Hindawi
Advances in Multimedia
Volume 2022, Article ID 9595286, 12 pages
https://doi.org/10.1155/2022/9595286

Research Article

Steel Plate Defect Recognition of Deep Neural Network Recognition Based on Space-Time Constraints

Chi Zhang,[1] **Zhiguang Wang**,[2] **Baiting Liu**,[2] **and Wang Xiaolei**[2]

[1]*Department of Industrial Engineering and Management, Peking University, Beijing 100871, China*
[2]*Science and Technology on Special System Simulation Laboratory, Beijing Simulation Center, Beijing 100854, China*

Correspondence should be addressed to Chi Zhang; z.chi@pku.edu.cn

Received 21 December 2021; Revised 21 January 2022; Accepted 5 February 2022; Published 10 March 2022

Academic Editor: Qiangyi Li

In order to improve the effect of real-time defect recognition in steel plate online production, this paper studies the method of steel plate defect recognition based on the deep neural network algorithm based on space-time constraints. Moreover, this paper improves the space-time constraint algorithm, optimizes the encryption structure of the traditional ABE scheme, and obtains a neural network feature recognition method based on space-time constraints. In order to process the massive image data stream generated instantaneously and ensure the real-time performance, accuracy, and stability of the detection system, this paper constructs a distributed parallel computing system structure based on the client/server (CC/S) model to obtain an intelligent recognition system. Through experimental research, it can be seen that the deep neural network recognition system based on space-time constraints proposed in this paper has a good effect in the recognition of steel plate defects.

1. Introduction

In the past few decades, steel plate rolling production technology and product quality have achieved rapid development. With the improvement of equipment and production technology, the dimensional accuracy, plate shape, mechanical properties, and other quality characteristics of the steel plate have been well controlled. However, the main downstream users of steel plates such as automobiles, aerospace, machinery and electronics, home appliances, and architectural decoration have higher and higher requirements for the surface quality of the steel plates, which makes the surface quality problems of the steel plates more and more prominent [1]. In the application, the surface defects of the steel plate affect the deep pressure of the processing, the coating effect of the final product, the electromagnetic characteristics, and the aesthetics. Therefore, whether it is a steel plate manufacturer or a steel plate user, it is necessary to attach great importance to the surface quality inspection of the steel plate [2].

Whether it is automobile production, shipbuilding, machinery manufacturing, or chemical, aerospace, and other industries, steel is used as raw material for processing and production [3]. The development and growth of various industries has made steel's raw material status more and more important, and its demand is also expanding. However, in the actual production environment, the aging of rolling equipment, different processing techniques, and differences in raw materials may affect the quality of the steel plate, resulting in the appearance of various types of defects such as cracks, scars, and scratches on the surface of the steel plate. As a result, steel products are of low quality and poor performance. Therefore, we must attach great importance to the quality inspection of steel products to obtain high-quality steel products. At present, large-scale steel plants in China are mainly using machine inspection and manual inspection as a supplementary inspection method. Although this detection method has largely replaced with the traditional manual visual inspection and improved the labor force, most of the similar automatic defect detection equipment in China mostly rely on foreign technical support, and sufficient funds must be invested in repair and maintenance. For minor defects (such as scratches, etc.), manual inspection is still required, which does not meet the needs of production development. This increases our design to meet our own needs. The necessity of the defect detection system is on demand.

Based on the above analysis, this paper uses a deep neural network algorithm based on space-time constraints to build an intelligent system that can be used for steel plate defect recognition and improve the intelligent detection effect of steel plate defects.

2. Related Work

Literature [4] constructed an online surface inspection device for continuous casting slabs, which combines linear CCD chips with digital image processing components. Among them, the CCD chip enhances the effective pixels of the image while accelerating according to the frame transmission rate, which can effectively detect tiny defects. According to the frame transfer rate, the device can effectively detect tiny defects; literature [5] applies the iLearn learning classifier software system to the detection system and develops the iS2000 defect detection system, which uses two to increase A/D conversion. The CCD camera that automatically corrects the light source unit simultaneously acquires bright and dark areas with more uniform illumination. This device takes into account the impact of uneven illumination and effectively reduces the impact of this factor; literature [6] proposed a detection system, tried the theory of combining neural network classifier and CCD camera, and achieved certain success. Literature [7] developed an intelligent steel plate inspection system, which increased the defect data utilization function compared with the original system.

Although these foreign companies have studied steel plate defect detection systems relatively early and the detection technology is relatively mature, there are still problems such as low image acquisition quality, low system versatility, insufficient integration, and use of defect data, which require further research.

Literature [8] applied surface defect detection technology to the quality inspection of cold-rolled steel plate production and added a DSP image processing system based on the original research, using multiple array CCDs to obtain images. The more serious defects such as heavy skin, holes, and edge cracks are effective. However, due to the limitation of the imaging principle of the area array CCD, the use of multiple area array CCDs will produce a large amount of defect image redundant data, and the price is expensive, so this system has not been widely used; literature [9] developed an online inspection system that uses multiple area array CCDs as imaging modules and can be used for steel plate surface defect detection. The system can analyze and process the collected image data in parallel. Although the system guarantees real-time data processing functions, its versatility is low; literature [10] uses a new LED light source and FPGA embedded processing system to complete the design of a linear CCD steel plate defect detection system, which can store data by itself and function and display the test results on the screen. If there is a situation that does not meet the standard, the alarm function is completed, but as the function increases, it is not conducive to the production and development of the steel plate at the expense of reducing the production speed.

Literature [11] developed a system capable of detecting multiple functions of strip steel. Compared with the mainstream testing equipment in the world at that time, the system was comparable in detection rate and defect resolution, but the system needs to be improved. The point is the information on the surface of the strip can be used. Literature [12] uses the edge of the defect to detect the defect target and analyzes the texture feature through the PHOTO algorithm to determine the defect location, which improves the detection accuracy. However, the system has many detection characteristics and the system response time is slow.

Literature [13] designed a fuzzy neural network classifier. The feature of this classifier is to use a multilayer feedforward neural network model combined with a fuzzy algorithm. The classifier improves the classification accuracy while reducing the number of iterations, but the classifier has limitations in identifying defects; literature [14] designed the LVQ neural network classifier, which uses the "reward-punishment" iterative learning algorithm to speed up the classification speed of the strip surface defect detection system. However, the classification accuracy of this classification method is highly dependent on the amount of data; literature [15] studies genetic algorithm and BP neural network algorithm, using the global optimal characteristics of genetic algorithm and BP neural network local accuracy characteristics. With complementary advantages, a combined classifier is designed. However, whether it is genetic algorithm or BP neural network algorithm, more data is needed, which increases the amount of data processing; literature [16] uses the BP neural network classifier; the difference is that the high-order classifier is added to this classifier. High-order perturbation can prevent the training data from converging to a local extreme value and at the same time can improve the training speed. However, the addition of high-order disturbances has little contribution to the accuracy of defect detection. Literature [17] combined wavelet transform and BP neural network to the steel plate defect classification system, but the defect detection accuracy is about 90%, which needs to be improved. Compared with the traditional BP neural network, the improved neural network classification method has a more obvious classification effect on the trained data, but it often produces errors for the data outside the training library [18]. In practical applications, too large data set samples are not easy to obtain. Therefore, it is of great practical significance to study small sample data.

3. Deep Neural Network Recognition Based on Space-Time Constraints

For the attribute $A_i \in A$ of the user, it means that there are two sets of $(l_{i,j}, l_{i,k})$ and $(l_{i,j}, l_{i,k})$ that, respectively, represent the range of covering user attribute values, which correspond to two cryptographic values $v\{l_i, l_k\}$ and $v\{l'_i, l_k\}$. If $l_{i,j} \leq l'_{i,j}$ and $l_{i,k} \geq l_{i,k}$ exist, then it will be easy to calculate $v\{l_i, l_k\}$ from $v\{l_i, l_k\}$, and vice versa.

In order to construct an MDRDF function for integer comparison of multiple attributes, this paper uses RSA and assumes that $G_{n'}$ is a multiplicative cyclic group, where φ is a generator in the $G_{n'}$ multiplicative cyclic group. $n' = p'q'$ and $\varphi^{n'} = 1$, $\{\lambda_i, \mu_i\}_{A_i \in A}$ is a set of the largest random number in Z_n^*, and each λ_i and μ_i are prime numbers relative to the other elements of the set.

$u = \{l_{i,j}, l_{i,k}\}_{A_i \in A}$ is an upper and lower bound of the attribute value corresponding to all attributes $A_i \in A$, $\psi \longrightarrow U$ is a cryptographic mapping about the order of user U, which is mapped to a value $v\{l_{i,j}, l_{i,k}\}_{A_i \in A}$ that reflects a cryptographic boundary of each attribute, and z is the largest integer value of user U. If $l_{i,j} < l'_{i,j}$ and $l_{i,k} < l'_{i,k}$, this order-preserving cryptographic mapping means that there are two situations:

$$\begin{aligned} v\{l_{i,j}, l_{i,k}\}A_i \in A &= \psi\left(\{l_{i,j}, l_{i,k}\}A_i \in A\right) \leq v\{l'_{i,j}, l_{i,k}\}A_i \in A \\ &= \psi\left(\{l'_{i,j}, l_{i,k}\}A_i \in A\right), \\ v\{l_{i,j}, l_{i,k}\}A_i \in A &= \psi\left(\{l_{i,j}, l_{i,k}\}A_i \in A\right) \leq v\{l_{i,j}, l'_{i,k}\}A_i \in A \\ &= \psi\left(\{l_{i,j}, l'_{i,k}\}_{A_i \in A}\right). \end{aligned} \tag{1}$$

This article defines the formula of this mapping function $\psi(\cdot)$ from user set U to V as follows [19]:

$$\begin{aligned} &v\{l_{i,j}, l_{i,k}\}_{A_i \in A} \longleftarrow \psi\left(\{l_{i,j}, l_{i,k}\}_{A_i \in A}\right) \\ &v\{l_{i,j}, l_{i,k}\}_{A_i \in A} = \psi\left(\{l_{i,j}, l_{i,k}\}_{A_i \in A}\right) \\ &\qquad = \left(\varphi \prod A_i \in A \lambda_i l_{i,j} \mu i^{z - l_{i,k}}\right) \in G_{n'}. \end{aligned} \tag{2}$$

If the following two conditions are met, the function F from $V \longrightarrow v$ is based on the mapping of U to v.

(1) If $l_{i,j} < l'_{i,j}, l_{i,k} \geq l'_{i,k}, \forall A_i \in A$, there are

$$\begin{aligned} v\{l'_{i,j}, l'_{i,k}\}A_i \in A \longleftarrow F\{1_{i,j} \leq l'_{i,j}, l_{i,k} \geq l'_{i,J}\}A_i \in A\left(v\{1_{i,j}, l_{i,k}\}A_i \in A\right) \\ = \left(v\{l'_{i,j}, l'_{i,k}\}A_i\right)\Pi_{A_i} \in A\lambda_i^{1'_{i,j}-1_{i,j}}\mu_i^{1'_{i,k}-1_{i,k}} \\ = \left(\varphi\Pi_{A_i} \in A\lambda_i^{1_{i,j}z-1_{i,k}}\right)\Pi_{A_i} \in A\lambda_i^{1'_{i,j}-1_{i,j}}\mu_i^{1'_{i,k}-1_{i,k}} \\ = \varphi\Pi_{A_i} \in A\lambda_i^{1_{i,j}}\mu_i^{z-1_{i,k}} \in G_{n'}. \end{aligned} \tag{3}$$

(2) If for a certain attribute $A_i \in A$, there is $l_{i,j} > l'_{i,j}$ or $l_{i,k} < l'_{i,k}$, then it is impossible for a polynomial algorithm to derive $v\{l'_{i,j}, l'_{i,k}\}Ai \in A$ at any time [20].

$$F\left(v\left(l_{i,j}, l_{i,k}\right)\right) = v\left(l'_{i,j}, l'_{i,k}\right) \text{ if } \left(l_{i,j}, l_{i,k}\right) \subseteq \left(l'_{i,j}, l'_{i,k}\right). \tag{4}$$

Figure 1 shows that when $(l_{i,j}, l_{i,k}) \subseteq (l_{i,j}, l_{i,k})$, $v\{l_{i,j}, l_{i,k}\}_{Ai \in A}$ is mapped to $v\{l_{i,j}, l_{i,k}\}A_{i \in A}$ through the function F.

In the process of data encryption, this article optimizes the encryption structure of the traditional ABE scheme. The traditional ABE scheme uses the structure of the access tree to encrypt the user's attributes. Based on this, this paper redesigned the structure of the access tree. This paper considers embedding spatiotemporal constraints into the access tree structure and uses the multidimensional distance derivation function (MDRDF) combined with the threshold to determine the legitimacy of the user.

As shown in Figure 2, this paper uses the structure of the access tree to enhance fine-grained access. Among them, each leaf node represents the attributes $A_1, A_2, \ldots, A_n$ owned by the user, and the nonleaf nodes represent logic gates (AND, OR, Threshold). In Figure 2, node x is represented by num_x, and the threshold is represented by k_x.

In Figure 2, we consider embedding spatiotemporal constraints into any nonleaf node. At the same time, $\text{TD}^x_{\{ta,b\}}$ represents the time range constraint related to node x, and $\text{LD}^x_{a,lb}$ represents the location range constraint related to node x. In the structure of the access tree in Figure 2, the child access policy corresponding to node n1 needs to meet "$A1 \wedge A2$", and the child access policy corresponding to node $n2$ needs to meet " + ". If and only if the user wants to access the resource, it needs to meet the attribute requirement of "$(A1 \wedge A2) \vee (A3 \wedge A4)$", and at the same time, his access time and space can be successfully accessed within the scope of $[t_a, t_b][l_a, l_b]$.

It chooses any two generators $g \in G_S, \varphi \in G_{n'}, \omega \in G, \lambda, \mu \in Z_{n'}^*$, so $e(g, \varphi) = 1$, but $e(g, \omega) \neq 1$. At the same time, three hash functions are used: H_0, H_1: $\{0,1\}^* \longrightarrow G_{S'}, H_2$: $G_T \longrightarrow Z_n^*$ and KGC arbitrarily selects two exponents $\alpha \in Z_n^*$ and $\beta \in Z_n^*$ and generates $h = \omega^\beta, \eta = g^{1/\beta}, \varsigma = e(g, \omega)^\alpha$. Publishing system is as follows:

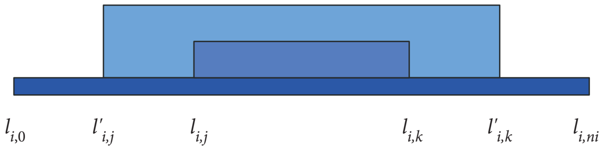

Figure 1: Multidimensional range derivation function.

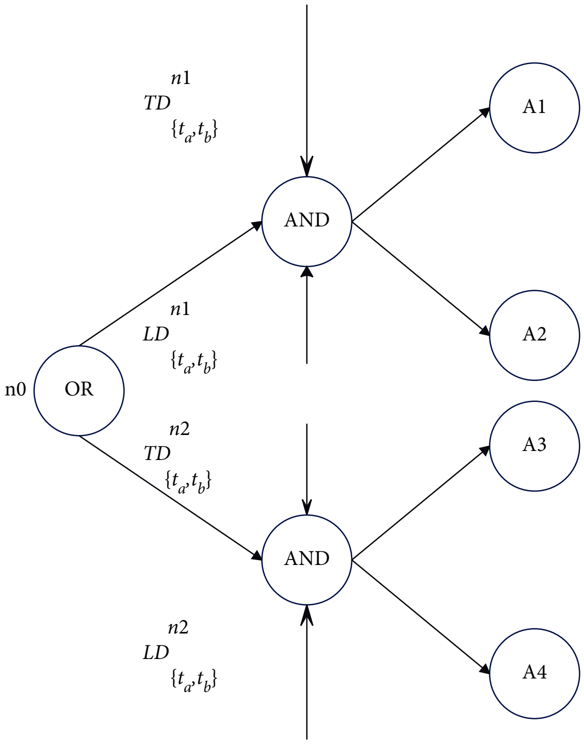

Figure 2: Access policy tree structure.

$$\mathrm{PK} = \left(S_N, g, \eta, \omega, h, \varsigma, \varphi, \lambda, \mu, H_0, H_{1,H_2,}\right). \tag{5}$$

The master key of the system is as follows:

$$\mathrm{MK} = \left(p, q, n', \alpha, \beta\right). \tag{6}$$

For each user there is a set of attributes S_{gid}; the key generation center KGC first selects $u_j \in Z_n^*$ and selects $r_i \in Z_n^*$ for each attribute $i \in S_{\mathrm{gid}}$ belonging to the user set. Then calculate the key as follows:

$$\mathrm{SK}_{\mathrm{attr}} = \left\{ D = g \frac{\alpha + uj}{\beta} H_0(\mathrm{gid}) \frac{uj}{\beta}, D' = \omega^{uj} \right\},$$
$$\forall i \in S_{\mathrm{gid}}\colon D_i = \left\{ \left(gH_0(\mathrm{gid})\right)^{uj} H_1(i)^{r_i}, D_1' = \omega^{ri} \right\}. \tag{7}$$

We assume that the user gid is assigned time and space access rights $[t_a, t_b]$ and $[l_a, l_b]$, where F_t and F_{loc} represent the unique format of time and space in the system, respectively. It should be noted that $[t_a, t_b]$ and $[l_a, l_b]$, respectively, represent the starting time and ending time and the starting position and ending position in the system, and all elements are discrete integers with a total order. KGC selects an arbitrary $r_t \in Z_n^*, r_l \in Z_n^*$ for each time attribute and space attribute and generates the corresponding time key and space key as shown below:

$$\mathrm{DK}_{[ta,tb]} = \left\{ \begin{array}{c} D_t = \left(gH_0(\mathrm{gid})^{uj}\right) \cdot H_1(F_t)^{rt}, D_t' = \omega^{rt} \\ D_t^{''} = \left(v_{\{ta,tb\}}\right)^{rt} = \varphi^{rt} \lambda ta \mu^{z-tb} \end{array} \right\},$$
$$\mathrm{DK}_{[l_a,l_b]} = \left\{ \begin{array}{c} D_l = \left(gH_0(gid)^{u_j}\right) \cdot H_1(F_l)^{rl}, D' = \omega^{rl} \\ D_l^{''} = \left(v_{\{l_a,l_b\}}\right)^{rl} = \varphi^{rl\lambda la\mu z - lb} \end{array} \right\}. \tag{8}$$

Finally, it outputs the user key $\mathrm{SK}_{\mathrm{gid}} = \left\{\mathrm{SK}_{\mathrm{Attr}}, \mathrm{DK}_{[ta,tb]}, \mathrm{DK}_{la,lb}\right\}$.

Among them, q_x^0 is the parameter inherited from its parent node (or if x is the root node, it is used to process the root node), and q_x^1 is the parameter shared with its child nodes (or if x is a leaf node, then it is used for processing the relevant attributes of the user). If the time threshold or position threshold is related to node x, then it will be related to $t_x^0 \in Z_n^*$ and $l_x^0 \in Z_n^*$, respectively. For any node, the value q_x^1 can be used to calculate as follows:

$$\left\{ \begin{array}{ll} q_X^1 = q_x^0 - l_x^0 - t_x^0 & x \text{ is location and time dependent} \\ q_X^1 = q_x^0 - l_x^0 & x \text{ is location dependent} \\ q_X^1 = q_x^0 - t_x^0 & x \text{ is time dependent} \\ q_X^1 = q_x^0 & \text{Other situations} \end{array} \right\}. \tag{9}$$

The polynomial q_x arbitrarily chooses $q_x(0) = q_x^1 \sqrt{a^2 + b^2}$ and $d_x = k_x - 1$. $q_x^0 = q_{\mathrm{paraent}}(x)(\mathrm{index}(x))$. Among them, χ is the collection of leaf nodes in the access tree, γ represents the attribute collection related to the time range $[t_a, t_b]$, and Z represents the attribute collection related to the location range $[l_a, l_b]$. Therefore, the structure of the ciphertext is as follows:

$$\mathrm{CT} = \left\{ \begin{array}{c} T, \widetilde{C} = \mathrm{Enc}(\kappa, m), C = \kappa e(g, \omega)^{\alpha s}, C' = h^s \\ \forall x \in \chi, C_x = \omega^{q_x^1}, C_x' = H_1(\mathrm{att}(x))^{q_x^1} \\ \forall y \in \gamma, C_y = \omega^{t_y\prime}, C_y' = H_1(A_t)^{t_y^0} \\ C_y^{''} = \left(v_{\{t_i,t_j\}}\right)^{t_y^0} = \varphi^{t_y^0 \lambda^{t_i} \mu^{z-t_j}} \\ \forall z \in Z, C_z = \omega^{l_z^0}, C_z' = H_1(A_l)^{l_z^0} \\ C_z^{''} = \left(v_{\{l_i,l_j\}}\right)^{l_z^0} = \varphi^{l_z^0 \lambda^i \mu^{z-l_j}} \end{array} \right\}. \tag{10}$$

The cloud server will check whether the time t_c and location l_c of the node meet the time range constraint $[t_a, t_b]$ and the location range constraint $[l_a, l_b]$. If it is not satisfied, $C_y^{''}$ and $C_z^{''}$ are set to the special symbol ⊥. If it is satisfied:

$$\begin{aligned}
\widetilde{C}_y^{''} &= C_y \cdot F_{\{t_i \le t_c, t_j \ge t_c\}}\left(C_y^{''}\right) \\
&= C_y \cdot F_{\{t_i \le t_c, t_j \ge t_c\}}\left(v_{\{t_i, t_j\}}\right)^{t_y^0} \\
&= C_y \cdot \left(\varphi^{t_y^0 \lambda^{t_i} \mu^{z - t_j}}\right)^{\lambda_c^{t_c - t_i} \mu^{t_{j} - t_c}} \\
&= C_y \cdot \varphi^{t_y^0 \lambda^{t_c} \mu^{z - t_c}} \\
&= \omega^{t_y^0} \cdot \left(v_{\{t_c, t_c\}}\right)^{t_y^0} \\
&= \left(v_{\{t_c, t_c\}} \omega\right)^{t_y^0},
\end{aligned} \tag{11}$$

$$\begin{aligned}
\widetilde{C}_z^{''} &= C_z \cdot F_{\{l_i \le l_c, l_j \ge l_c\}}\left(C_z^{''}\right) \\
&= C_z \cdot F_{\{l_i \le l_l, l_j \ge _l\}}\left(v_{\{l_i, l_j\}}\right)^{l_t^0} \\
&= C_z \cdot \varphi^{l_y^0 \lambda^{l_c} \mu^{z - l_c}} \\
&= \omega^{l_2^0} \cdot \left(v_{\{l_c, l_c\}}\right)^{l_3^0} \\
&= \left(v_{\{l_c, l_c\}} \omega\right)^{l_0^l / 3}.
\end{aligned}$$

Therefore, the final reencrypted ciphertext is

$$\begin{aligned}
\mathrm{CT} = \Big\{ &T, \widetilde{C}, C, C', \left\{C_x, \widetilde{C}_x\right\} \forall x \in \chi, \\
&\left\{C_y, \widetilde{C}_y\right\} \forall y \in \gamma, \left\{C_z, \widetilde{C}_z\right\} \forall z \in Z \Big\}.
\end{aligned} \tag{12}$$

Finally, the cloud server sends the encrypted ciphertext CT', the current time t_c, and the current location l_c to the user.

Here, a recursive algorithm is used to define our decryption process.

First, pass a recursive algorithm DecrytNode $(CT', \mathrm{SK}_{\mathrm{gid}}, x)$ through the input cipher text CT' and the private key $\mathrm{SK}_{\mathrm{gid}}$ and visit the node V in the tree x. If the node is a leaf node, then we set $i = \mathrm{attr}(x)$ and have the following definitions. If $i \in S_{\mathrm{gid}}$, then we have

$$\begin{aligned}
F_x^{\mathrm{atrr}} &= \mathrm{DecrytNode}\left(CT', \mathrm{SK}_{\mathrm{gid}}, x\right) \\
&= \frac{e\left(D_i, C_x\right)}{e\left(D_i', C_x'\right)} \\
&= \frac{e\left(\left(gH_0(\mathrm{gid})\right)^{u_j} H_1(i)^{r_i}, \omega^{q_x{}'}\right)}{e\left(\omega^{r_i}, H_1(\mathrm{att}(x))^{q_x{}'}\right)} \\
&= e\left(gH_0(\mathrm{gid}), \omega\right)^{u_j q_x{}'}.
\end{aligned} \tag{13}$$

If $i \notin S_{\mathrm{gid}}$, then we define Decryt Node $(CT', \mathrm{SKgid}, x) = \perp$.

$t_c \in [t_i, t_j]$ and $t_c \in [t_a, t_b]$ are required to ensure safe time access control. It should be noted that $A_t[t_i, t_j]$ is the access strategy for node y in the access tree, and $A_t[t_a, t_b]$ is the access authority range of the time the user has. When the user's access time is valid, it is calculated as follows:

$$\begin{aligned}
\widetilde{D}_t^{''} &= F_{\{t_a \le t_c, t_b \ge t_c\}}\left(D_t^{''}\right) \\
&= F_{\{t_a \le t_c, t_b \ge t_c\}}\left(v\{t_a, t_b\}\right)^{r_t} \\
&= \left(\varphi^{r_i \lambda^t \mu} \mu^{z - t_b}\right)^{\lambda^{t_c - t_a} \mu^{tb - t_c}} \\
&= \varphi^{r_i \lambda^{t_c} \mu^{- - t_c}} \\
&= \left(v_{\{t_c, t_c\}}\right)^{r_t},
\end{aligned}$$

$$\begin{aligned}
F_y^{\mathrm{time}} &= \frac{e\left(D_t, \widetilde{C}_y^{''}\right)}{e\left(D_t' \widetilde{D}_t^{''}, C_y'\right)} \\
&= \frac{e\left(\left(gH_0(\mathrm{gid})\right)^{u_j} H_1(A_t)^{r_t}, \left(v_{\{t_c, t_c\}} \omega\right)^{t_y^0}\right)}{e\left(\omega^{r_i}\left(v_{\{t_c, t_c\}}\right)^{r_t}, H_1(A_t)^{t_y^0}\right)} \\
&= e\left(gH_0(\mathrm{gid}), \omega\right)^{u_t t_y^0} \cdot e\left(gH_0 \le (\mathrm{gid}), v_{\{t_c, t_c\}}\right)^{u_j t_y^0} \\
&= e\left(gH_0(\mathrm{gid}), \omega\right)^{u_t t_y^0}.
\end{aligned} \tag{14}$$

Similarly, we will consider access to node $\forall z \in Z$ in the tree and require $l_c \in [l_i, l_j]$ and $l_c \in [l_a, l_b]$ to ensure safe time access control. It should be noted that $A_l[t_l, t_l]$ is the access strategy for node z in the access tree, and $A_l[l_a, l_b]$ is the access authority range of the time the user has. When the user's access time is valid, it is calculated as follows:

$$\begin{aligned}
\widetilde{D}_l'' &= F_{\{l_a \le l_c, l_b \ge l_c\}}\left(D_1''\right)\\
&= F_{\{l_a \le l_c, l_b \ge l_c\}}\left(v\{l_a, l_b\}\right)^{r_l}\\
&= \left(\varphi^{r_i \lambda^l \mu^{z-l_b}}\right)^{\lambda^{l_c - l_a}\mu^{l_b - l_c}}\\
&= \varphi^{r_i \lambda^{l_c}\mu^{z-l_c}}\\
&= \left(v_{\{l_c, l_c\}}\right)^{r_l},\\
F_y^{\text{time}} &= \frac{e\left(D_t, \widetilde{C}_y''\right)}{e\left(D_t' \widetilde{D}_t'', C_y'\right)}\\
&= \frac{e\left(\left(gH_0(\text{gid})\right)^{u_j} H_1(A_t)^{r_i}, \left(v_{\{t_c, t_c\}}\omega\right)^{t_y^0}\right)}{e\left(\omega^{r_i}\left(v_{\{t_c, t_c\}}\right)^{r_t}, H_1(A_t)^{t_y'}\right)}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_t t_y^0} \cdot e\left(gH_0(\text{gid}), v_{\{t_c, t_c\}}\right)^{u_t t_y^0}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_t t_y^0}.
\end{aligned} \tag{15}$$

Similarly, we will consider access to node $\forall z \in Z$ in the tree and require $l_c \in [l_i, l_j]$ and $l_c \in [l_a, l_b]$ to ensure safe time access control. It should be noted that $A_l[t_l, t_l]$ is the access strategy for node z in the access tree, and $A_l[l_a, l_b]$ is the access authority range of the time the user has.

$$\begin{aligned}
D_l'' &= F_{\{l_a \le l_c, l_b \ge l_c\}}\left(D_1''\right)\\
&= F_{\{l_a \le l_c, l_b \ge l_c\}}\left(v\{l_a, l_b\}\right)^{r_l}\\
&= \left(\varphi^{r_i \lambda^l a^{\mu} - l_b}\right)^{\lambda_c^{l_c - l_a}\mu^{l_b - l_c}}\\
&= \varphi^{r_i \lambda^{l_c}\mu^{z-l_c}}\\
&= \left(v_{\{l_c, l_c\}}\right)^{r_l},\\
F_z^{\text{loc}} &= \frac{e\left(D_l, C_z''\right)}{e\left(D_l' \widetilde{D}_l'', C_z'\right)}\\
&= \frac{e\left(\left(gH_0(\text{gid})\right)^{u_j} H_1(A_l)^{r_i}, \left(v_{\{l_c, l_c\}}\omega\right)^{l_z^0}\right)}{e\left(\omega^{l_i}\left(v_{\{l_c, l_c\}}\right)^{r_i}, H_1(A_l)^{l_z^0}\right)}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_l l_z^0} \cdot e\left(gH_0(\text{gid}), v_{\{l_c, l_c\}}\right)^{u_j l_z^0}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_l l_z^0}.
\end{aligned} \tag{16}$$

If it has nothing to do with any time limit gate and position limit gate, then we can get the following formula:

$$\begin{aligned}
F_x &= F_x^{\text{atrr}}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_i q_x'}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_i g_{x^{\circ}}}.
\end{aligned} \tag{17}$$

We can get the following formula:

$$\begin{aligned}
F_x &= F_x^{\text{attr}} \cdot F_x^{\text{time}}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_j\left(q_x^1 + t_x^0\right)}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_i q_k^0}.
\end{aligned} \tag{18}$$

For any node x, if it is related to any position limit gate, then we can get the following formula:

$$\begin{aligned}
F_x &= F_x^{\text{atrr}} \cdot F_x^{\text{loc}} = e\left(gH_0(\text{gid}), \omega\right)^{u_j\left(q_x^1 + l_x^0\right)}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_i q_e^0}.
\end{aligned} \tag{19}$$

With limit gate and position limit gate, then we can get the following formula:

$$\begin{aligned}
F_x &= F_x^{\text{attr}} \cdot F_x^{\text{loc}} \cdot F_x^{\text{time}}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_j\left(q_x^1 + l_x^0 + t_x^0\right)}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_j q_x^0}.
\end{aligned} \tag{20}$$

Finally, we need to consider when x is a nonleaf node recursively; for node x, its child node set S_x composed of all its child nodes z has k_x children. If there is no such set, the node is not satisfied and the decryption function returns ⊥. Otherwise, we will calculate

$$\begin{aligned}
F_x^{\text{atrr}} &= \prod_{z \in S_x} F_z^{\Delta, S_x(0)} \text{ among} \begin{Bmatrix} i = \text{index}(z) \\ S_x' = \{\text{index}(z) \colon z \in S_x\} \end{Bmatrix}\\
&= \prod_{z \in S_x}\left(e\left(gH_0(\text{gid}), \omega\right)^{u_i q_t^0}\right)^{\Lambda_i, S_x(0)}\\
&= e\left(gH_0(\text{gid}), \omega\right)^{u_j q_x'}.
\end{aligned} \tag{21}$$

We can decrypt the plaintext as follows:

$$\begin{aligned}
\kappa &= \frac{C}{e\left(D, C'\right) \cdot F_R},\\
m &= \text{Dec}(\kappa, C).
\end{aligned} \tag{22}$$

The DCNN-FDM deep convolutional neural network defect recognition model used in this paper is shown in Figure 3.

4. Steel Plate Defect Recognition of Deep Neural Network Recognition System Based on Space-Time Constraints

In order to process the massive image data stream generated instantaneously and ensure the real-time performance,

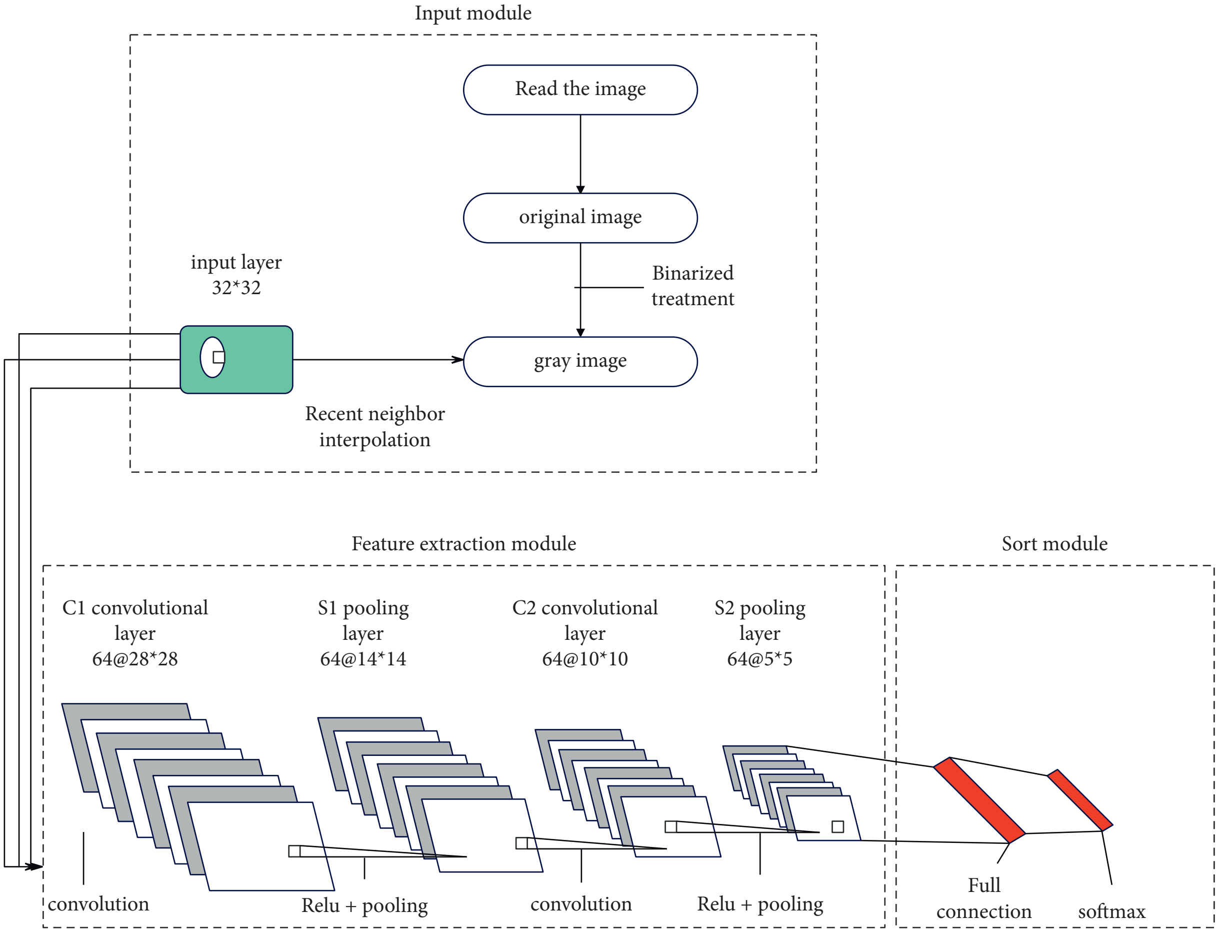

Figure 3: DCNN-FDM deep convolutional neural network defect recognition model.

accuracy, and stability of the detection system, a distributed parallel computing system structure based on the client/server (C/S) model can be constructed, as shown in Figure 4. The overall architecture diagram of the online detection system for steel plate surface defects is shown. It can be seen from the figure that the entire detection system is composed of a client (lower computer), a server (upper computer), and a control terminal. Each camera is controlled by an independent client to ensure parallel processing of image data. The client receives related instructions from the control terminal to control the camera to realize the image collection function. At the same time, the client transmits the processed image data stream to the server and displays the relevant processing results in real time on the control terminal. The control terminal realizes the control of the entire detection system by setting the detection method and related parameters of the hardware device. Among them, the interaction of the image data stream between the server and the client can be carried out by using gigabit (1 Gb/s) industrial Ethernet, and the interaction of the control signal flow between the server and the client can be carried out by using a field bus.

It is necessary to design a software system with a reasonable structure and clear levels to manage and deploy all hardware devices. At the same time, it is also necessary to design a fast, accurate, and stable image detection method to process and analyze the image, so that the entire detection system can run efficiently and in an orderly manner. Figure 5 shows the software architecture diagram of the online detection system for steel sheet surface defects.

The image detection method of steel plate surface defects can be divided into two parts: online processing and offline processing. Among them, the online processing adopts a streamlined operation mode that combines real-time processing and quasi-real-time processing. Figure 6 shows the schematic flowchart of the image detection method for steel sheet surface defects. The preprocessing and screening operations of the steel plate surface image must be completed online and in real time in order to find and mark the steel plate surface defect image in time. However, operations such as segmentation and classification of the image of steel surface defects can be processed in the online quasi-real-time link, and the training of the classifier can be processed in the offline link.

Due to the single color of the steel plate surface, the difference between the defective area and the normal area can be distinguished by the difference in light intensity. Therefore, this experimental platform uses a black and white line scan camera. Figure 7 shows the imaging schematic diagram of a line scan camera. The yellow area in the figure

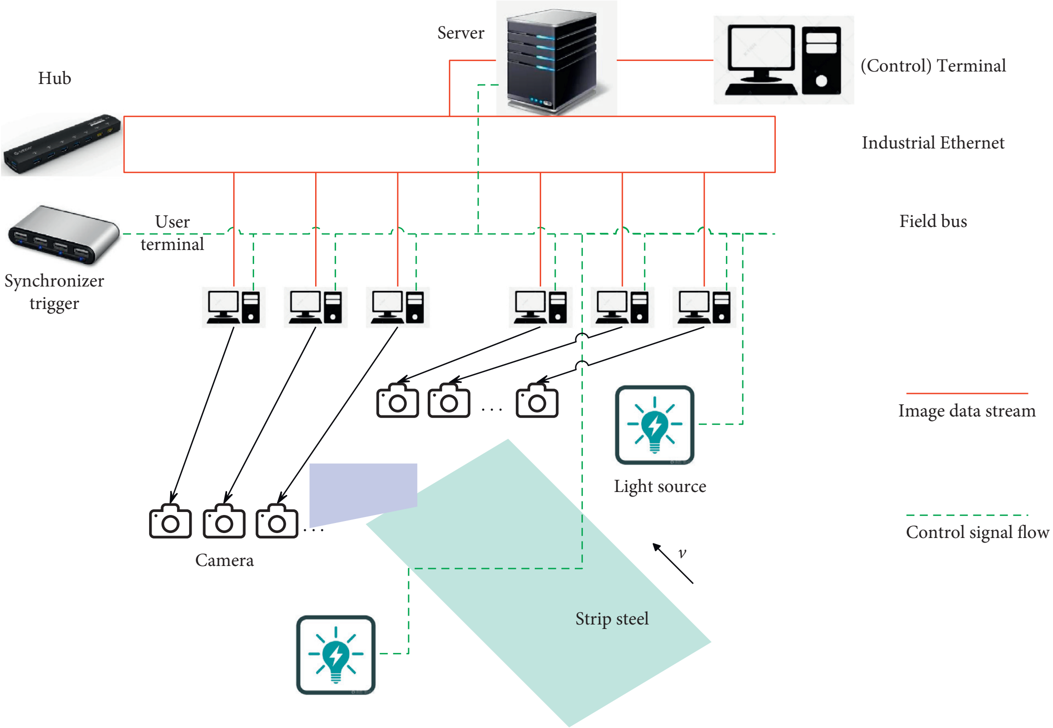

Figure 4: The overall architecture diagram of the online detection system for steel plate surface defects.

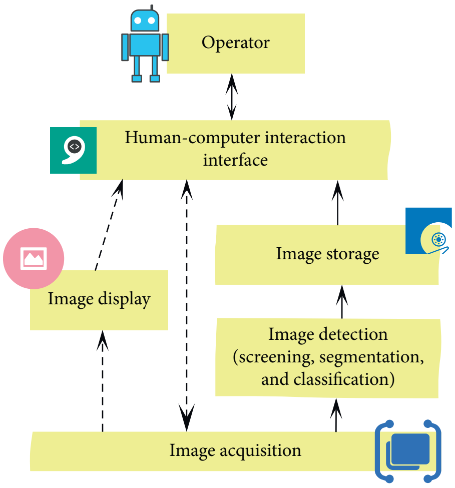
Operator
Human-computer interaction interface
Image storage
Image display
Image detection (screening, segmentation, and classification)
Image acquisition

Figure 5: Software architecture diagram of the online detection system for steel plate surface defects.

represents an image pixel (dot), rw represents the horizontal resolution, and rh represents the vertical resolution.

The steel plate defect recognition is carried out through the system of this article, as shown in Figure 8. It is a schematic diagram of the steel plate defect recognition.

The system introduces the overall structure, analyzes the light source lighting module and the image acquisition module, and effectively reduces the impact of the illumination angle on the image quality of the steel plate. Moreover, this article separately elaborates the main functions of the client, server, and monitoring console to complete the macrocontrol.

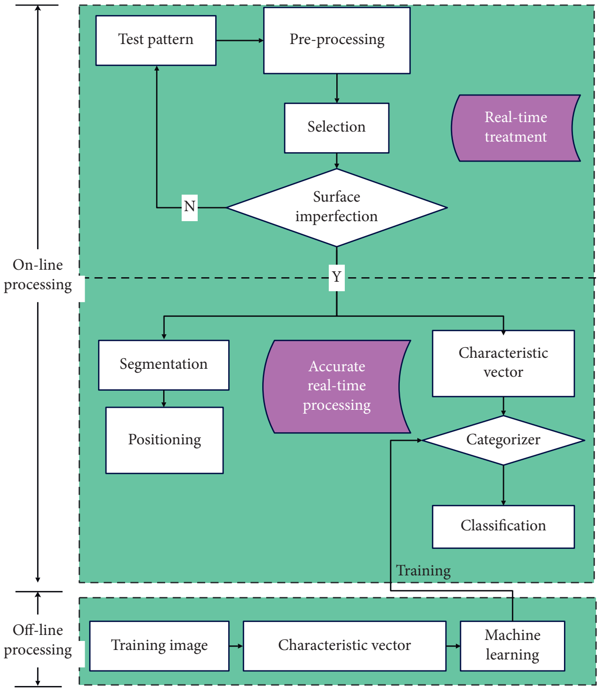

Figure 6: Schematic diagram of the image detection method of steel plate surface defects.

The data used to describe the defect information is huge. Therefore, accurate and effective recording of these data is very important for the system to quickly and accurately obtain the recognition results. The database management system is used to separate the detection system and the database, and

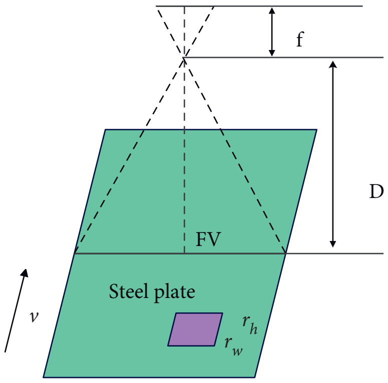

Figure 7: Schematic diagram of imaging of a line scan camera.

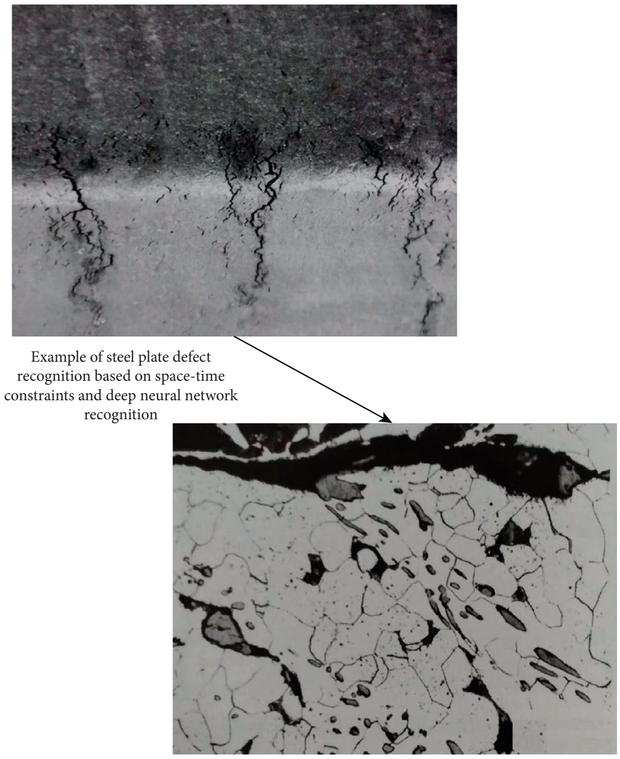
Example of steel plate defect recognition based on space-time constraints and deep neural network recognition

Figure 8: Schematic diagram of steel plate defect recognition.

the collected steel plate defect image information is converted into the set standard format and stored in the database. While ensuring real-time detection, it can ensure the safety and stability of steel plate image data and its dependence on historical data. In the steel plate surface defect detection system designed in this paper, the database is mainly used in two modules, one module is the application of real-time image acquisition, and the other module is the application of image punctual processing. In the database design process, for different types of data, a reasonable classification of the planning table is carried out to make the database hierarchical, organized, and easy to access and process the data. For different tables, this article uses reasonable definition of the fields of each table to avoid redundant data and combines the actual needs to determine the data type of each field reasonably and finally achieves the purpose of convenient retrieval and easy maintenance. Finally, this article chooses the only certain content as the primary key of the data table, and only one primary key can be defined in a table, and the connection between the tables is established through the primary key, thus forming a set of interconnected tables.

The simulation platform is used to construct a deep neural network recognition system based on space-time constraints. This paper selects 7 types of defect images collected at the production site. Table 1 shows the types and numbers of steel defect samples selected for testing.

The defect identification of the system of this paper mainly includes two parts: preprocessing and defect splitting. Steel plate defect pretreatment detection is to screen out defective steel plates, and the steel plate classification setting classifies the defective steel plates, and the pretreatment results are shown in Table 2 below.

Table 1: Types and numbers of steel defect samples selected for testing.

Types of defects	Category number
Scarring	A
Transverse crack	B
Horizontal scratch	C
Longitudinal crack	D
Longitudinal scratch	E
Reticulated	F
Pitting	G

Table 2: Statistical table of accuracy of steel plate defect pretreatment detection.

Group no	Predetection accuracy rate (%)	Group no	Predetection accuracy rate (%)
1	98.92	16	96.97
2	96.18	17	98.92
3	96.48	18	96.61
4	97.19	19	98.69
5	98.68	20	97.54
6	97.50	21	96.98
7	97.94	22	96.33
8	98.75	23	97.40
9	98.55	24	97.86
10	97.03	25	98.01
11	98.75	26	97.02
12	96.23	27	96.01
13	97.73	28	97.27
14	98.01	29	96.44
15	98.21	30	97.08

Table 3: Evaluation of the effect of steel plate defect classification.

Types of defects	Category number	Classification accuracy (%)
Scarring	A	90.99
Transverse crack	B	93.24
Horizontal scratch	C	91.61
Longitudinal crack	D	89.02
Longitudinal scratch	E	89.95
Reticulated	F	89.45
Pitting	G	87.23

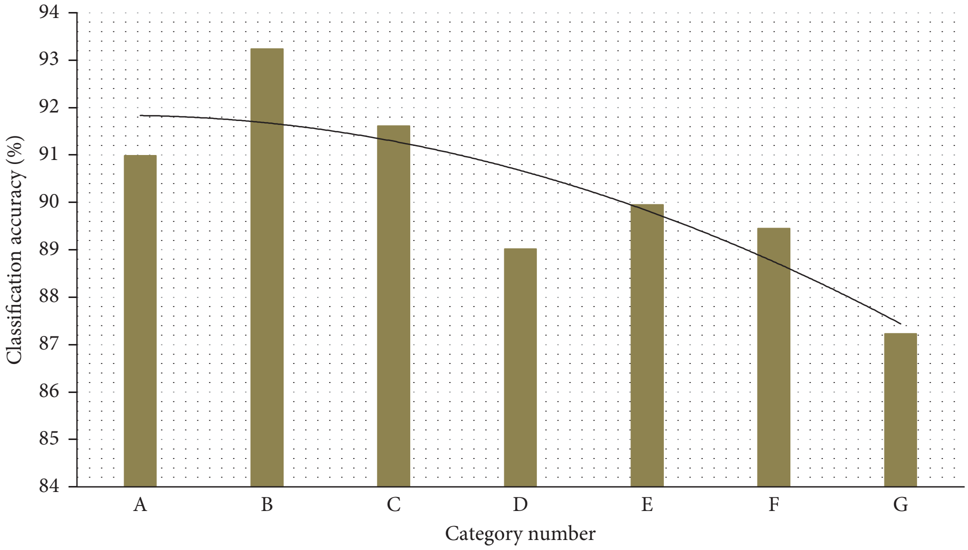

Figure 9: Statistical diagram of the effect of steel plate defect classification.

Based on the above research, this paper evaluates the effect of steel plate defect classification on the deep neural network recognition system based on space-time constraints, and the results are shown in Table 3 and Figure 9.

From the above research, it can be seen that the deep neural network recognition system based on space-time constraints proposed in this paper has a good effect in the recognition of steel plate defects, and it can be applied to the automatic detection of steel plate online production.

5. Conclusion

The defects on the surface of the steel plate have different shapes and complex causes. According to the appearance characteristics of defects, it can be divided into point defects, line defects, and surface defects. Point defects appear on the surface of the steel plate in the form of dots. Some are single-point occurrences, some are continuous occurrences in relatively concentrated areas, some are periodic, such as roll prints, sticky rolls, etc., and some have no regularity, such as spot rust, pitting, etc. Line defects often appear in the form of lines. Some are continuous, some are intermittent, some are horizontal, and some are vertical, such as scratches, cracks, and streaks. Surface defects often appear in slices on local areas of the surface of the steel plate, which have a certain area, such as emulsion spots, embroidered spots, and oil stains. This article will build an intelligent system that can be used for steel plate defect recognition based on the deep neural network algorithm based on space-time constraints. Through experimental research, it can be seen that the deep neural network recognition system based on space-time constraints proposed in this paper has a good effect in the recognition of steel plate defects, and it can be applied to the automatic detection of steel plate online production.

Data Availability

The labeled dataset used to support the findings of this study is available from the corresponding author upon request.

Conflicts of Interest

The authors declare no competing interests.

Acknowledgments

This study was sponsored by Science and Technology on Special System Simulation Laboratory of Beijing Simulation Center and Peking University.

References

[1] L. Hu, M. Zhou, F. Xiang, and Q. Feng, "Modeling and recognition of steel-plate surface defects based on a new backward boosting algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 94, no. 9, pp. 4317–4328, 2018.

[2] Y. Wang, H. Xia, X. Yuan, L. Li, and B. Sun, "Distributed defect recognition on steel surfaces using an improved random forest algorithm with optimal multi-feature-set fusion," *Multimedia Tools and Applications*, vol. 77, no. 13, pp. 16741–16770, 2018.

[3] Z. Mentouri, A. Moussaoui, D. Boudjehem, and H. Doghmane, "Steel strip surface defect identification using multiresolution binarized image features," *Journal of Failure Analysis and Prevention*, vol. 20, no. 6, pp. 1917–1927, 2020.

[4] Z. Peng, X. U. Ke, and Y. Chaolin, "Surface defect recognition for moderately thick plates based on a SIFT operator," *Journal of Tsinghua University*, vol. 58, no. 10, pp. 881–887, 2018.

[5] Q. Luo, X. Fang, L. Liu, C. Yang, and Y. Sun, "Automated visual defect detection for flat steel surface: a survey," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 3, pp. 626–644, 2020.

[6] J. Li, Z. Su, J. Geng, and Y. Yin, "Real-time detection of steel strip surface defects based on improved YOLO detection network," *IFAC-PapersOnLine*, vol. 51, no. 21, pp. 76–81, 2018.

[7] H. Wang, S. Wei, R. Huang et al., "Recognition of plate identification numbers using convolution neural network and character distribution rules," *ISIJ International*, vol. 59, no. 11, pp. 2044–2051, 2019.

[8] R. Hao, B. Lu, Y. Cheng, X. Li, and B. Huang, "A steel surface defect inspection approach towards smart industrial monitoring," *Journal of Intelligent Manufacturing*, vol. 32, no. 7, pp. 1833–1843, 2021.

[9] J. Liu, G. Xu, L. Ren, and Z. Qian, "Defect intelligent identification in resistance spot welding ultrasonic detection based on wavelet packet and neural network," *International Journal of Advanced Manufacturing Technology*, vol. 90, no. 9, pp. 2581–2588, 2017.

[10] J. Zhang, X. Kang, H. Ni, and F. Ren, "Surface defect detection of steel strips based on classification priority YOLOv3-dense network," *Ironmaking and Steelmaking*, vol. 48, no. 5, pp. 547–558, 2021.

[11] J. D. Kothari, "detecting welding defects in steel plates using machine learning and computer vision algorithms," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 7, no. 9, pp. 3682–3686, 2018.

[12] P. P. Sarkar, S. K. Dhua, S. K. Thakur, and S. Rath, "Analysis of the surface defects in a hot-rolled low-carbon C-Mn steel plate," *Journal of Failure Analysis and Prevention*, vol. 17, no. 3, pp. 545–553, 2017.

[13] M.-H. Yao and Q.-L. Gu, "A sparse representation method for image-based surface defect detection," *Optoelectronics Letters*, vol. 14, no. 6, pp. 476–480, 2018.

[14] E. D. Cho and G. B. Kim, "A study on illumination mechanism of steel plate inspection using wavelet synthetic images," *Journal of the Semiconductor & Display Technology*, vol. 17, no. 2, pp. 26–31, 2018.

[15] A. Anvar and Y. I. Cho, "Automatic metallic surface defect detection using shuffledefectnet," *Journal of The Korea Society of Computer and Information*, vol. 25, no. 3, pp. 19–26, 2020.

[16] B. Li, Z. Zhang, H. Liu, P. Lan, and H. Tang, "Characteristics and evolution of the spot segregations and banded defects in high strength corrosion resistant tube steel," *Acta Metallurgica Sinica*, vol. 55, no. 6, pp. 762–772, 2019.

[17] Y. Chen, H.-W. Ma, and M. Dong, "Automatic classification of welding defects from ultrasonic signals using an SVM-based RBF neural network approach," *Insight - Non-Destructive Testing and Condition Monitoring*, vol. 60, no. 4, pp. 194–199, 2018.

[18] S. J. Kim and G. B. Kim, "A study on the defect classification of low-contrast· uneven· featureless surface using wavelet transform and support vector machine," *Journal of the Semiconductor & Display Technology*, vol. 19, no. 3, pp. 1–6, 2020.

[19] Z. Liu, J. Yao, C. He, Z. Li, X. Liu, and B. Wu, "Development of a bidirectional-excitation eddy-current sensor with magnetic shielding: detection of subsurface defects in stainless steel," *IEEE Sensors Journal*, vol. 18, no. 15, pp. 6203–6216, 2018.

[20] Y. Feng, K. Li, Y. Gao, J. Tan, and H. Zhen, "Shaft surface defect detection method based on feature and morphology reconstruction," *Journal of Zhejiang University (Science Edition)*, vol. 54, no. 3, pp. 427–434, 2020.