for sequential recommendation," IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 5, pp. 4741-4753, 2023. https://doi.org/10.1109/TKDE.2022.3151618

[18] M. Ince, "Automatic and intelligent content visualization system based on deep learning and genetic algorithm," Neural Computing and Applications, vol. 34, no. 3, pp. 2473-2493, 2022. https://doi.org/10.1007/s00521-022-06887-1

[19] J. Zhou, S. Huang, T. Zhou, D. J. Armaghani, and Y. Qiu, "Employing a genetic algorithm and grey wolf optimizer for optimizing RF models to evaluate soil liquefaction potential," Artificial Intelligence Review, vol. 55, no. 7, pp. 5673-5705, 2022. https://doi.org/10.1007/s10462-022-10140-5

[20] P. Preethi, and H. R. Mamatha, "Region-based convolutional neural network for segmenting text in epigraphical images," Artificial Intelligence and Applications, vol. 1, no. 2, pp. 119-127, 2023. https://doi.org/10.47852/bonviewAIA2202293

[21] Z. Yu, Z. Si, X. Li, D. Wang, and H. Song, "A novel hybrid particle swarm optimization algorithm for path planning of UAVs," IEEE Internet of Things Journal, vol. 9, no. 22, pp. 22547-22558, 2022. https://doi.org/10.1109/JIOT.2022.3182798

[22] I. S. Mohamad Hashim, A. Al-Hourani, and B. Ristic, "Satellite localization of iot devices using signal strength and doppler measurements," IEEE Wireless Communications Letters, vol. 11, no. 9, pp. 1910-1914, 2022. https://doi.org/10.1109/LWC.2022.3187065

[23] H. Liu, Y. Li, and S. Wang, "Request scheduling combined with load balancing in mobile-edge computing," IEEE Internet of Things Journal, vol. 9, no. 21, pp. 20841-20852, 2022. https://doi.org/10.1109/JIOT.2022.3176631

[24] D. He, X. Yu, T. Li, S. Chan, and M. Guizani, "Firmware vulnerabilities homology detection based on clonal selection algorithm for IoT devices," IEEE Internet of Things Journal, vol. 9, no. 17, pp. 16438-16445, 2022. https://doi.org/10.1109/JIOT.2022.3152364

[25] L. Zhai, and S. Feng, "A novel evacuation path planning method based on improved genetic algorithm," Journal of Intelligent & Fuzzy Systems, vol. 42, no. 3, pp. 1813-1823, 2022. https://doi.org/10.3233/JIFS-211214

[26] Y. Li, X. Li, and L. Gao, “An effective solution space clipping-based algorithm for large-scale permutation flow shop scheduling problem,” IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 53, no. 1, pp. 635-646, 2023. https://doi.org/10.1109/TSMC.2022.3187082

[27] S. T. Shishavan, and F. S. Gharehchopogh, “An improved cuckoo search optimization algorithm with genetic algorithm for community detection in complex networks,” Multimedia Tools and Applications, vol. 81, no. 18, pp. 25205-25231, 2022. https://doi.org/10.1007/s11042-022-12409-x

[28] A. Boughida, M. N. Kouahla, and Y. Lafifi, "A novel approach for facial expression recognition based on Gabor filters and genetic algorithm," Evolving Systems, vol. 13, no. 2, pp. 331-345, 2022. https://doi.org/10.1007/s12530-021-09393-2

[29] F. H. Rizk, S. Arkhstan, A. M. Zaki, M. A. Kandel, and S. K. Towfek, “Integrated CNN and waterwheel plant algorithm for enhanced global traffic detection,” Journal of Artificial Intelligence and Metaheuristics, vol. 6, no. 2, pp. 36-45, 2023. https://doi.org/10.54216/JAIM.060204

[30] S. R. Waheed, M. S. M. Rahim, N. M. Suaib, and A. A. Salim, “CNN deep learning-based image to vector depiction,” Multimedia Tools and Applications, vol. 82, no. 13, pp. 20283-20302, 2023. https://doi.org/10.1007/s11042-023-14434-w

[31] K. Shi, Z. Wu, B. Jiang, and H. R. Karimi, “Dynamic path planning of mobile robot based on improved simulated annealing algorithm,” Journal of the Franklin Institute, vol. 360, no. 6, pp. 4378-4398, 2023. https://doi.org/10.1016/j.jfranklin.2023.01.033

[32] X. Mo, Z. Huang, Y. Xing, and C. Lv, "Multi-agent trajectory prediction with heterogeneous edge-enhanced graph attention network," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 9554-9567, 2022. https://doi.org/10.1109/TITS.2022.3146300

[33] X. Guo, and L. Zhao, "A systematic survey on deep generative models for graph generation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 5, pp. 5370-5390, 2023. https://doi.org/10.1109/TPAMI.2022.3214832