TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 19/20 pp1851–1871
DOI: 10.26599/TST.2024.9010039
Volume 29, Number 6, December 2024

Adaptive Model Compression for Steel Plate Surface Defect Detection: An Expert Knowledge and Working Condition-Based Approach

Maojie Sun, Fang Dong*, Zhaowu Huang, and Junzhou Luo

Abstract: The steel plate is one of the main products in steel industries, and its surface quality directly affects the final product performance. How to detect surface defects of steel plates in real time during the production process is a challenging problem. The single or fixed model compression method cannot be directly applied to the detection of steel surface defects, because it is difficult to consider the diversity of production tasks, the uncertainty caused by environmental factors, such as communication networks, and the influence of process and working conditions in steel plate production. In this paper, we propose an adaptive model compression method for steel surface defect online detection based on expert knowledge and working conditions. First, we establish an expert system to give lightweight model parameters based on the correlation between defect types and manufacturing processes. Then, lightweight model parameters are adaptively adjusted according to working conditions, which improves detection accuracy while ensuring real-time performance. The experimental results show that compared with the detection method of constant lightweight parameter model, the proposed method makes the total detection time cut down by 23.1%, and the deadline satisfaction ratio increased by 36.5%, while upgrading the accuracy by 4.2% and reducing the false detection rate by 4.3%.

Key words: steel surface defect detection; inference acceleration; model compression; expert knowledge; pruning; quantization

1 Introduction

As one of the main product forms in the steel industry, the steel plate has become an indispensable raw material for automobile, machinery manufacturing, aerospace, and marine industries. The quality of its surface directly affects the performance of final products. How to detect the surface defect of steel plates in real time during the production process, so as to control and improve the surface quality of steel plates, has been a great concern for steel production and processing enterprises[1–4]. In recent years, the booming development of artificial intelligence technology has provided new ideas for steel plate surface defect detection. The Deep Neural Networks (DNN) method has significantly improved detection accuracy, but also brought a larger amount of computation time and longer detection time. How to improve the real-time performance of DNN detection while ensuring accuracy is a very challenging research direction[5–8].

Model compression techniques are widely used to accelerate the execution of DNNs through a series of technical ways that improve the efficiency of the model in terms of computation, storage, transmission, etc.

• Maojie Sun is with School of Computer Science and Engineering, Southeast University, Nanjing 211189, China, and also with Najing Iron and Steel Co., Nanjing 210035, China. E-mail: sunmj@njsteel.com.cn.
• Fang Dong, Zhaowu Huang, and Junzhou Luo are with School of Computer Science and Engineering, Southeast University, Nanjing 211189, China. E-mail: fdong@seu.edu.cn; zwh@seu.edu.cn; jluo@seu.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 2023-11-30; revised: 2024-02-02; accepted: 2024-02-22

Common methods include knowledge distillation, pruning, quantization, etc.[9–11]. Knowledge distillation obtains another simple network by employing the output of a pre-trained model as a supervised signal to be trained[12]. Pruning techniques reduce the redundant parts of the model by removing unnecessary neurons or connections[13]. Quantization is the conversion of a model's parameters from high precision (e.g., 32-bit floating-point numbers) to low-precision representations (e.g., 8-bit integers) to reduce the model's storage and computational burden[14]. Although these model compression techniques have achieved rich results, they are limited to lightweight compression of large complex models from a single dimension (i.e., model structure and parameter bit width). Supposed that the model structure and parameter bit-width are compressed excessively to meet the real-time demand of steel surface defect detection, some important model structures may be cropped or the parameter accuracy may be too low, which may seriously affect the recognition accuracy.

Therefore, this paper explores the flexible compression of the model in two dimensions using a combination of pruning and quantization. In this way (considering that the knowledge distillation is difficult to be parameterized), we use pruning techniques to eliminate "unimportant" weights at the model structure level to reduce the number of parameters and computational effort of the model, followed by quantization techniques to reduce the model size by decreasing the parameter bit-widths of the model, which further reduces the computational overhead. Through the combination of pruning and quantization techniques, the model structure and parameter width can be compressed into two dimensions, and a more lightweight model can be realized while ensuring accuracy.

However, simply combining pruning and quantization techniques does not provide the desired advantages. Our pre-experimental results show (see Section 2) that even with simultaneous pruning and quantization of the pre-trained model, it still struggles to meet the performance requirements when applied to online production of steel surface defect detection applications. This is due to the fact that the fixed model compression method does not take the influence of different process parameters and actual production conditions into account, resulting in a lightweight model that is not adaptive to the diversity and dynamics of steel surface defect detection application. Different grades of steel plates have different process parameters (such as rolling pressure, slab thickness, steel grade, etc.), which results in various types of defects with different probability distributions. This diversity results in a constant compression mode cannot be adapted to multiple process parameters. What's more, the working conditions in the steel production environment show the characteristics of dynamic change. This dynamic nature results in the chosen lightweight model no longer matching the production environment. Therefore, according to the diversity and dynamic characteristics of steel plate surface detection, the model compression needs to be adjusted adaptively according to the process and working conditions to meet the requirements of high precision and low delay of defect detection.

Unfortunately, there are challenges to achieve this effect. (1) Due to the huge value space of the combined pruning and quantization parameters of the pre-trained model, if all the combined lightweight models are trained in advance as a collection of candidate models, it will lead to a huge training overhead. (2) Iron and steel production process parameters are complex and heterogeneous, the matching relationship between the lightweight model and the process parameters is unknown and difficult to be formally described, so it is difficult to confirm the optimal selection of pruning and quantization parameters according to different process parameters. (3) The network environment, the system load, and the dynamic change of the steel roll state are uncertain, and it is difficult to accurately estimate the future change of the working conditions in advance to configure the corresponding lightweight model. Therefore, how to achieve adaptive model lightweight to meet the application of high-precision and low-latency needs is an important but very difficult problem in the steel surface defect detection scenario with the diversity of process parameters and the dynamic change of working conditions.

Most of existing works have mainly focused on the high-precision recognition for a given image or the effective compression for a given model[15, 16]. DNN methods on the steel plate surface defect detection are also concerned with the optimization of network structures, aiming to improve the classification of defects and the accuracy of localization[17, 18]. Few of them have considered the adaptive lightweight of the model in the steel surface defect detection scenarios, and hence they cannot be applied to solve the problem in this paper.

To solve the above problems, this paper proposes an adaptive model compression mechanism for the steel plate surface defect detection based on the expert knowledge and the working condition. The core principle of the method is to pre-train a mold lightweight model set by combining pruning and quantization techniques, and select the most suitable lightweight model quickly and adaptively according to different working conditions, so as to maximize the detection accuracy under the condition of ensuring time constraints. (1) For the problem on the huge space for pruning and quantization, we propose a clustering method based on K-means, which greatly reduces the space of candidate models. (2) For the problem on the unknown relationship between the model and the process parameters, an expert system based on the process parameters is constructed to select the lightweight parameters of the model. (3) For the problem on the dynamic change of working conditions, an online optimization method of lightweight parameters based on feedback of detection results is proposed.

Specifically, we first analyze the correlation between process parameters, such as steel grade, size, limiting time, and surface defects of steel plate, based on which an expert knowledge system of lightweight parameters is constructed. Secondly, the knowledge base is clustered to establish an offline lightweight model library, which provides the basis for algorithm scheduling. Then, for a given production task, based on the model compression scheme given by the expert system, a lightweight parameter selection optimization model is established according to the actual working conditions and detection results, and solved by particle swarm optimization algorithm. Finally, the optimal solution is fed back to the model library in real time for algorithm selection and scheduling, so as to realize the adaptive lightweight of the model.

In terms of experiments, we carry out the defect detection on steel plates under different processes and working conditions. For the same production task, two methods are used: the constant lightweight parameter model and the adaptive lightweight model (that is, the method proposed in this paper). Then, comparisons are made on the accuracy, the false detection rate, the total detection time, the maximum time for single sheet detection. In addition, the effects of the model library size, the single-picture constraint time, and the total time constraint on the proposed method are examined. The experiments show that our proposed method achieves the adaptive lightweight of the model, and has advantages in both accuracy and timeliness. Compared with the detection methods of constant lightweight parameter model, it cuts down the total detection time by 23.1%, and increases the deadline satisfaction ratio by 36.5%, while upgrading the accuracy by 4.2% and reducing the false detection rate by 4.3%.

The main contributions of this paper consist of the following three aspects.

(1) This paper finds that a single or fixed model compression method is difficult to meet the diversity and the dynamics of steel surface defect detection scenarios, and for the first time considers the problem of adaptive model compression according to the process and working conditions, which solves the problem of balancing the high accuracy and the low latency in steel plate surface defect detection.

(2) A new method of adaptive model compression based on real-time feedback is proposed, which establishes a lightweight model library through an expert system and offline training, integrates the current detection results and working condition parameters into the optimization of lightweight parameters, and carries out online scheduling according to the production tasks. This ensures the timeliness of detection while adapting to the complicated and changeable environment.

(3) To verify the effectiveness of the method proposed in this paper, a large number of experiments are carried out in the actual steel production environment, and the results show that the method in this paper is better than the algorithm with the constant lightweight parameter in terms of real time and accuracy.

2 Background and Motivation

This section gives the background, and aims to clarify the main issues to be addressed existing in the DNN-based detection of steel plate surface, and carry out some pre-experiments to illustrate the motivation.

2.1 Background

How to detect the surface defects of steel plates in real-time during the production process, so as to control and improve the surface quality of steel plates, has been a great concern for steel production and processing enterprises. The basic structure of the steel plate surface defects detection system is shown in Fig. 1, which is composed of a hardware subsystem and a software subsystem. The hardware subsystem mainly consists of cameras, line light source, computers, and encoders. The light source illuminates the steel plate, the encoder triggers the camera to collect the image of the steel plate surface on the conveyor rollers. The computer is the carrier and the executor of the software subsystem, and processes the collected image to obtain the defects on the steel plate surface.

In the production process, the surface of the steel plate may appear various defects, due to the quality of raw materials, mechanical damage during processing, improper processing temperature or pressure, non-standard operation, as well as humidity, moisture, dust and other external environment. These defects not only affect the appearance of the product, but also reduce the performance of the product, such as corrosion resistance, wear resistance, and fatigue strength. Common defects on the surface of the steel plate are mainly of two types. One is the linear defect, such as cracks; the other is the block defect, such as inclusions, scars, double skin, impurity pressed-in defect, etc., samples of which are shown in Fig. 2.

In actual detection, we take note of the facts: once the crack is detected, the next steel plate should be carefully inspected, because the crack is often longer; once detected scab defects, the same motherboard number under the sub-board number should focus on detection; double skin defects are often found at the beginning and end of the roll (end of sub-plate number 02 and the penultimate plate); for inclusion defects, it is necessary to focus on the head and tail billet number, and the same furnace number of rolled pieces to focus on the detection. These show that the surface quality of the steel plate is closely related to the process and working conditions, revealing that we should take full account of the process parameters in the model compression, and at the same time, the algorithm should be adjusted adaptively according to the working conditions.

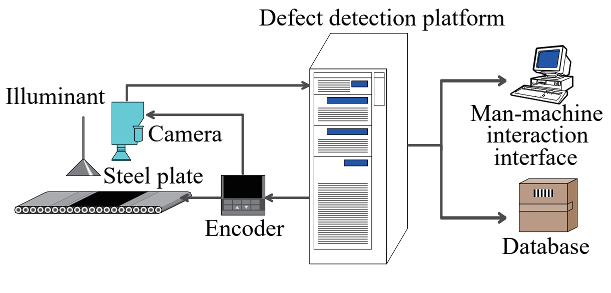


Fig. 1 Steel plate surface detection system.

2.2 Motivation

This subsection carries out some pre-experiments to clarify the research motivation. For the two types of defects, there are pre-trained DNN models in practice, which may be called the Linear Defect Detection (LDD) model and Block Defect Detection (BDD) model. The LDD model is a detection model for cracks and other linear defects, and the BDD model is a detection model for block defects. Three forms of compression processing are adopted for the LDD model and the BDD model, the first one is the same compression form for both, with 8 bits of quantization and a pruning ratio of 0.4 for depth and width, which is called Model Ⅰ. The second, the LDD model has 4 bits of quantization and a pruning ratio of 0.6 for depth and width, and the BDD model has 8 bits of quantization and a pruning ratio of 0.3 for depth and width, referred to as Model Ⅱ. The third, the LDD model with 8 bits of quantization and a pruning ratio of 0.3 for depth and width, and the BDD model with 4 bits of quantization and a pruning ratio of 0.6 for depth and width, called Model Ⅲ.

The following examples illustrate that it is better to consider the process parameters, working conditions, and detection results when the model is compressed.

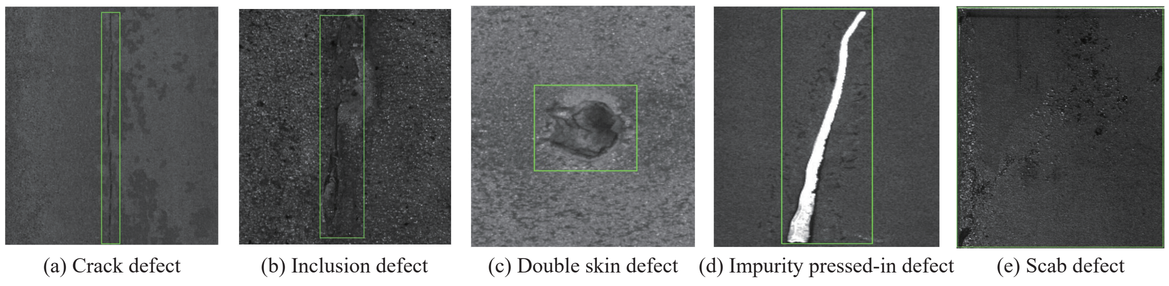


Fig. 2 Defect samples on surface of the steel plate.

Therefore, in the case that the constant lightweight parameter model cannot meet the demand, the model can be compressed adaptively according to the process and working conditions to get the expected results.

(1) On the motivation to consider process parameters, such as steel grade in the model compression

There are 35 pictures of steel plates available. The corresponding relationship between steel plate number and steel grade is as follows: Nos. 1–5, bridge steel; Nos. 6–10, shipboard steel; Nos. 11–15, medium carbon steel; Nos. 16–20, Engineering machinery steel; Nos. 21–25, pipeline steel; Nos. 26–30, wind tower steel; Nos. 31–35, market circulation steel.

Two methods are taken to process these 35 pictures. Method 1-1 uses only accelerated Model Ⅰ throughout the whole process, Method 1-2 uses Model Ⅰ for Nos. 1–20, and Model Ⅱ for Nos. 21–35. Both methods are able to detect defects and meet the requirements in terms of accuracy, but the detection time is different, as shown in Fig. 3a.

It can be seen that Model Ⅱ takes less time than Model Ⅰ to process the pictures of steel plates Nos. 21–35. According to the expert knowledge, it is known that the probability of block defects is relatively large for steel grades Nos. 21–35. The compression of the LDD model in Model Ⅱ is greater than that of the LDD model in Model Ⅰ, and the compression of the BDD model in Model Ⅱ is less than that of the BDD model in Model Ⅰ. Therefore, Model Ⅱ performs better than Model Ⅰ in the detection of steel plates of Nos. 21–35, and the detection time is reduced by 16%. This tells us that different steel grades may require different compression models, because the steel grades are closely related to the defects. Therefore, the influence of process parameters, such as steel grade, should be taken into account in the model compression.

(2) On the motivation to consider working conditions, such as the steel billet number in the model compression

There are another 35 steel plates that come from 5 billets of the same furnace and are all medium carbon steel. The corresponding relationship between steel plate number and billet is as follows: Nos. 1–7, billet 1; Nos. 8–14, billet 2; Nos. 15–21, billet 3; Nos. 22–28, billet 4; Nos. 29–35, billet 5.

Two methods are taken to process these 35 pictures. Method 2-1 uses only accelerated Model Ⅰ throughout. Method 2-2 uses Model Ⅰ for Nos. 1–7 and Nos. 29–35, and Model Ⅲ for Nos. 8 –28. Both methods are able to detect the defects, and the detection time is shown in Fig. 3b.

It can be seen that Model Ⅲ uses less time than Model Ⅰ in processing the pictures of steel plates Nos. 8–28. According to the expert knowledge, we know that the probability of lumpy defects in the steel plates Nos. 8–28 is smaller. The compression degree of the LDD model in Model Ⅲ is smaller than that of the LDD model in Model Ⅰ, and the compression degree of the BDD model in Model Ⅲ is larger than that of the BDD model in Model Ⅰ. As a result, Model Ⅲ performs better than Model Ⅰ in detecting the steel plate of Nos. 8–28, and the detection time is shortened by about 17%. This tells us that different billets may require different compression models because the working conditions are closely related to the defects. Therefore, the influence of working condition parameters such as steel billet should be considered in model lightweight.

(3) On the motivation to consider the detection result of the last steel plate in the model compression

Another 35 plates, which come from the same billet in the same furnace (same steel grade), have a long crack in plates 1–10. Two methods are taken to process the 35 pictures. Method 3-1 uses only accelerated Model Ⅰ throughout. Method 3-2 is that Model Ⅰ is used for the 1st picture, and when a crack is detected, Model Ⅲ is started until no crack is detected, after which it is reverted back to Model Ⅰ. The detection time is shown in Fig. 3c.

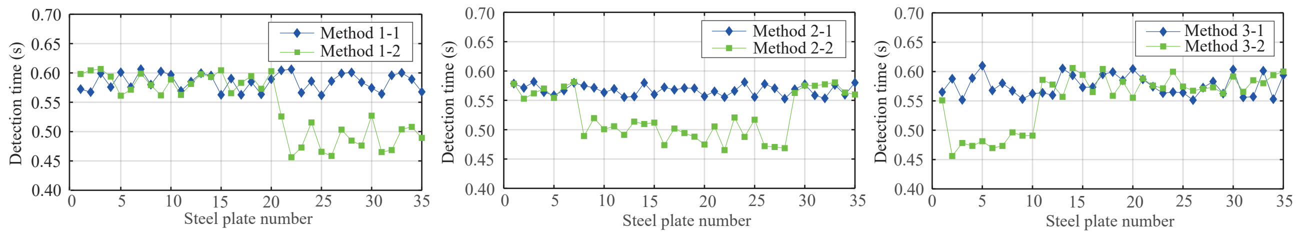


Fig. 3 Comparison of detection time.

It can be seen that Model Ⅲ takes less time than Model Ⅰ to process the steel plate pictures of Nos. 2–10. The detection of Nos. 2–10 with Model Ⅲ can be interpreted as feedback on the results of the defects from the previous sheet. The compression of the LDD model in Model Ⅲ is less than the compression of the LDD model in Model Ⅰ. The compression of the BDD model in Model Ⅲ is greater than the compression of the BDD model in Model Ⅰ, and the compression of the BDD model in Model Ⅲ is greater than the compression of the BDD model in Model Ⅰ. In consequence, Model Ⅲ performs better than Model Ⅰ in the detection of steel plate Nos. 2–10, the detection time is reduced by about 20%. This tells us that the detection results can be used to guide the choice of model. Because the location of the defects in the front plate is closely related to the defects in the subsequent plates. Therefore, the detection results of the previous plate should be taken into account in the model compression.

The above and many more other practical examples show that the direct use of compressed models is often difficult to achieve the desired time effect. On the one hand, it can not be adapted to a variety of production needs. On the other hand, it is difficult to adapt to the complexity of the field environment. One of main reasons is that they can not be designed to take into account the actual working conditions in the training phase. In order to meet the expected time requirements, firstly, the models should be compressed from the perspective of the models themselves. Secondly, the adaptability of the model should be enhanced through real-time feedback according to the production and detection conditions in the field.

3 Design of System Framework

Based on the discussion in the previous section, the expert experience, the production process, and the working condition are three factors that must be considered to realize adaptive model compression in the scene of steel plate surface detection.

The field experts have deep expertise and rich practical experience, and building up the expert knowledge base from this experience and knowledge has an important value for the lightweight design of the algorithm. In addition, the various situations and conditions faced in the production line can directly affect the quality of steel plates. The real-time feedback of working condition information to the algorithm design can effectively deal with various uncertainties. With this in mind, this paper proposes an adaptive model lightweight design method based on expert knowledge and working condition-driven optimization.

The methodology of this paper consists of three main modules. The first module is the establishment of an expert system based on process parameters, mainly including the knowledge representation of process parameters and surface defects, case base establishment of model lightweight parameter, and case reasoning based on sequential search, etc., which will be introduced in Section 4.1. The second module is the model library building based on the knowledge base, including K-means clustering of knowledge base, pre-trained model compression based on lightweight parameters, nearest-neighbor matching principle, etc., which will be introduced in Section 4.2. The third module is the lightweight parameter optimization based on the working condition-driven algorithm, including the correlation analysis between the detection results and the working conditions, the working condition-driven lightweight parameter optimization modeling, the optimal lightweight parameter solution, etc., which will be introduced in Section 4.3. The three modules cooperate to realize the adaptive compression of the model. The expert system is fundamental and provides the basis for the establishment of the model library. The model library is the guarantee for the implementation of the mechanism, providing different models according to different working conditions. The lightweight parameter optimization of the algorithm is the core to realize the adaptive algorithm while taking into account the accuracy and timeliness. The whole process is shown in Fig. 4.

In general, the whole process can be summarized as follows. The expert system gives a candidate lightweight parameter set. Then, the particle swarm algorithm is used to give the optimal lightweight parameters from the candidate lightweight parameter set according to the working conditions. Then, based on the nearest-neighbor principle, the lightweight model applicable to the steel plate picture is selected from the model library. In this cycle, different steel plates use different compression models for adapting to the diversity and dynamics of detection.

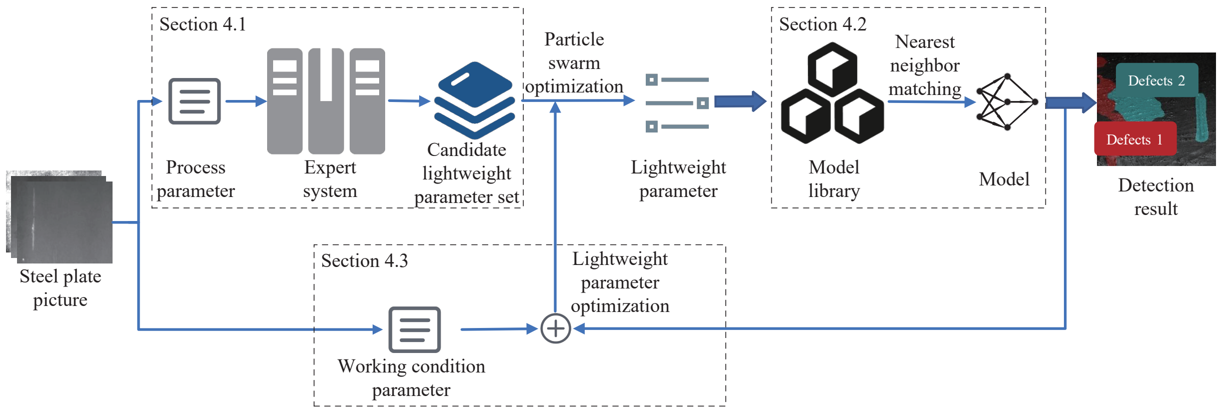


Fig. 4 Adaptive model lightweight mechanism.

4 Adaptive Model Compression Mechanism Based on Expert Knowledge and Working Condition

This section first analyzes the correlation between process parameters (such as steel grade and size of steel plate) and surface defects, and uses this as the basis to construct an expert knowledge system for lightweight parameters. Secondly, the expert knowledge base is clustered to establish an offline lightweight model library. Then, according to the production tasks, working conditions, and detection results, the lightweight parameters selection optimization model is given, and solved by particle swarm optimization algorithm, to realize the adaptive lightweight of the model.

4.1 Expert system for lightweight model parameters on the steel plate process

According to the construction process of expert system, the first is knowledge representation of process parameters and surface defects, then is the case database of lightweight model parameters, and finally is the case reasoning rule of sequential search.

4.1.1 Knowledge representation of process parameters and surface defects

The process parameter of steel plate picture is denoted as $\alpha = [\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}} \in \mathbf{R}^3$, where α_1 denotes the steel grade, including bridge steel, shipboard plate steel, pipeline steel, medium carbon steel, engineering machinery steel, wind tower steel, and market circulation steel; α_2 means the size, including thick gauge, thin gauge, wide gauge, extra long gauge, and common gauge; α_3 is the processing limited time, which gives the upper limit of the execution time of the algorithm when processing the picture.

In the knowledge base of expert system, a specific case can be understood as an instance of this type of problem, and at the same time, this case also includes a specific solution to this type of problem. In this expert system, a case is expressed using a formulation based on the steel plate picture process parameter $\alpha = [\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}}$, where the value of α_1 is given by Table 1 in relation to the steel grade, the relation between the value of α_2 and the size is given by Table 2.

The solution of the case is represented by a set of lightweight parameters $\varphi = [q_1, q_2, c_1, c_2]^{\mathrm{T}} \in \mathbf{R}^6$. q_i denotes the number of quantization bits of model i, $c_i = [c_{i,d}, c_{i,w}]$ denotes the pruning ratio of model i, where its components $c_{i,d}$ and $c_{i,w}$ denote the pruning proportions in depth and width, respectively, $i = 1, 2$. In this form, the process parameters of the case are represented and stored in the knowledge base with the lightweight parameters of the corresponding algorithm. The inference mechanism has to reason based on the

Table 1 Value of α_1 and the corresponding steel grade.

α_1	Steel grade
1	Bridge steel
2	Shipboard steel
3	Pipeline steel
4	Medium carbon steel
5	Engineering machinery steel
6	Wind tower steel
7	Market circulation steel

Table 2 Value of α_2 and the corresponding steel size.

α_2	Steel size
1	Thick gauge
2	Thin gauge
3	Wide gauge
4	Extra long gauge
5	Common gauge

contents of the knowledge base.

According to expert experience, the type of defects on the surface of a steel plate (denoted by r and given by Table 3) is closely related to the grade of steel. The relationship is shown in Table 4, where the first row is the grade of steel, the first column is the type of defect, and each of the remaining elements indicates the correlation between the corresponding steel grade and the type of defect, with "+" indicating a greater correlation and "/" indicating a lesser correlation. Similarly, the correlation between defect type and plate size is given by Table 5.

4.1.2 Case database of lightweight model parameters

The next content focuses on the issue that when given a case list $\alpha=[\alpha_1,\alpha_2,\alpha_3]^{\mathrm{T}}$, how to give its solution $\varphi=[q_1,q_2,c_1,c_2]^{\mathrm{T}}$ based on the knowledge of experts.

From Table 4, it can be seen that when $\alpha_1\in\{5,6,7\}$, the possibility of block defects is greater, and the likelihood of cracks is smaller. The possibility of cracks is smaller, and the pruning ratio of the LDD model is larger than that of the BDD model. For the quantization digit, the LDD model is smaller than the BDD model, i.e., there are

Table 3 Value of r and the corresponding defect.

r	Defect
1	Crack defect
2	Inclusion defect
3	Scab defect
4	Double skin defect
5	Impurity pressed-in defect

Table 4 Correlation between steel grade and defect type.

r	α_1						
	1	2	3	4	5	6	7
1	+	+	+	+	/	/	/
2	+	+	+	+	+	+	+
3	/	/	/	/	+	+	+
4	+	/	+	/	/	/	/
5	/	/	/	/	+	/	+

Table 5 Correlation between steel size and defect type.

r	α_2				
	1	2	3	4	5
1	+	/	/	/	/
2	/	/	/	/	/
3	/	/	/	+	/
4	/	/	/	/	/
5	/	/	+	/	/

$$q_1\leqslant q_2,\ c_{1,d}\geqslant c_{2,d},\ c_{1,w}\geqslant c_{2,w} \tag{1}$$

From Table 5, it can be seen that when $\alpha_2=1$, the possibility of line defects is greater than that of fast defects, when there are

$$q_1\geqslant q_2,\ c_{1,d}\leqslant c_{2,d},\ c_{1,w}\leqslant c_{2,w} \tag{2}$$

And the possibility of block defects is higher when $\alpha_2\in\{3,4\}$, Formula (1) is still present at this point.

Define the following set:

$$P_1=\Big\{[q_1,\ q_2,\ c_1,\ c_2]^{\mathrm{T}}\in\mathbf{R}^6:\ q_1\geqslant q_2,\ c_{1,d}\leqslant c_{2,d},\ c_{1,w}\leqslant c_{2,w}\Big\} \tag{3}$$

$$P_2=\Big\{[q_1,\ q_2,\ c_1,\ c_2]^{\mathrm{T}}\in\mathbf{R}^6:\ q_1\leqslant q_2,\ c_{1,d}\geqslant c_{2,d},\ c_{1,w}\geqslant c_{2,w}\Big\} \tag{4}$$

According to Formulas (1) and (2), expert knowledge tells us: When $\alpha_2=1$, the solution of α favors $\varphi\in P_1$; When $\alpha_1\in\{5,6,7\}$ or when $\alpha_2\in\{3,4\}$, $\varphi\in P_2$.

The pre-trained model of the LDD model is denoted as $M_0^{\{1\}}$, and the one of the BDD model is denoted as $M_0^{\{2\}}$. Their quantization bits are denoted as $q_0^{\{i\}}$, and the computational cost is denoted as $C_0^{\{i\}}$. The depth, width, and input resolution are denoted as $\mathcal{D}(M_0^{\{i\}})$, $\mathcal{W}(M_0^{\{i\}},l)$, and $\mathcal{R}(M_0^{\{i\}})$, respectively. Specifically, $\mathcal{D}(M_0^{\{i\}})$ denotes the number of blocks contained in the model $M_0^{\{i\}}$, $\mathcal{W}(M_0^{\{i\}},l)$ denotes the number of filters in a layer l of $M_0^{\{i\}}$, $\mathcal{R}(M_0^{\{i\}})$ is the edge length of the $M_0^{\{i\}}$ input image, $i=1,2$.

Let $M^{\{i\}}$ be a lightweight model of $M_0^{\{i\}}$, defining its depth, width, and pruning ratio columns of input resolution as follows:

$$c_{i,d}=\frac{\mathcal{D}(M^{\{i\}})}{\mathcal{D}(M_0^{\{i\}})},\ c_{i,w}=\frac{\mathcal{W}(M^{\{i\}},\ l)}{\mathcal{W}(M_0^{\{i\}},\ l)},$$
$$s_i=\frac{\mathcal{R}(M^{\{i\}})}{\mathcal{R}(M_0^{\{i\}})},\ i=1,2 \tag{5}$$

Note that the quantization bits of the model $M^{\{i\}}$ are $q_i=s_iq_0^{\{i\}}$, which indicates that q_i and s_i can be expressed linearly with respect to each other. Therefore, in the sense of ignoring a constant term coefficient, the expression for s_i can be replaced by q_i, or vice versa. This will not be repeated in the following. In a given computational setting, the execution time τ_i of $M^{\{i\}}$ is closely related to the number of quantization bits and the pruning ratio. This relationship can be given empirically or fitted numerically, notated as $\tau_i=\tau_i\,(c_{i,d},\ c_{i,w},\ q_i)$, $i=1,2$.

For a given case $\alpha = [\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}}$, the solution $\varphi = [q_1, q_2, c_1, c_2]^{\mathrm{T}}$ is obtained based on expert knowledge, and can be described, shown at the bottom of this page,

$$\underset{q_1, q_2, c_{1,d}, c_{1,w}, c_{2,d}, c_{2,w}}{\arg\max} \min \{\mathcal{F}_1(c_{1,d}, c_{2,w}, q_1; \Theta_1), \mathcal{F}_2(c_{2,d}, c_{2,w}, q_2; \Theta_2)\} \tag{6}$$

$$\text{s,t.,} \ \tau_1(c_{1,d}, c_{1,w}, q_1) + \tau_2(c_2, c_{2,w}, q_2) \leqslant \alpha_3 \tag{7}$$

$$[q_1, q_2, c_{1,d}, c_{1,w}, c_{2,d}, c_{2,w}]^{\mathrm{T}} \in P(\alpha_1, \alpha_2) \tag{8}$$

where $P(\alpha_1, \alpha_2) \subseteq \mathbf{R}^6$ is a set given in the following:

$$P(\alpha_1, \alpha_2) = \begin{cases} P_1, \ \alpha_2 = 1; \\ P_2, \ \alpha_1 \in \{5, 6, 7\} \text{ or } \alpha_2 \in \{3, 4\} \end{cases} \tag{9}$$

and $\mathcal{F}_i(c_d, c_w, q; \Theta_i)$ is the Model Accuracy Predictor (MPA). For ease of calculation, a polynomial form of MPA in Ref. [19] is used,

$$\mathcal{F}_i(c_d, c_w, q; \Theta_i) = \sum_{\lambda, j, k=0}^{\kappa} \theta_{p,j,k}^{\{i\}} c_d^{\lambda} c_w^{j} q^k, \quad i = 1, 2 \tag{10}$$

where $\Theta_i \in \mathbf{R}^{(\kappa+1)\times(\kappa+1)\times(\kappa+1)}$ is a tensor, and $\theta_{\lambda,j,k}^{\{i\}}$ is its element.

Subject to satisfying the execution time constraints, the solution of the optimization Formulas (6)–(8) gives the number of quantization bits and pruning ratios of the LDD and BDD models that make the MPA the most accurate, which gives the solution of the case $\alpha = [\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}}$.

In order to realize the stable operation of the expert system, the effective management of knowledge base is indispensable, mainly including the addition, deletion, and modification of cases. If the source case cannot be found to match the current process parameters, after obtaining a solution for the current process parameters through other means, the current process parameters should be edited and added to the knowledge base. It is necessary to edit the current process parameters and add them to the knowledge base, and it is also necessary to add all the information with the current case to the case base. When the expression is not accurate, the case information needs to be deleted or modified.

4.1.3 Case-based reasoning of sequential search

This system adopts case inference based on sequential search, which first obtains a subset of the case base with high similarity to the target case. Then by detecting the similarity of the features corresponding to the target case and the existing cases in this subset, the comparison between the target case and the existing cases is calculated by the weights of different features, and the most matching case is finally selected.

For a given target case $\alpha^0 = [\alpha_1^0, \alpha_2^0, \alpha_3^0]^{\mathrm{T}}$, denote the set consisting of process parameters of all cases in the case base as D. A subset of the case base is obtained by first identifying the case from the case base that is identical to α_1, i.e., $D_1 = D \cap \{[\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}} : \alpha_1 = \alpha_1^0\}$. Then find the same case as α_2 from D_1 and get $D_2 = D_1 \cap \{[\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}} : \alpha_2 = \alpha_2^0\}$. Finally, find the same case as α_3 from D_2, yielding $D_3 = D_2 \cap \{[\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}} : \alpha_3 = \alpha_3^0\}$. Equation (11) calculates the set of cases with the closest distance between the existing cases and the target case, as shown at the bottom of this page, where γ_i denotes the weight of the attribute, $i = 1, 2, 3$; p is a positive integer, which implies that when $p = 2$, it means Euclidean distance is used; $\varnothing$ denotes the empty set.

$$S\left(\alpha^0, D\right) = \begin{cases} D_3, \ D_3 \neq \varnothing; \\ \underset{\alpha = [\alpha_1, \alpha_2, \alpha_3]^T \in D_3}{\arg\min} \left(\left(\alpha_3^0 - \alpha_3\right)^p\right)^{\frac{1}{p}}, \ D_2 \neq \varnothing, \ D_3 = \varnothing; \\ \underset{\alpha = [\alpha_1, \alpha_2, \alpha_3]^T \in D_3}{\arg\min} \left(\sum_{i=2}^{3} \gamma_i \left(\alpha_i^0 - \alpha_i\right)^p\right)^{\frac{1}{p}}, \ D_1 \neq \varnothing, \ D_2 = \varnothing; \\ \underset{\alpha = [\alpha_1, \alpha_2, \alpha_3]^T \in D_3}{\arg\min} \left(\sum_{i=2}^{3} \gamma_i \left(\alpha_i^0 - \alpha_i\right)^p\right)^{\frac{1}{p}}, \ D_1 \neq \varnothing \end{cases} \tag{11}$$

The most matching case is selected from $S(\alpha^0, D)$. When $D_3 \neq \varnothing$, $S(\alpha^0, D)$ has only one element, which is the most matching case; when $D_3 = \varnothing$, the element in $S(\alpha^0, D)$ may not be unique. At this point, the case with the smallest α_3 is chosen as the most matching case, i.e., there is

$$\alpha^* = \underset{\alpha = [\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}} \in S(\alpha^0, D)}{\arg\min} \{\alpha_3\} \tag{12}$$

Given $\varepsilon > 0$, define the set as follows:

$$\Phi = \left\{[\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}} : \alpha_1 = \alpha_1^*, \alpha_2 = \alpha_2^*, \alpha_3 \in [\alpha_3^*, \alpha_3^*(1+\varepsilon)]\right\} \tag{13}$$

The set of lightweight parameters corresponding to the elements in set Φ^0 is called the set of candidate lightweight parameters for the target case $\alpha^0 = [\alpha_1^0, \alpha_2^0, \alpha_3^0]^{\mathrm{T}}$, which is the output of the expert system.

The reasoning of the expert system and how it works is shown in Fig. 5.

4.2 Library of lightweight models

The number of cases in the expert knowledge base is often huge, and they need to be clustered to obtain a smaller set of representative lightweight parameters from which a model library can be built for online scheduling. In addition, the criterion for selecting models from the model library is the nearest neighbor matching principle.

4.2.1 Construction of lightweight model library

Suppose that there are m cases in the case base, and the set of corresponding solutions is denoted as Ψ. There are K models in the model library to be built. In this paper, we use the K-means algorithm to divide Ψ into K sets. The center point of each set is used as the lightweight parameter of the model in the model library.

The steps of K-means clustering are as follows:

Initialization: give a set of samples $\Psi=\{\varphi_1,\varphi_2,\ldots,\varphi_m\}$, and the number of clusters K.

(1) The first step is to randomly select K samples in the sample set as the initial mean vector $\mu_1,\mu_2,\ldots,\mu_K$.

(2) Define the loss function $J(c,\mu)=\min\sum_{i=1}^{m}\|\varphi_i-\mu_{c_i}\|$, where $\|\cdot\|$ denotes the L_2 norm of a vector or matrix.

(3) Determine the cluster labeling of φ_j based on the closest mean vector $\lambda_j=\arg\min_{i\in\{1,2,\ldots,K\}} d_{ji}$, and assign the sample φ_j to the corresponding cluster $C_{\lambda_j}=C_{\lambda_j}\bigcup\{x_j\}$.

(4) Compute and follow the update of the mean vector $\mu'_i=\frac{1}{\#(C_i)}\sum_{x\in C_i}x$, where $\#(\cdot)$ denotes the number of elements in a set.

(5) Until the mean vector is not iteratively updated, it is the clustering result.

After obtaining the lightweight parameters in the model library to be built, the pre-trained models $M_0^{\{1\}}$ and $M_0^{\{2\}}$ of the LDD model and the BDD model are pruned to the targets $c_{1,d}$, $c_{1,w}$, $c_{2,d}$, and $c_{2,w}$ by filter-level pruning and layer-level pruning. The model is then fine-tuned with images of size q_1 and q_2, as shown in Fig. 6. Throughout the pruning process, both layer-first pruning and filter-first pruning are feasible and result in the same pruned model[19].

Remark 1 Regarding the choice of model library size, theoretically the larger the size of the model library, the better the method will work, which will be verified in the experimental part. In practice, we need

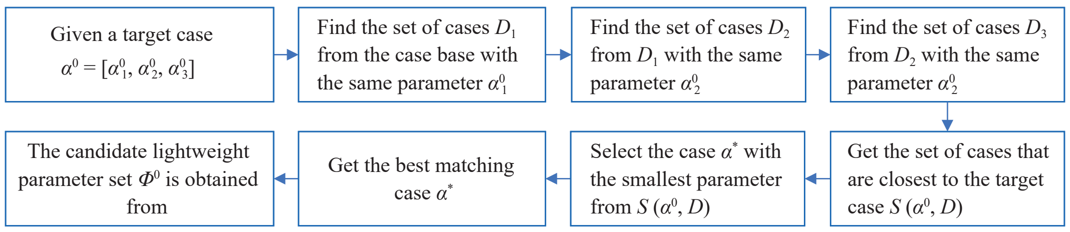

Given a target case $\alpha^0=[\alpha_1^0,\alpha_2^0,\alpha_3^0]$
Find the set of cases D_1 from the case base with the same parameter α_1^0
Find the set of cases D_2 from D_1 with the same parameter α_2^0
Find the set of cases D_3 from D_2 with the same parameter α_2^0
Get the set of cases that are closest to the target case $S(\alpha^0, D)$
Select the case α^* with the smallest parameter from $S(\alpha^0, D)$
Get the best matching case α^*
The candidate lightweight parameter set Φ^0 is obtained from

Fig. 5 Case reasoning process.

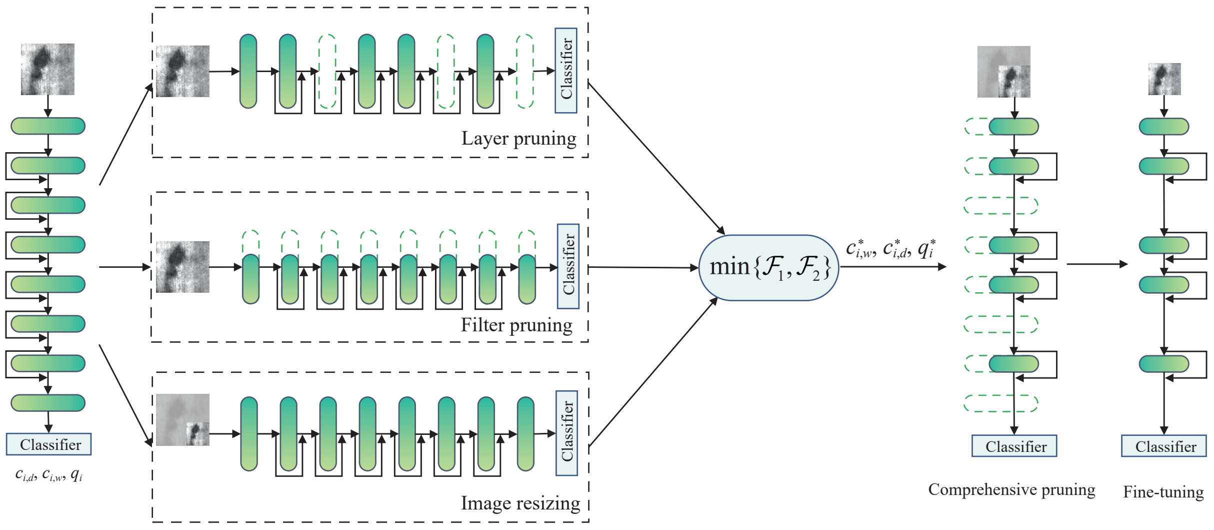


Fig. 6 Model compression schematic.

to consider the limitations of equipment and other factors. The larger the model library, the more storage space is required, which may exceed the storage capacity of the device. In addition, larger model libraries require larger computational resources, which may lead to slower computation and thus affect the performance of real-time performance. Therefore, when choosing the size of the model library, we need to weigh the situation on a case-by-case basis. If the storage capacity and computational resources of the device are sufficient, then we can choose a larger model library, i.e., a larger value of K, to obtain better performance. Otherwise, a smaller value of K.

4.2.2 Nearest neighbor matching

Since various lightweight parameters cannot be fully covered in the model library, for a given lightweight parameter φ^*, how to pick the corresponding model from the K models in the model library? In this paper, we use the nearest neighbor matching method.

Introduce the distance $d(\varphi^*, \mu_i)$, given by the following equation:

$$d(\varphi^*, \mu_i) = \|\varphi^* - \mu_i\|,\ i = 1, 2, \ldots, K \tag{14}$$

By calculating the distance between the target vector φ^* and μ_i, and comparing it, the model corresponding to the lightweight parameter in the case base closest to the target vector is selected as the target model. That is, one can get the lightweight parameter μ^* for the model that best matches φ^* in the model library by solving

$$\mu^* = \mathop{\arg\min}_{1 \leqslant i \leqslant K} d(\varphi^*, \mu_i) \tag{15}$$

4.3 Working condition-driven adaptive model lightweight mechanism

For a production task, there is a total of N steel plate pictures with a total limited detection time of T. The working condition parameter of the current picture (k-th picture) is denoted as $\beta_k = [\beta_{k,1}, \beta_{k,2}, \beta_{k,3}]^{\mathrm{T}}$, where $\beta_{k,1}$, $\beta_{k,2}$, and $\beta_{k,3}$ denote the billet number, furnace number, and plate number, respectively. The classification result is denoted as R_k, and the execution time of the model is denoted as τ_k, $k = 1, 2, \ldots, N$.

The values of R_k are various combinations of defects, i.e., there are $R_k \in \Omega$, where Ω denotes the set consisting of all subsets of the set {1, 2, 3, 4, 5}, and the meanings of the numbers 1, 2, 3, 4, 5 are given by the Table 3. As shown in Fig. 7, $(p_{k1}, p_{k2}) = (x/2, y/2)$, which denotes the center of the defect coordinates. The original map is centered at $(p^0_{k1}, p^0_{k2}) = (x^0/2, y^0/2)$.

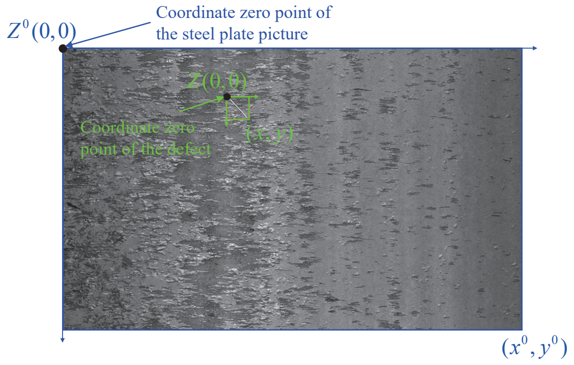

Coordinate zero point of the steel plate picture
$Z^0(0,0)$
$Z(0,0)$
Coordinate zero point of the defect
(x, y)
(x^0, y^0)

Fig. 7 Coordinates of defects.

4.3.1 Analysis of detection results in relation to working conditions

Based on the detection result, they are analyzed in the following three cases.

Case I: 1∈R_k

If the crack appears at the end of the plate, then we scrutinize the next plate. That is, if $p_{k2}/p^0_{k2} \geqslant 0.7$, then focus on detecting cracks. At this point, it means that the accuracy of the LDD model is required to be higher, so the number of quantization bits of the LDD model in the detection algorithm of the next picture (($k+1$)-th picture) should be no less than the number of quantization bits of the BDD model, and the pruning ratio is no greater than the pruning ratio of the BDD model, i.e., $q_1 \geqslant q_2$, $c_{1,d} \leqslant c_{2,d}$, and $c_{1,w} \leqslant c_{2,w}$.

Case Ⅱ: 2∈R_k or 5∈R_k

At this time, the probability of inclusions and Impurity pressed-in defect in the head and tail billet number and the same furnace number rolled parts is higher, and it is necessary to focus on checking. That is, if $\beta_{k+1,1} = 1$, $\beta_{k+1,1} = \overline{\beta}$, or $\beta_{k+1,2} = \beta_{k,2}$, then it is necessary to focus on checking the inclusions, which means that the accuracy of the BDD model is required to be a little higher, so in the next step of the detection algorithm for the picture the number of quantization bits of the BDD model should be not less than the number of quantization bits of the LDD model, and the pruning ratio is not greater than the pruning ratio of the LDD model, i.e., $q_1 \leqslant q_2$, $c_{1,d} \geqslant c_{2,d}$, and $c_{1,w} \geqslant c_{2,w}$. Here, the value of $\overline{\beta}$ is determined by the size of the steelmaking furnace and indicates the tail billet number.

Case Ⅲ: 3∈R_k or 4∈R_k

In this case, the probability of scab and double skin of steel plates with the same plate number is higher, and the detection needs to be focused. That is, if $\beta_{k+1,3} = \beta_{k,3}$, then it is necessary to focus on checking the scab and double skin, which means that the accuracy of the BDD model is required to be higher, so in the next step of the detection algorithm the number of quantization bits of the BDD model should be not less than the number of quantization bits of the LDD model, and the pruning ratio is not greater than the pruning ratio of the LDD model, i.e., $q_1 \leqslant q_2$, $c_{1,d} \geqslant c_{2,d}$, and $c_{1,w} \geqslant c_{2,w}$.

4.3.2 Optimization modeling of lightweight parameters

Let $\Delta_k = \tau_k - \tau_k^0$ denote the difference between the actual and theoretical execution time of the current model, which indicates the impact of factors such as the network environment on the model performance.

From the analysis above, according to the detection results of the current picture and the process parameters and working condition parameters of the next picture, the lightweight parameter selection of the detection model for the $(k+1)$-th picture can be described as the optimization Formulas (16)–(19) with constraints,as shown at the bottom of this paper, where $\mathcal{F}_i\,(c_d,c_w,q;\Theta_i)$ is the MAP, given by Eq. (10), $i=1,2$; Φ_{k+1} denotes the set of candidate lightweight parameters given by the expert system based on the process parameters of the $(k+1)$-th picture; $h\,(r;P_1,P_2)$ is defined as:

$$\max_{[q_1,\,q_2,\,c_{1,d},\,c_{1,w},\,c_{2,d},\,c_{2,w}]^{\mathrm{T}}\in\Phi_{k+1}} \min\{\mathcal{F}_1\,(c_{1,d},\ c_{2,w},\ q_1;\ \Theta_1),\ \mathcal{F}_2\,(c_{2,d},\ c_{2,w},\ q_2;\ \Theta_2)\} \tag{16}$$

$$\text{s.t.},\ \tau_1\,(c_{1,d},\ c_{1,w},\ q_1)+\tau_2\,(c_{2,d},\ c_{2,w},\ q_2)\leqslant \alpha_3-\Delta_k \tag{17}$$

$$\tau_1\,(c_{1,d},\ c_{1,w},\ q_1)+\tau_2\,(c_{2,d},\ c_{2,w},\ q_2)\leqslant \frac{1}{N-k}\left(T-\sum_{i=1}^{k}\tau_k\right) \tag{18}$$

$$[q_1,\ q_2,\ c_{1,d},\ c_{1,w},\ c_{2,d},\ c_{2,w}]^{\mathrm{T}}\in h(r;\ P_1,\ P_2) \tag{19}$$

$$h\,(r;P_1,P_2)=\begin{cases}P_1, & 1\in R_k;\\ P_2, & 1\notin R_k\end{cases} \tag{20}$$

The sets P_1 and P_2 in the above equation are defined by Eqs. (3) and (4).

The purpose of building this optimization problem is to find a set of lightening parameters to maximize the accuracy of the model, and since the LDD and BDD models run sequentially and need to be lightened for both models, so in terms of accuracy, it is necessary to find the total maximum accuracy, i.e., to find the maximum of the smallest of the minimum values in the MAP of the two models, which gives Formula (16). In the process of solving the optimization problem, it is first necessary to ensure that the processing time of each image is up to the standard, i.e., to constrain the time of a single image, which gives Formula (17). Secondly, it is a must to ensure that the entire inspection process is within the required time, i.e., to constrain the overall time of inspection throughout the production task, which gives Formula (18). Finally, the subsequent parameter selection needs to be restricted based on the detection results of the working conditions and the previous picture, which gives Formula (19).

4.3.3 Solving of the optimal lightweight parameter

Particle Swarm Optimization (PSO) is a branch of evolutionary algorithms, which performs function solving by simulating the foraging behavior of birds, and has the advantages of fast convergence, few parameters, and easy implementation, and is often used to solve nonlinear problems[20]. Moreover, PSO does not rely on the derivative information of the objective function, which makes it still effective in dealing with some non-smooth and non-derivable problems. At the same time, the performance of the PSO is to some extent less sensitive to the choice of parameters and relatively easy to adjust, which makes the algorithm more flexible in practical applications.

Since PSO algorithm is usually used to solve the optimization problem for the minimum value, the objective function of the above problem is rewritten as

$$\min\left\{-\min\{\mathcal{F}_1(c_{1,d},\ c_{2,w},\ q_1;\Theta_1),\ \mathcal{F}_2(c_{2,d},\ c_{2,w},\ q_2;\Theta_2)\}\right\} \tag{21}$$

Secondly, there are two inequality constraints in the optimization problem, to apply them to the particle swarm algorithm, it is necessary to transform the constraint problem into an unconstrained problem with the help of a penalty function, and here the constraints of Formulas (17) and (18) are transformed into penalty terms,

$$e_1=\sigma_1(\max\{0,\tau_1(c_{1,d},\ c_{1,w},\ q_1)+\tau_2(c_{2,d},\ c_{2,w},\ q_2)-(\alpha_3-\Delta_k\}) \tag{21}$$

$$e_2=\sigma_2\left(\max\left\{0,\tau_1(c_{1,d},\ c_{1,w},\ q_1)+\tau_2(c_{2,d},\ c_{2,w},\ q_2)\left(\frac{1}{N-k}\left(T-\sum_{i=1}^{k}\tau_k\right)\right)\right\}\right) \tag{23}$$

where σ_i denotes the penalty factor, $i = 1, 2$. During the iteration process, the penalty term forces the iteration points to approach the feasible domain by applying a penalty to the infeasible points. Adding the penalty term to the original objective function, the final objective function is obtained as

$$\min\Big\{-\min\{\mathcal{F}_1(c_{1,d},\ c_{2,w},\ q_1;\Theta_1),\\ \mathcal{F}_2(c_{2,d},\ c_{2,w},\ q_2;\Theta_2)\} + e_1 + e_2\Big\} \tag{22}$$

Finally, the range $[q_1, q_2, c_{1,d}, c_{1,w}, c_{2,d}, c_{2,w}]^{\mathrm{T}} \in \Phi_{k+1}$, $[q_1, q_2, c_{1,d}, c_{1,w}, c_{2,d}, c_{2,w}]^{\mathrm{T}} \in h\,(r; P_1, P_2)$ of the parameter can be transformed into the positional boundaries of the particle during the search process, i.e., the range restriction of the parameter is realized by specifying the upper and lower bounds of the dimensions of the particle in each direction.

After the above series of operations are completed, the velocity v_i^d and position x_i^d of the particle are updated to find the optimal solution of the objective function by

$$v_i^d = w \times v_i^d + z_1 \times \mathrm{rand}_1^d \times \left(p\,\mathrm{Best}_i^d - x_i^d\right) +\\ z_2 \times \mathrm{rand}_2^d \times \left(g\,\mathrm{Best}_i^d - x_i^d\right) \tag{23}$$

$$x_i^d = x_i^d + v_i^d \tag{24}$$

where w is the weight inertia, z_1 and z_2 are the acceleration coefficients, and rand_1^d and rand_2^d are two random numbers in $[0, 1]$.

4.3.4 Algorithm implementation

The implementation of the model adaptive lightweight mechanism given above can be described as follows. For a given production task, the process parameters and time constraints must be analyzed first, so as to set the size of the model library, and then the lightweight parameter set corresponding to the model library is obtained according to the expert system and clustering. According to this, the model library is obtained by quantizing and pruning the pre-trained model. In the process of task execution, according to the working conditions, the optimization model of the algorithm's lightweight parameters is established, and the optimal solution is fed back to the model library in real time for algorithm selection and scheduling. While adapting to the environment, the accuracy and effectiveness are taken into account.

Specifically, when the real-time detection of steel plate surface is carried out, the following steps are required:

Sept 1: Input the picture of the steel plate to be detected, as well as its process parameters and working condition parameters.

Sept 2: The expert system selects and outputs a candidate lightweight parameter set according to the process parameters.

Sept 3: Based on the detection results of the current picture and the working condition parameters of the next picture, and taking into account the detection time limitation of the whole production task, give the optimal lightweight model parameters of the next picture from the candidate lightweight parameter set.

Sept 4: Based on the nearest neighbor matching principle, select the detection model for the next picture from the model library and output the detection results.

Sept 5: Repeat Septs 3 and 4.

The above can be summarized as Algorithm 1.

Remark 2 This paper only considers five kinds of the most common defects. The methodology can be generalized to more than five kinds. Accordingly, the method proposed in this paper is re-used to build a new expert knowledge base, a new model library, and a new optimization model to find the optimal lightweight parameters, to achieve adaptive model lightweight.

Algorithm 1 Adaptive model lightweight mechanism

Input: A sequence of steel plate pictures (total number of steel plate pictures N, total detection time limit T)

Output: Results of classification of surface defects on steel plates

```
1. Give the initial value $R_0$, $(p_{0,1}, p_{0,2})$, and $\tau_0$;
2. for $k = 1, 2, \ldots, N$ do
3.     Obtain the $k$-th process parameters $\alpha^{\{k\}}$ of the steel platepicture and input them into the expert system;
4.     Perform sequential search-based case-based reasoning, give acandidate lightweight parameter set $\varphi^{\{0,k\}}$ by the expert system;
5.     Get the $k$-th working condition parameter $\beta^{\{k\}}$;
6.     Based on the detection result $R_{k-1}$ of the $(k-1)$-th picture, the defect coordinates $(p_{k-1,1}, p_{k-1,2})$, and the execution time $\tau_{k-1}$, the particle swarm algorithm is used to give the model lightweight parameter $\varphi_k^*$ for processing the $k$-th picture;
7.     Based on the nearest neighbor matching principle, the lightweight model that best matches $\varphi_k^*$ is selected from the model library;
8.     Process the $k$-th steel plate picture;
9. end for
10. Return: Defect classification results
```

5 Experiment

This section illustrates the effectiveness of the method proposed in this paper by comparing it with the detection method of constant parameter compressed model. Further, it examines the influence of factors, such as the size of the model library, whether to use an expert system or not, whether to consider single picture constraint time or not, and whether to consider total time constraint on the method of this paper.

5.1 Experiment settings

There are existing LDD models for line defect detection and BDD models for block defect detection, which are trained according to the following process. 3000 grayscale pictures of hot rolled steel plates from Nanjing Iron and Steel Group Co., Ltd. are collected, of which crack defects contain 1000 samples and the rest (inclusions, scars, heavy skins, and foreign objects pressed in) contain 500 samples for each type of defects. The dataset of line (crack) and block (the rest of the defects) defects is divided into training set and test set according to 4:1 (four to one), respectively. The training set is labeled with line defects and block defects. Two ResNet network models are trained separately, and the optimal results obtained from the training set are applied to the test set. Thus, the LDD model and the BDD model are obtained.

There is an existing steel plate production task to be inspected for surface quality defects, where there are $N = 1000$ pictures of steel plates, and the total detection time is limited to $T = 1000$ s, which implies that the processing of a single picture is limited to 1 s. The process parameters and the working condition parameters are given in Table 6. The defect distribution is given by Table 7.

Pre-experiments are conducted to obtain the pruning ratios and quantization bits for the constant lightweight model. Firstly, the pre-trained experiments of the pruning model are conducted, and the results of the acceleration ratio and accuracy loss of the pruning model with pruning ratios of 0.1, 0.2, 0.3, 0.4, and 0.5 are obtained, as shown in Fig. 8. Secondly, the pre-trained experiments of the quantization model are conducted, and the results of the acceleration ratio and accuracy loss of the quantization models, with 16 bits, 12 bits, 6 bits, and 4 bits are obtained, as shown in Fig. 9. The first is the detection method of constant parameter compressed model, i.e., the same compressed model is used for all pictures. The other is to use adaptive model compression detection method proposed by this paper. Further, we compare and analyze the results of these two methods.

For the first method, we introduce the acceleration ratio $A(M,M^*) = S/S^*$ for the selection of the lightweight parameters, where M is the original model, M^* is the compressed model, S is the time spent running the original model, and S^* is the time spent running the compressed model. The LDD model and BDD model set the same pruning ratio and number of quantization bits, and the depth pruning ratio is set to be the same as the width pruning ratio. As can be seen from Figs. 8 and 9, it is better to set the number of quantization bits and the pruning ratio to 6 and 0.3, respectively, for both the LDD model and the BDD model.

Table 6 Process parameters and working conditions.

α_1	α_2	α_3(s)	β_1	β_2	β_3
Engineering machinery steel	Wide gauge	1	1	1–37	1–150
Bridge steel	Thick gauge	1	1	38–51	151–200
Pipeline steel	Thin gauge	1	1	52–64	201–250
Wind tower steel	Extra long gauge	1	1	65–75	251–290
Bridge steel	Thick gauge	1	1	76–80	291–305
Engineering machinery steel	Common gauge	1	1	81–107	306–410
Bridge steel	Thick gauge	1	1	108–125	411–500
Market circulation steel	Common gauge	1	2	126–181	501–720
Engineering machinery steel	Common gauge	1	2	182–190	721–750
Medium carbon Steel	Thick gauge	1	2	191–201	751–790
Pipeline steel	Thin gauge	1	2	202–212	791–830
Wind tower steel	Extra long gauge	1	2	213–242	831–950
Bridge steel	Thick gauge	1	2	243–250	951–1000

Table 7 Defect Distribution.

Steel plate number	Defect type	Number of defects	Defect type	Number of defects
32	3	2	5	1
128	2	1	5	1
188	1	1	2	2
189	1	1	2	1
232	2	2	4	1
272	2	1	3	2
293	1	1	2	1
309	2	1	3	1
420	1	2	4	2
516	2	1	3	2
596	2	2	5	2
621	2	1	–	–
719	5	2	–	–
726	5	2	–	–
753	1	1	–	–
792	2	1	4	1
819	4	1	–	–
846	3	1	–	–
936	3	2	–	–
997	1	2	4	2

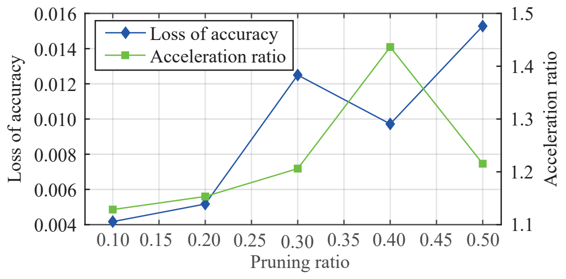


Fig. 8 Effect of quantile on model compression.

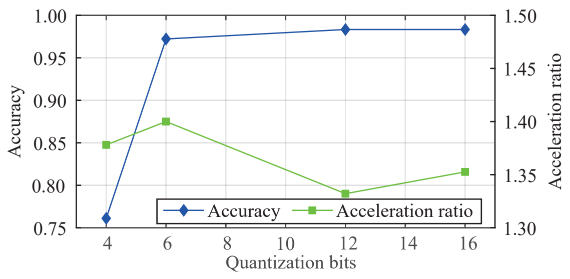


Fig. 9 Effect of pruning scale on model compression.

The accuracy rate, false detection rate, total time of detection, and maximum time of single detection are used as indicators for comparative analysis, in which the accuracy rate and false detection rate are defined as

$$\text{Accuracy rate} = \frac{\text{Number of correct defects detected}}{\text{Total number of defects}},$$

$$\text{False detection rate} = \frac{\text{Number of faulty defects detected}}{\text{Total number of defects}}.$$

5.2 Experiment results

The results of the detection method of constant parameter compressed model are given by Table 8.

In the adaptive model compression detection method, the size of the model library is set to $K = 15$. The experiment is executed according to the flow of Algorithm 1, and the experimental results are obtained, shown in Table 9.

5.2.1 Comparison of the two detection methods

A comparison of the values detection time of the two methods is shown in Fig.10a. As a whole, the detection time under the detection method of constant parameter compressed model is almost always higher than that of the adaptive model compression detection method. Partially, the former sometimes exceeds the limited time and fluctuates a lot, while the values detection time of the adaptive model compression detection method are all within the limits and fluctuate less. This shows that the method in this paper has significant advantages.

Table 8 Defect detection results based on the detection method of constant parameter compressed model.

Steel plate number	Defect type	Number of defects	Defect type	Number of defects	Detection time (s)
32	3	1	5	1	0.859
128	2	1	5	1	0.932
188	1	1	1	2	1.219
189	1	1	1	1	0.971
232	2	2	4	1	1.121
272	3	2	–	–	0.932
293	1	1	1	1	0.945
309	2	1	3	1	1.245
420	1	2	4	2	0.941
516	2	1	3	2	1.141
596	2	2	5	2	0.959
621	2	1	–	–	0.859
719	5	2	3	1	1.147
726	5	1	–	–	0.823
753	1	1	–	–	0.865
792	2	2	4	1	1.112
819	4	1	–	–	0.796
846	3	1	–	–	0.878
936	3	2	–	–	1.047
997	1	2	4	2	1.216

Table 9 Detection results based on the adaptive model compression detection method.

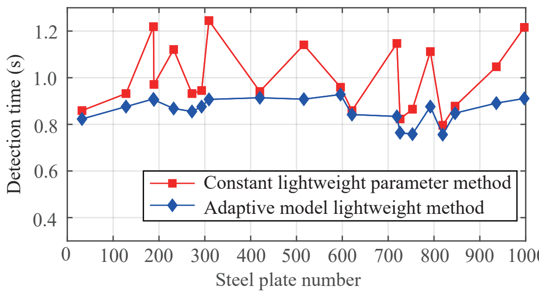
Steel plate number	Defect type	Number of defects	Defect type	Number of defects	Detection time (s)
32	3	2	5	1	0.823
128	2	1	5	1	0.876
188	1	1	1	2	0.909
189	1	1	1	1	0.905
232	2	2	4	1	0.868
272	2	1	3	2	0.855
293	1	1	1	1	0.876
309	2	1	3	1	0.907
420	1	2	4	2	0.914
516	2	1	3	2	0.908
596	2	2	5	2	0.928
621	2	1	–	–	0.842
719	5	2	–	–	0.834
726	5	2	–	–	0.764
753	1	1	–	–	0.758
792	2	2	4	1	0.876
819	4	1	–	–	0.756
846	3	1	–	–	0.848
936	3	2	–	–	0.891
997	1	2	4	2	0.911

In order to compare the gap between the two methods in terms of detection efficiency and deadline satisfaction rate, Fig. 10b gives the box plot, with

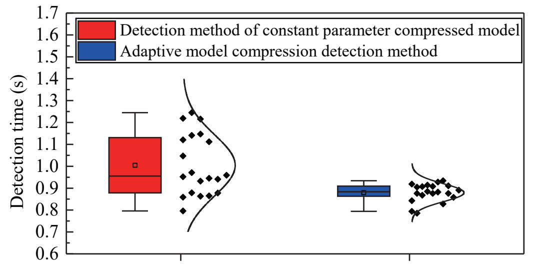
$$\text{Deadline satisfaction ratio} = \frac{\text{Number of pictures not exceeding the limited time}}{\text{Total number of pictures}}.$$

It can be seen that the average time of the adaptive model compression detection method is in the vicinity of 0.9 s, while the average time of the detection method of constant parameter compressed model has exceeded 1 s, and the deadline satisfaction rate of the adaptive model compression detection method has been improved by 36.5% in comparison with that of the detection method of constant parameter compressed model. This is because in the detection method of constant parameter compressed model, there is no consideration of the process and working conditions, and it is not possible to dynamically decide the amount of compression on the model, while the adaptive model compression detection method uses expert experience to assist in these bases, which increases the detection efficiency to a large extent.

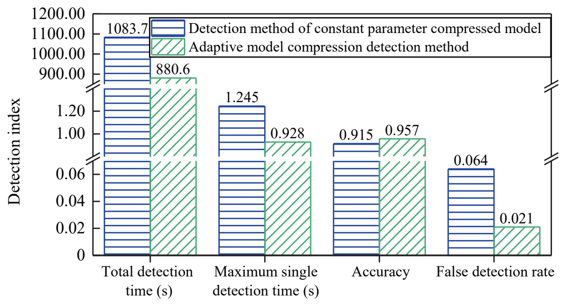
Figure 10c shows the comparison of the total detection time and the maximum time for single sheet detection, the accuracy rate, and the false detection rate, where the adaptive model compression detection method improves the detection accuracy by 4.2% and reduces the false detection rate by 4.3%. The total detection time is reduced by 144.9 s and the maximum single detection time is reduced by 0.317 s. This is because, when the expert system is introduced to select the initial lightweight parameters for detection, a more appropriate lightweight model can be selected at the beginning of detection, thus improving the accuracy at the beginning of detection, thereby increasing the accuracy at the start of the detection and reducing the detection time. In the condition-driven adaptive model lightweight, the optimization of the accuracy under the constraints of the overall time and the maximum detection time of the sheet results in higher detection accuracy and faster detection speed.



(a) Detection time



(b) Box plot of detection time



(c) Detection index

Fig. 10 Comparison between the two detection methods.

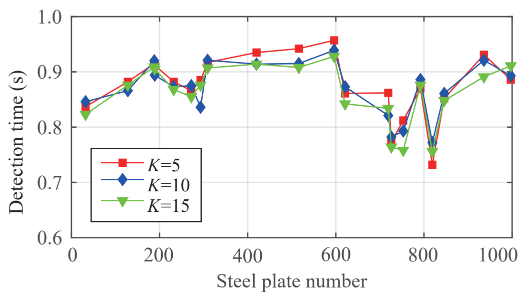
5.2.2 Influence of the main factors in the method of this paper

(1) Influence of the model library size

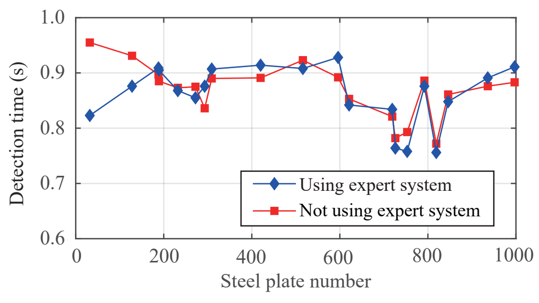
To validate the effect of the size of the model library in the proposed method, a comparison of three sizes of $K = 5, 10$, and 15 is carried out. The results are shown in Figs. 11a and 11e. We can see that the more the number of models in the model library, the smaller the total time of detection and the maximum time of single detection, the accuracy is also higher, which improves the detection speed while ensuring the accuracy and reducing the false detection rate, i.e., when the model library is built more finely, its detection speed and accuracy are higher. This is because that the more models are included in the model library, the more models can be called, i.e., the higher the utilization rate of the expert experience, and thus the speed and accuracy of the steel plate picture detection is increased.

(2) Comparison of the use of expert systems or not

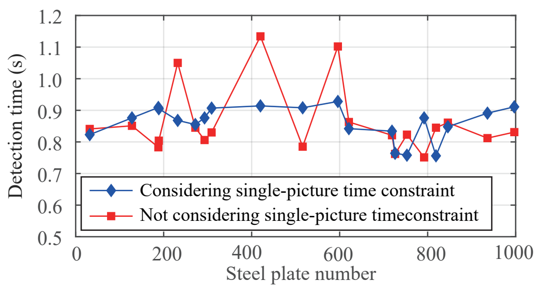
To verify the effect of an expert system in the proposed method, a comparison of using an expert system or not is carried out. The results are shown in Figs.11b and 11f, which shows compared with the case



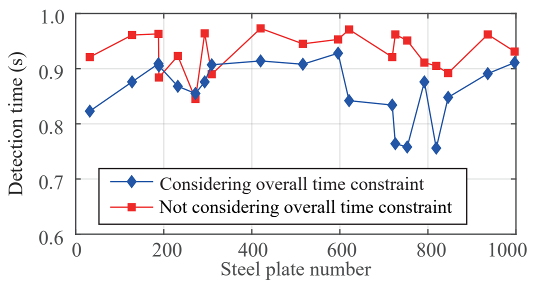
(a) Detection time of different model libraries



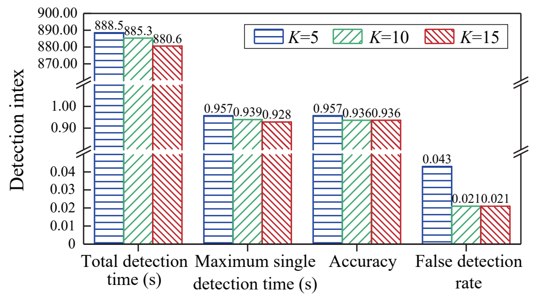
(b) Detection time of using expert system or not



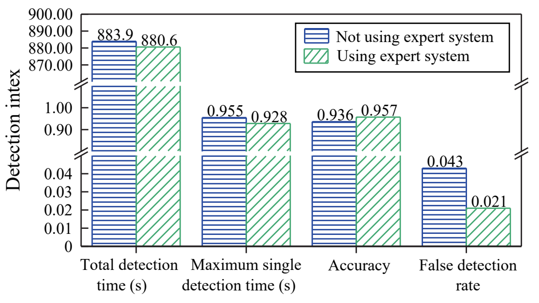
(c) Detection time of considering a single picture time constraint or not



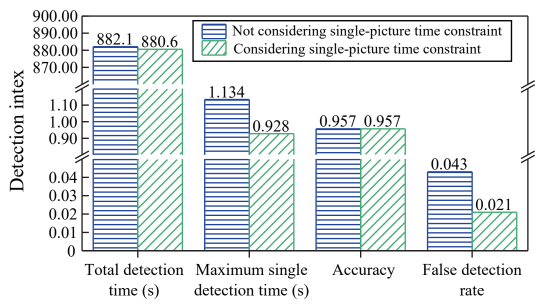
(d) Detection time of considering overall time constraint or not



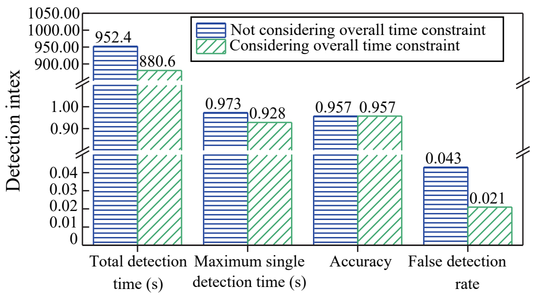
(e) Indexes of different model libraries



(f) Indexes of using expert system or not



(g) Indexes of considering a single picture time constraint or not



(h) Indexes of considering a overall time constraint or not

Fig. 11 Detection performance under the influence of the main factors on adaptive model compression detection method.

without expert system, the result has 2.1% increase in the accuracy rate, 2.2% reduction in false detection rate, 0.027 s reduction in maximum detection time for a single sheet, and 3.2 s reduction in total time when the expert system is used. This is because that the expert system is to get the initial lightweight parameters, when the expert system is not used, the initial value of the lightweight parameters may be improperly selected, which leads to a longer detection time for the first few steel sheet pictures, and the overall time is extended, and the accuracy and false detection rate perform poorly accordingly.

(3) Comparison of considering single-picture time constraint or not in the lightweight parameter optimization

To verify the effect of single picture time constraint in the lightweight parameter optimization, a comparison of considering it or not is conducted. The results are shown in Figs.11c and 11g. When the single picture time constraint is considered in the adaptive model compression detection method, the total detection time and the maximum time of single detection are reduced by 1.5 s and 0.206 s, respectively, and the false detection rate is reduced by 2.2% under the same accuracy. This is because, if the single picture time constraint is not considered, to improve the accuracy rate, it will inevitably lead to the lightweight model selected for a certain picture taking longer time to exceed the single one's limited time, but the total time can be guaranteed due to the overall time constraint.

(4) Comparison of considering overall time constraint or not in the lightweight parameter optimization

To verify the effect of the overall time constraint in the proposed method on the detection performance, a comparison of considering the overall time constraint or not is carried out. The results are shown in Figs.11d and 11h. When the overall time constraint is considered, the total detection time and the maximum time of single-picture detection are reduced by 71.8 s and 0.045 s, respectively, and the false-detection rate is reduced by 2.2% with the same accuracy. This is because when the work-driven feedback considers the single picture constraints without considering the overall time constraints, the total detection time is sacrificed to satisfy the higher accuracy rate as well as the time constraints of the single sheet.

6 Related Work

The superior performance of deep learning usually comes at the cost of a large computational effort, but real applications often struggle to meet the demand in terms of timeliness. It has also been shown that many large models are over-parameterized and that there are many redundant parameters or neurons, which can be trimmed down to reduce the size of the model. In recent years, model compression techniques have made significant progress. The main techniques include knowledge distillation, pruning, and quantization. These techniques aim to reduce the size and computational burden of deep learning models in resource-limited environments[21, 22].

Knowledge distillation is a model compression approach to extract knowledge from a larger deep neural network into a smaller network, and there are many excellent knowledge distillation methods available today, such as knowledge distillation based on association of latent representations, knowledge distillation based on auxiliary models, and knowledge distillation methods based on attentional relations[23–25]. Model pruning, also known as model sparsification, is a very widely used model compression method that directly reduces the number of parameters in a model. These include gradient-limited pruning, sensitivity-based pruning, and weak sub-network pruning, among others[26–28]. Model quantization, as one of the general neural network model optimization methods, can reduce the size of the deep neural network model and model inference time, which breeds a variety of quantization methods based on the quantization granularity, bit quantization, and whether the mapping function is linear or not, such as quantization scale reparameterization, non-uniform to uniform quantization, and so on[29–31].

Many effective methods have been bred in the steel plate surface defect detection scenario. Reference [15] applies the target detection technique to steel plate surface defect detection algorithms, and implemented multi-size defect localization and classification with a deformed convolutional augmented backbone network, a feature fusion network with a balanced feature pyramid, and a detector network. Reference [16] proposes a detection method that combines domain adaptation and adaptive convolutional neural network, by introducing additional domain classifiers and constraints on labeling probabilities, as well as normal distribution and quadratic function to optimize the network. functions to optimize the network, enabling cross-domain and cross-task recognition. Reference [17] proposes a deformable convolutional deep learning approach that improves the detection of tiny defects by designing deformable convolutional layers that act as an attention mechanism and merging multiple layered features. Reference [18] proposes a defect detection system that combines a baseline convolutional neural network with a multilayer feature fusion network to achieve strong defect classification capability. The above-proposed detection methods in defect detection on steel plate surfaces are improvements in the network structure, which improve the classification of defects as well as the accuracy of localization, but do not take into account the real-time requirements of this scenario.

In summary, most of existing works focus on high-precision recognition of a given picture and effective compression of a given model. In particular, knowledge distillation, due to its difficult parameterization, is often not directly applicable when facing problems that require a given specific quantization. Due to the diversity and dynamics of steel plate defect detection scenarios, it is difficult to use a single or fixed model compression method to achieve desired results in steel plate surface defect detection. Compared with the existing work, the advantage of this paper's method lies in the adaptability of model compression. It dynamically adapts different lightweight models according to different production tasks and actual working conditions. It dynamically adapts different lightweight models according to different production tasks and actual working conditions, which ensures the timeliness and accuracy of detection while adapting to the environment.

7 Conclusion

To balance the real-time and accuracy of steel plate surface defect detection, this paper proposes an adaptive model lightweight mechanism based on expert knowledge and working conditions. This method can quickly select the appropriate model in the dynamic and changing steel production environment, and maximize the detection accuracy under the time constraint. Based on analyzing the correlation between process parameters, such as steel grade and size, and surface defects on steel plates, an expert knowledge system construction method based on process parameters is proposed for assisting in the selection of lightweight parameters for the algorithm. After clustering the expert knowledge base and establishing an offline lightweight model library, a working condition-driven lightweight parameter optimization method based on detection results is proposed, which greatly improves the detection efficiency while ensuring accuracy. Based on the production tasks, working conditions, and detection results, an optimization model for lightweight parameter selection is established. An online scheduling method based on the model library is proposed to adapt to the changes in complex working conditions. The experimental results show that the method in this paper, compared with the detection method of constant parameter compressed model, makes the total detection time cut down by 23.1%, and the deadline satisfaction ratio increased by 36.5%, while upgrading the accuracy by 4.2% and reducing the false detection rate by 4.3%.

Acknowledgment

This work was supported by the National Key R&D Program of China (No. 2018AAA0100500), the National Natural Science Foundation of China (Nos. 62232004 and 61632008), the Jiangsu Provincial Key Laboratory of Network and Information Security (No. BM2003201), the Key Laboratory of Computer Network and Information Integration of Ministry of Education of China (No. 93K-9), and the Collaborative Innovation Center of Novel Software Technology and Industrialization. We also thank the Big Data Computing Center of Southeast University in China for providing the experiment environment and computing facility.

References

[1] G. Wang, Q. Xiao, M. Guo, and J. Yang, Optimal frequency of AC magnetic flux leakage testing for detecting defect size and orientation in thick steel plates, *IEEE Trans. Magn.*, vol. 57, no. 9, p. 6200708, 2021.

[2] C. Song, J. Chen, Z. Lu, F. Li, and Y. Liu, Steel surface defect detection via deformable convolution and background suppression, *IEEE Trans. Instrum. Meas.*, vol. 72, p. 5017709, 2023.

[3] R. Hao, B. Lu, Y. Cheng, X. Li, and B. Huang, A steel surface defect inspection approach towards smart industrial monitoring, *J. Intell. Manuf.*, vol. 32, no. 7, pp. 1833–1843, 2021.

[4] X. Xiang, Z. Wang, J. Zhang, Y. Xia, P. Chen, and B. Wang, AGCA: An adaptive graph channel attention module for steel surface defect detection, *IEEE Trans. Instrum. Meas.*, vol. 72, p. 5008812, 2023.

[5] D. He, K. Xu, and P. Zhou, Defect detection of hot rolled steels with a new object detection framework called classification priority network, *Comput. Ind. Eng.*, vol. 128, pp. 290–297, 2019.

[6] A. I. Kusuma and Y.-M. Huang, Product quality prediction in pulsed laser cutting of silicon steel sheet using vibration signals and deep neural network, *J. Intell. Manuf.*, vol. 34, no. 4, pp. 1683–1699, 2023.

[7] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, NV, USA, 2016, pp. 770–778.

[8] Y. Liang, J. Li, J. Zhu, R. Du, X. Wu, and B. Chen, A lightweight network for defect detection in nickel-plated punched steel strip images, *IEEE Trans. Instrum. Meas.*, vol. 72, p. 3505515, 2023.

[9] X. Zhao, Y. Chen, J. Guo, and D. Zhao, A spatial-temporal attention model for human trajectory prediction, *IEEE/CAA J. Autom. Sin.*, vol. 7, no. 4, pp. 965–974, 2020.

[10] B. Guo, Y. Wang, S. Zhen, R. Yu, and Z. Su, SPEED: Semantic prior and extremely efficient dilated convolution network for real-time metal surface defects detection, *IEEE Trans. Ind. Inform.*, vol. 19, no. 12, pp. 11380–11390, 2023.

[11] V. Sampath, I. Maurtua, J. J. Aguilar Martín, A. Rivera, J. Molina, and A. Gutierrez, Attention-guided multitask learning for surface defect identification, *IEEE Trans. Ind. Inform.*, vol. 19, no. 9, pp. 9713–9721, 2023.

[12] D. Bang, J. Lee, and H. Shim, Distilling from professors: Enhancing the knowledge distillation of teachers, *Inf. Sci.*, vol. 576, pp. 743–755, 2021.

[13] S.-K. Yeom, P. Seegerer, S. Lapuschkin, A. Binder, S. Wiedemann, K.-R. Müller, and W. Samek, Pruning by explaining: A novel criterion for deep neural network pruning, *Pattern Recognit.*, vol. 115, p. 107899, 2021.

[14] Y. Huang, Y. Hao, J. Xu, and B. Xu, Compressing speaker extraction model with ultra-low precision quantization and knowledge distillation, *Neural Netw.*, vol. 154, pp. 13–21, 2022.

[15] R. Hao, B. Lu, Y. Cheng, X. Li, and B. Huang, A steel surface defect inspection approach towards smart industrial monitoring, *J. Intell. Manuf.*, vol. 32, no. 7, pp. 1833–1843, 2021.

[16] S. Zhang, Q. Zhang, J. Gu, L. Su, K. Li, and M. Pecht, Visual inspection of steel surface defects based on domain adaptation and adaptive convolutional neural network, *Mech. Syst. Signal Process.*, vol. 153, p. 107541, 2021.

[17] Z. Liu, B. Yang, G. Duan, and J. Tan, Visual defect inspection of metal part surface via deformable convolution and concatenate feature pyramid neural networks, *IEEE Trans. Instrum. Meas.*, vol. 69, no. 12, pp. 9681–9694, 2020.

[18] Y. He, K. Song, Q. Meng, and Y. Yan, An end-to-end steel surface defect detection approach via fusing multiple hierarchical features, *IEEE Trans. Instrum. Meas.*, vol. 69, no. 4, pp. 1493–1504, 2020.

[19] W. Wang, M. Chen, S. Zhao, L. Chen, J. Hu, H. Liu, D. Cai, X. He, and W. Liu, Accelerate CNNs from three dimensions: A comprehensive pruning framework, in *Proc. 38th International Conference on Machine Learning*, Virtual Event, 2021, pp. 10717–10726.

[20] J. Kennedy and R. Eberhart, Particle swarm optimization, in *Proc. ICNN'95 - Int. Conf. Neural Networks*, Perth, Australia, 1995, pp. 1942–1948.

[21] Q. Li, C. Li, and H. Chen, Filter pruning via probabilistic model-based optimization for accelerating deep convolutional neural networks, in *Proc. 14th ACM Int. Conf. Web Search and Data Mining*, Virtual Event, 2021. pp. 653–661.

[22] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, Learning structured sparsity in deep neural networks, in *Proc. 30th Conference on Neural Information Processing Systems*, Barcelona, Spain, 2016, pp. 2082–2090.

[23] Y. Liu, W. Zhang, and J. Wang, Adaptive multi-teacher multi-level knowledge distillation, *Neurocomputing*, vol. 415, pp. 106–113, 2020.

[24] M. Gao, Y. Wang, and L. Wan, Residual error based knowledge distillation, *Neurocomputing*, vol. 433, pp. 154–161, 2021.

[25] J. Gou, L. Sun, B. Yu, S. Wan, W. Ou, and Z. Yi, Multilevel attention-based sample correlations for knowledge distillation, *IEEE Trans. Ind. Inform.*, vol. 19, no. 5, pp. 7099–7109, 2023.

[26] Z. Wang, F. Li, G. Shi, X. Xie, and F. Wang, Network pruning using sparse learning and genetic algorithm, *Neurocomputing*, vol. 404, pp. 247–256, 2020.

[27] C. Yang and H. Liu, Channel pruning based on convolutional neural network sensitivity, *Neurocomputing*, vol. 507, pp. 97–106, 2022.

[28] Q. Guo, X.-J. Wu, J. Kittler, and Z. Feng, Weak sub-network pruning for strong and efficient neural networks, *Neural Netw.*, vol. 144, pp. 614–626, 2021.

[29] Y. Liu, W. Zhang, and J. Wang, Zero-shot adversarial quantization, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Nashville, TN, USA, 2021, pp. 1512–1521.

[30] Z. Li, J. Xiao, L. Yang and Q. Gu, Repq-vit: Scale reparameterization for post-training quantization of vision transformers. in *Proc. IEEE/CVF Int. Conf. on Computer Vision* (*ICCV*), Vancouver, Canada, 2023, pp. 17227–17236.

[31] Z. Liu, K. Cheng, D. Huang, E. P. Xing, and Z. Shen, Nonuniform-to-uniform quantization: Towards accurate quantization via generalized straight-through estimation, in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), New Orleans, LA, USA, 2022, pp. 4942–4952.

Maojie Sun received the BEng degree from Northeastern University, China in 1993, and the MEng degree from Nanjing Normal University, China in 1998. He is currently a PhD candidate at School of Computer Science and Engineering, Southeast University, China. His main research interest is artificial intelligence.

Fang Dong received the BEng and MEng degrees in computer science from Nanjing University of Science & Technology, China in 2004 and 2006, respectively, and the PhD degree in computer science from Southeast University, China in 2011. He is currently a professor at Southeast University, China, and the director of the Big Data Computing Center of Southeast University, China. He is a member of both IEEE and ACM, and also serves as the co-chair of ACM Nanjing Chapter, China. His current research interests include edge intelligence, cloud computing, and industrial Internet.

Zhaowu Huang received the BEng degree from Nanjing University of Science and Technology, China in 2018. He is currently a PhD candidate at School of Computer Science and Engineering, Southeast University, China. His main research interest is edge computing.

Junzhou Luo received the BS degree in applied mathematics from Southeast University, China in 1982, and the MEng and PhD degrees in computer network from Southeast University, China in 1992 and 2000, respectively. He is currently a full professor at School of Computer Science and Engineering, Southeast University, China. He is a member of both IEEE and ACM, and co-chair of IEEE SMC Technical Committee on Computer Supported Cooperative Work in Design. His research interests include network security and management, cloud computing, and wireless LAN.