

Article

Lightweight Tunnel Defect Detection Algorithm Based on Knowledge Distillation

Anfu Zhu , Bin Wang, Jiaxiao Xie and Congxiao Ma

School of Electronic Engineering, North China University of Water Resources and Electric Power, Zhengzhou 450045, China; z20211181076@stu.ncwu.edu.cn (B.W.); z202210181146@stu.ncwu.edu.cn (J.X.); z20201181029@stu.ncwu.edu.cn (C.M.)

* Correspondence: zhuanfu@ncwu.edu.cn

Abstract: One of the greatest engineering feats in history is the construction of tunnels, and the management of tunnel safety depends heavily on the detection of tunnel defects. However, the real-time, portability, and accuracy issues with the present tunnel defect detection technique still exist. The study improves the traditional defect detection technology based on the knowledge distillation algorithm, the depth pooling residual structure is designed in the teacher network to enhance the ability to extract target features. Next, the MobileNetv3 lightweight network is built into the student network to reduce the number and volume of model parameters. The lightweight model is then trained in terms of both features and outputs using a multidimensional knowledge distillation approach. By processing the tunnel radar detection photos, the dataset is created. The experimental findings demonstrate that the multidimensional knowledge distillation approach greatly increases the detection efficiency: the number of parameters is decreased by 81.4%, from 16.03 MB to 2.98 MB, while the accuracy is improved by 2.5%, from 83.4% to 85.9%.

Keywords: tunnel defects; target detection; knowledge distillation; attention mechanism



Citation: Zhu, A.; Wang, B.; Xie, J.; Ma, C. Lightweight Tunnel Defect Detection Algorithm Based on Knowledge Distillation. *Electronics* **2023**, *12*, 3222. <https://doi.org/10.3390/electronics12153222>

Academic Editor: KC Santosh

Received: 17 June 2023

Revised: 19 July 2023

Accepted: 21 July 2023

Published: 26 July 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Road tunnels of 24,698,900 long meters had been constructed nationwide by the end of 2021, according to the Statistical Bulletin of Transport Industry Development 2021. However, there are several flaws in tunnels, including uncompacted, empty, and water-filled tunnels, because of the circumstances surrounding tunnel construction and the surrounding environment. Tunnel defect detection is the cornerstone of ensuring the safe operation of tunnels because these flaws can significantly impair the service life and safety of tunnels.

Traditional tunnel defect detection methods include visual inspection methods, acoustic inspection methods, and multi-sensor inspection methods. For instance, Minardo and Monsberger [1,2] highlighted the shortcomings of conventional structural monitoring techniques. Infrared thermal imaging was employed by Afshani et al. [3] for tunnel inspection. It is challenging to satisfy the demands of large-scale, effective, and high-precision defect detection using these conventional methods, which are not only susceptible to missing parts and leakage detection but also use a lot of human and material resources. Researchers' focus has steadily shifted to developing inspection techniques based on deep learning that can automatically identify and locate tunnel flaws. The issue of small size and damage overlap in defect identification was successfully resolved by Dong et al. [4] by combining the SegNet model with the focused loss function. A full convolutional network model (FCN) based on classification was utilized by Xue et al. [5], which significantly increased detection efficiency and accuracy while consuming fewer processing resources. By training and testing photos with various backdrop complexity levels, Zhou [6] and coworkers considerably increased the detection speed and accuracy of the Yolov4 network.

Three-dimensional ground-penetrating radar data and the Yolo model were merged by Liu et al. [7] to quickly identify road problems and accomplish automated defect detection. The project team enhanced the SGD network [8] to increase model accuracy and added an adversarial network [9] to increase the dataset. Researchers' attention is now focused on finding ways to ensure lightweight deployment in the context of the development of edge-enabled devices. To address the issue of data learning rate during training, Liu et al. [10] created variable convolutional layers as well as joint learning methodologies. In order to improve interference countermeasures and transfer the knowledge obtained by the model to other recognition models using transfer learning approaches, Liu et al. [11] created a network based on two-layer adversarial. This significantly increases training efficiency and model accuracy. In order to implement the deployment on the robot side, Huang [12] et al. employed a weight quantization approach, which compressed the memory by 22.5 times but required continual parameter adjustment and a lot of labor. Class-aware tracking ratio optimization (CATRO) was utilized by Hu [13] and others to minimize computation, which considerably boosted calculation performance but also increased model sparsity, increased hardware needs, and made it challenging to deploy at the edge devices. Zhang [14] et al. used a knowledge distillation method for human posture prediction to enable lightweight deployment without altering the model's structure. A similar information distillation technique was applied by Zhao [15] et al. to produce a lightweight fusion of infrared and visible pictures. Hinton [16], Romero [17], and Zagoruyko [18] have each put forth a variety of knowledge distillation-based methodologies.

Based on conventional tunnel defect detection methods, deep learning methods, and knowledge migration methods, the study enhances the knowledge distillation methodology. The following are the research's main contributions:

- (1) Create a deep pooling residual structure to pool and weight feature information deeply
- (2) Use MobileNetv3 to optimize the network backbone for improved model lightweight
- (3) Construct a method for multidimensional knowledge extraction that can extract information from both the feature layer and the output layer.

2. Related Work

Deep learning is one of the most popular methods in the field of computer vision, which typically uses deep neural networks to extract features from images. These neural network models often have several layers of neurons, each of which can separate higher-level, more abstract properties from the input data. A back-propagation technique optimizes the models during deep learning training by progressively changing the network parameters and enhancing model performance. Computer vision systems can learn to extract helpful features from input photos for a range of different vision programs by employing deep learning models for training.

2.1. YOLO Method

A highly popular real-time target identification technique, the YOLO (You Only Look Once) [19–21] algorithm offers very quick target recognition at the tradeoff of very little accuracy. As seen in Figure 1, the YOLO algorithm divides the entire image into numerous grids and forecasts bounding boxes and class probabilities for each grid. This method is extremely quick and enables the YOLO algorithm to detect several targets in a single forward pass, which is ideal for the real-time demands of tunnel detection.

The relationship between the network input x and output \hat{y} is as in Equation (1). The loss function in Equation (2) is used to characterize the error between the model output \hat{y} and the desired output y . For the regression problem, the output of the model $h(x)$ and the sample labeling are real numbers, and the common method to evaluate its performance on the sample set is the mean square error, i.e., Equation (3).

$$\hat{y} = h(x_i) \quad (1)$$

$$L(\hat{y}, y) = L(h(x_i), y) \quad (2)$$

$$E(h) = \frac{1}{N} \sum_{i=1}^N (h(x_i) - y_i)^2 \quad (3)$$

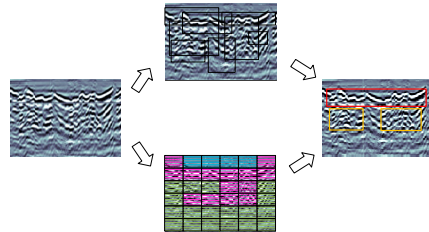


Figure 1. Process of yolo detection in a grid. Each fixed-size cell must identify one object in the image before a confidence score determines which prediction frame is the best.

As shown in Figure 1, in the YOLO algorithm the input image is divided into $S \times S$ grids, each grid contains five pieces of information: (x, y, w, h, c) , where x and y represent the horizontal and vertical coordinates of the prediction frame, w and h represent the width and height of the prediction frame, c represents the confidence level, and the formula is calculated as in Equation (4):

$$c = p_0 \times P_{IOU} \quad (4)$$

where p_0 denotes the probability that the prediction frame contains a detection object, if it does, then $p_0 = 1$, and vice versa $p_0 = 0$; P_{IOU} denotes the area of overlap between the prediction frame and the real detection object area. Then the YOLO algorithm loss function can be expressed as Equation (5):

$$\begin{aligned} L &= L_{IOU} + L_{obj} + L_{class} \\ &= \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^B l_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] + \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^B l_{ij}^{obj} \left[(\sqrt{\omega_i} - \sqrt{\hat{\omega}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \\ &+ \sum_{i=0}^{S^2} \sum_{j=0}^B l_{ij}^{obj} (c_i - \hat{c}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^B l_{ij}^{no} (c_i - \hat{c}_i)^2 + \lambda_{obj} \sum_{i=0}^{S^2} l_{ij}^{obj} \sum_{c \in class} (p_i(c) - \hat{p}_i(c))^2 \end{aligned} \quad (5)$$

where L_{IOU} denotes the regression loss, which measures the match between the prediction frame and the true frame, and is calculated as the intersection ratio of the prediction frame and the true frame, and L_{obj} denotes confidence loss, which measures the presence or absence of a target in the prediction frame, i.e., the probability that a target exists in the prediction frame; L_{class} denotes classification loss, which is used to measure the accuracy of the model in predicting the target class, and is calculated as the multivariate cross-entropy loss. x_i and y_i denote the true frame coordinates, the \hat{x}_i and \hat{y}_i denote the prediction frame coordinates; ω_i and h_i denote the true frame width and height, and $\hat{\omega}_i$ and \hat{h}_i denotes the predicted frame width and height; l_{ij}^{obj} denotes the part with loss of detection target, and l_{ij}^{noobj} denotes the part without detection target loss, and c_i is the confidence label, and \hat{c}_i is the prediction confidence label; $p_i(c)$ indicates the current category true value (0, 1), and $\hat{p}_i(c)$ denotes the current category probability obtained by the activation function.

2.2. Knowledge Distillation

Deep learning models often require large amounts of computational resources for training and inference, which limits their use in many practical applications. To address this problem, model compression, and acceleration techniques have become popular areas of research.

Among these, the knowledge distillation algorithm [16,22–25] is a very effective model compression and acceleration technique that extracts knowledge from large, complex deep teacher models and passes it on to small, simple student models. This approach can significantly reduce the complexity and computational cost of student models, while still maintaining model performance.

The loss function of the knowledge distillation algorithm utilizes the generalized softmax function defined in Equation (6):

$$q_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \quad (6)$$

where z_i denotes the output of the teacher model and q_i is the output of using the softmax function, the dark knowledge of the teacher's network can be better extracted to guide the learning of the student model. To solve the problem of prediction results converging to the zero phenomena in the case of multiple classifications, the cross-entropy loss function is constructed as in Equation (7):

$$L_i = -\log q_i = -\log \frac{\exp(z_i)}{\sum_j \exp(z_j)} \quad (7)$$

where L_i denotes the cross-entropy loss function for the i th sample, and q_i denotes the model output of the i th sample. By minimizing the cross-entropy loss function, the student model output can be made closer to the teacher model output. The final target loss function for the teacher model and the student model is obtained as in Equation (8):

$$L = \alpha L_{\text{soft}} + \beta L_{\text{hard}} \quad (8)$$

where L_{soft} denotes the teacher model with a soft loss of cross-entropy to the student model, L_{hard} denotes the student model with a hard loss of cross-entropy to the true value, and α and β denote the weights of the two loss components.

3. Improving the Model

The multidimensional distillation model demonstrated in this section is shown in Figure 2 and consists of three main components: the teacher network, the student network, and the distillation architecture, where rectangles indicate the neural network layers and arrows indicate the direction of information flow. During the training process, the input data were first fed into the teacher network for feature extraction, resulting in a series of complex feature representations. These feature representations are then fed into the distillation architecture, which is transformed by temperature parameters and soft targets to obtain an intermediate representation suitable for learning by the student network. Finally, the student network is trained based on the intermediate representation to learn the knowledge in the teacher network and obtain the final output.

3.1. Deep Pooling of Residual Structures

In the knowledge distillation process, a teacher model is often a high-performance, high-complexity model. The teacher model helps the student model learn a more accurate and generalized representation of knowledge by "teaching" what it has learned from the training data. Improving the teacher model can improve knowledge transfer efficiency and knowledge generalization, reduce overfitting, and improve the interpretability of the model, leading to better knowledge distillation and better applications.

The YOLO algorithm introduces the concept of residual blocks in the training process to avoid problems such as gradient disappearance and low training efficiency due to the depth of the network [26] as shown in Figure 3. The introduction of the residual structure allows YOLO to achieve deeper and easier training while improving classification accuracy [27,28].

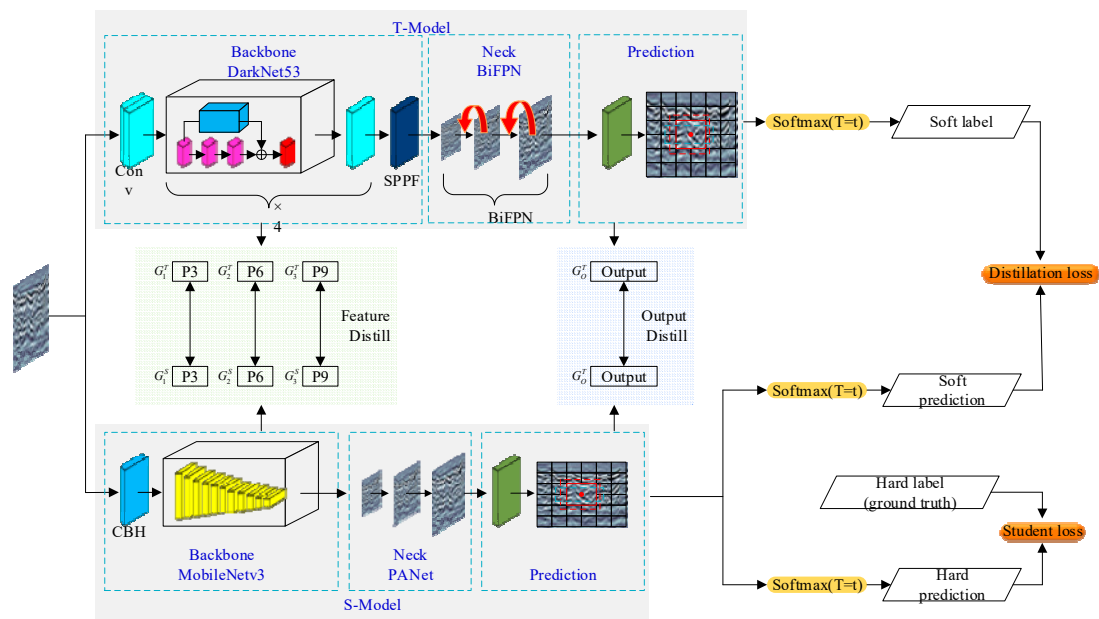


Figure 2. The multidimensional knowledge distillation structure flow, where T-Model is a complicated teacher model with high detection accuracy and S-Model is the soft loss function and the hard loss function connects these two models. Section, feature layer feature information is distilled, and in the Output Distill part, output image information is distilled.

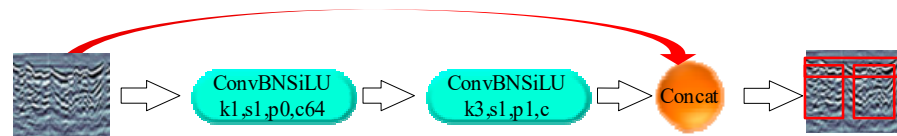


Figure 3. The most basic residual structure, with jump connections shown by red arrows.

The improved residual structure is applied to the C3 module of the YOLO algorithm, as shown in Figure 4. Two simultaneous pooling layers are added to the jump connection section, namely average pooling and maximum pooling, where average pooling can preserve the spatial information of the feature map and smooth the feature map to improve robustness; maximum pooling can smooth the input feature map noise; and maximum pooling can smooth out the noise of the input feature map and extract the most significant feature information from the feature map. In summary, the pooling layer can reduce the size of the input feature map, reduce the number of parameters in the network, and further learn the feature information.

The new feature information is obtained by normalizing the output of the two pooling layers by giving weights to the weights, which are fed into the convolution layer as an intermediate layer feature, and the deflated low-resolution feature map is mapped to the original feature map in higher space, thus restoring the expressiveness of the feature information, after which the non-linear expressiveness of the feature is enhanced by the Relu activation function.

3.2. Multidimensional Knowledge Distillation

Deep neural networks excel at learning multi-level feature representations of increasing abstraction, so both the image output of the final layer and the feature output of the middle layer can be used as knowledge for supervised student model training. In this study, a multidimensional knowledge distillation algorithm fusing the feature and output layers is designed to transfer pre-trained dark knowledge of complex teacher models to lightweight student models for better control of the compression performance trade-offs. Researchers have created thin network topologies including SqueezeNet, ShuffleNet, and

MobileNet for the lightweight student model. The employment of the Fire module, which lowers the number of network parameters while raising the nonlinear transformations, is the key component of the SqueezeNet network. The Fire module combines a Squeeze layer, which reduces the number of channels, with an Expand layer, which increases the number of channels. This significantly reduces the number of convolution kernels but at the expense of a deeper network architecture, which prolongs detection times and significantly lowers detection accuracy. ShuffleNet networks are most notable for their use of channel shuffle operations, which involve first segmenting the input channel into a number of groups and then rearranging each group's channels to improve inter-group communication. However, because the operation is so complex, it is more challenging to implement at the device's edge. The foundation of the MobileNet network is the division of convolution into Depthwise Conv and Pointwise Conv. It has been upgraded to the v3 version, which adds new technologies such as SE Attention Module, h-swish activation function, etc., and further reduces network size and computation amount in comparison to SqueezeNet and ShuffleNet networks and enhances precision without lengthening consumption time.

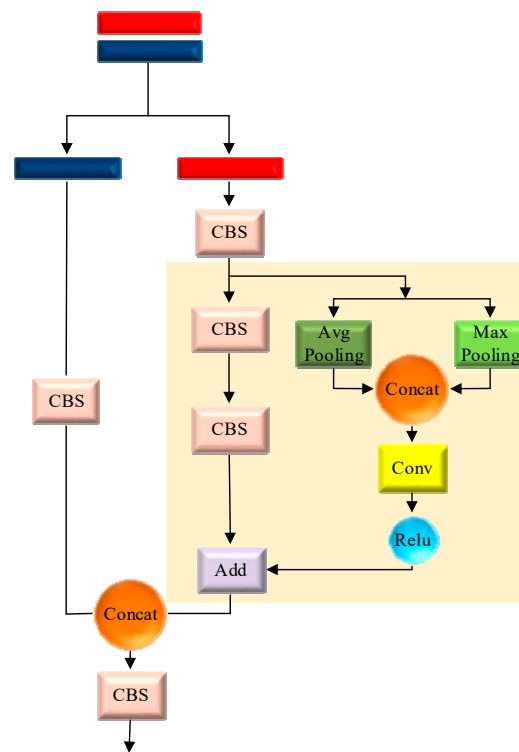


Figure 4. Improved deep pooling residual structure, where the improved portion is the yellow background region and is primarily connected by two pooling weights. The improved activation function is Relu, which enhances the capability of feature extraction.

The knowledge transfer process is shown in the Figure 5. In the training process, the original data were first trained using a complex teacher model, which records the feature representations of the intermediate and output layers. These feature representations are then passed on to the lightweight student model for training as dark knowledge. During the training of the student model, a multidimensional knowledge distillation algorithm is used to transfer the dark knowledge from the teacher model to the student model and to train it in conjunction with the student model's feature representations. The multidimensional knowledge distillation algorithm fuses the intermediate and output layer feature representations from the teacher model with the corresponding layer feature representations from the student model in a weighted manner, thus better guiding the learning of the student model.

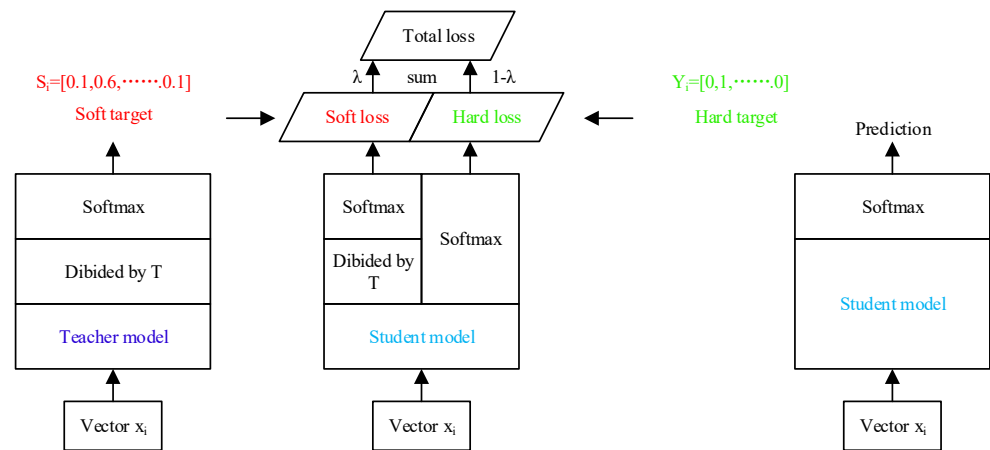


Figure 5. The multidimensional knowledge distillation process, where the leftmost component is the teacher model learning process, is connected to the middle student model through the distillation parameter “T” to form Soft loss, and the rightmost component is the real label with parameter “T” of 1, connected to the student model to form Hard loss, and the two losses are combined to form the full multidimensional knowledge distillation loss.

To help train the student network in depth, the outputs of the intermediate layers of the teacher network were introduced, and then each intermediate and output layer parameter was optimized by a loss function as in Equation (9):

$$L_{FeaD}(f_t(x), f_s(x)) = \mathcal{L}_F(\phi_t(f_t(x)), \phi_s(f_s(x))) \quad (9)$$

Of these, the $f_t(x)$ and $f_s(x)$ are the feature maps of the middle layer of the teacher and student models respectively. When the feature maps of the teacher and student models are not in the same shape, it is common to apply the transformation functions $\phi_t(f_t(x))$ and $\phi_s(f_s(x))$. $\mathcal{L}_F(\cdot)$ denotes the similarity function used to match the feature maps of the teacher and student models, and in this study $\mathcal{L}_F(\cdot)$ denotes the cross-entropy loss function.

The training process is distilled by distilling the output with the high-temperature match to obtain a hard loss L_{hard} defined as the cross-entropy loss between the ground truth label and the student model as in Equation (10):

$$L_{hard} = -\sum_i^N c_i \log(q_i^T) \quad (10)$$

$$q_i^T = \frac{\exp(z_i)}{\sum_k^N \exp(z_k)}$$

Soft loss L_{soft} is defined as the cross-entropy between the teacher model and the student model as in Equation (11):

$$L_{soft} = -\sum_i^N p_i^T \log(q_i^T) \quad (11)$$

$$p_i^T = \frac{\exp(\frac{v_i}{T})}{\sum_k^N \exp(\frac{v_k}{T})}$$

$$q_i^T = \frac{\exp(\frac{z_i}{T})}{\sum_k^N \exp(\frac{z_k}{T})}$$

where v_i denotes the output of the teacher network, z_i denotes the output of the student network and p_i^T , and q_i^T denotes the value of the softmax output of the teacher network and the student network at temperature T for class i, respectively. c_i denotes the value of ground truth at class i. $c_j \in \{0, 1\}$ N denotes the total number of samples.

The total loss in the output layer is obtained through Equation (8) as Equation (12):

$$L = \alpha \sum_i^N p_i^T \log(q_i^T) + \beta \sum_i^N c_i \log(q_i^T) \quad (12)$$

Through a multidimensional knowledge distillation approach, the dark knowledge of a pre-trained complex teacher model is successfully transferred to a lightweight student model, and model compression and acceleration are achieved while maintaining model performance. The experimental results show that the multidimensional knowledge distillation algorithm in this study, which fuses the feature and output layers, can better maintain the performance and robustness of the model while achieving higher compression rates and faster inference than direct pruning or distillation of the student model.

4. Experimental Studies

4.1. Data Processing

To construct the dataset required for this study, we collected multiple segments of tunnel defect radar data and performed post-processing operations such as uncompacted, hollow, and water-filling on these data, which mainly include types of defects such as uncompacted, emptying, hollow, and water-filled. Specifically, image enhancement algorithms were used to enhance the quality and usability of the data, and manual annotation of the data was carried out to obtain accurate annotation information. Some typical examples of defects are illustrated in the Figure 6.

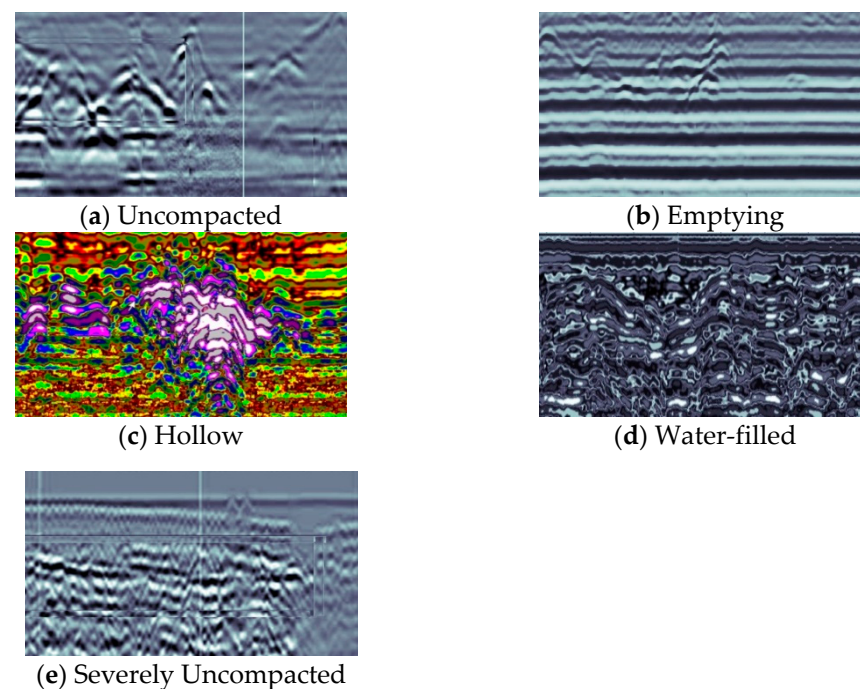


Figure 6. Examples of defects include those where (a) denotes an uncompacted defect, (b) an emptying defect, (c) a hollow defect, (d) a water-filled flaw, and (e) a serious uncompacted defect.

The group increased the data samples by mosaic data enhancement and cropping and stitching methods in previous studies, while the final data set of more than 5700 images was obtained by labeling with LabelImg software (v1.8.6.) to form the data set for this study. The defect images were classified into five categories, such as BM, TK, KD, CS, and YBM, which represent the five types of defects: uncompact, dehollowed, hollow, water-filled, and severely uncompacted defects. Table 1 shows the distribution of the defect images in the dataset.

Table 1. Distribution of Tunnel Defect Dataset.

Defect Type	Training Set	Validation Set	Test Set	Tags
BM	1301	114	76	0
TK	1815	127	139	1
KD	904	78	254	2
CS	1136	80	197	3
YBM	1175	111	107	4

4.2. Experimental Procedure

4.2.1. Experimental Configuration

The ground-penetrating radar vehicle is equipped with ground-penetrating radar equipment that emits radar waves into the ground and then receives the bounced signals to obtain information about underground objects or terrain with the relevant parameters shown in Table 2.

Table 2. Radar rover related parameters.

Name	Configure
Equipment Model	TGRI-GPR200
Center Frequency	200 MHz
Operating Bandwidth	100 MHz–500 MHz
Depth of detection	10 m
Dynamic Range	40 dB

The deep learning simulation experiments were built on a Linux system (20.04.1), using Python (3.8) and PyTorch (1.10.2) to build the deep learning framework. The hardware setup shown in Table 3 includes components such as CPU, GPU, memory, and storage.

Table 3. Experimental hardware configuration.

Name	Configure
Operating System	Linux
Video Card	NVIDIA RTX3090
Video Memory	24G
Processor	Intel(R) Core i3-8100
Programming Language	Python
Deep training framework	PyTorch
Programming Platforms	Pycharm

Table 4 shows some of the trainable parameters in the deep learning model, including weights and biases, which are obtained by back-propagation and optimization of the training data to minimize the loss function of the model.

Table 4. Experimental model parameters.

Name	Value
Pixel	640 × 640
Epoch	100
Batch size	32

4.2.2. Evaluation Indicators

To accurately assess the effectiveness of the target detection algorithm in detecting tunnel defects, the experiments in this study use the mean accuracy (mAP) and the number

of parameters to measure the optimization effect. mAP can be calculated by Precision and Recall, which are calculated as follows in Equation (13).

$$\begin{aligned}
 \text{Precision} &= \frac{TP}{TP+FP} \\
 \text{Recall} &= \frac{TP}{TP+FN} \\
 AP &= \int_0^1 P(R) dR \\
 mAP &= \frac{\sum_{i=1}^N AP_i}{N} \\
 mAP_{[@0.5, @0.95]} &= \frac{\sum_{i=0.5}^{0.95} mAP_{@0.5}}{10}
 \end{aligned} \tag{13}$$

where TP represents the number of positive examples correctly classified, TN represents the number of negative examples correctly classified, FP represents the number of positive examples incorrectly classified, and FN represents the number of negative examples incorrectly classified.

5. Experimental Results

The study used the distillation model for tunnel defect detection and successfully detected a wide range of defect types and sizes, with some of the defect detection results shown in Figure 7.

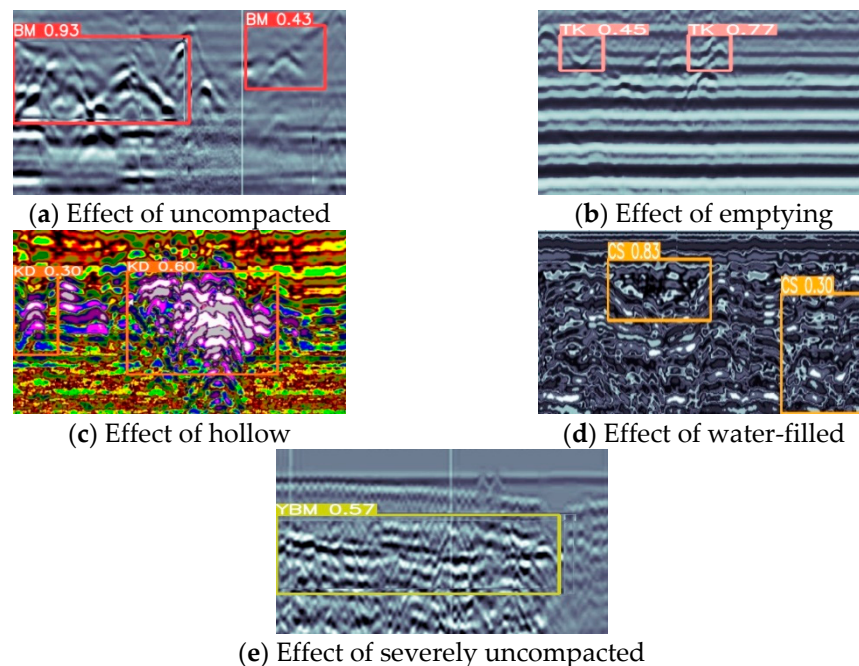


Figure 7. Detection of five different types of defects. (a) Effect of uncompacted; (b) Effect of emptying; (c) Effect of hollow; (d) Effect of water-filled; (e) Effect of severely uncompacted.

The loss function data during the training of the YOLO model, teacher model, student model, and multidimensional knowledge distillation model were represented as scatter plots, and the loss function curves shown in Figure 8 were obtained. By observing the images, it can be seen that in the 20th round, the loss function starts to converge and then tends to stabilize. The convergence and stabilization of the loss function values indicate that the model gradually reaches an optimal state.

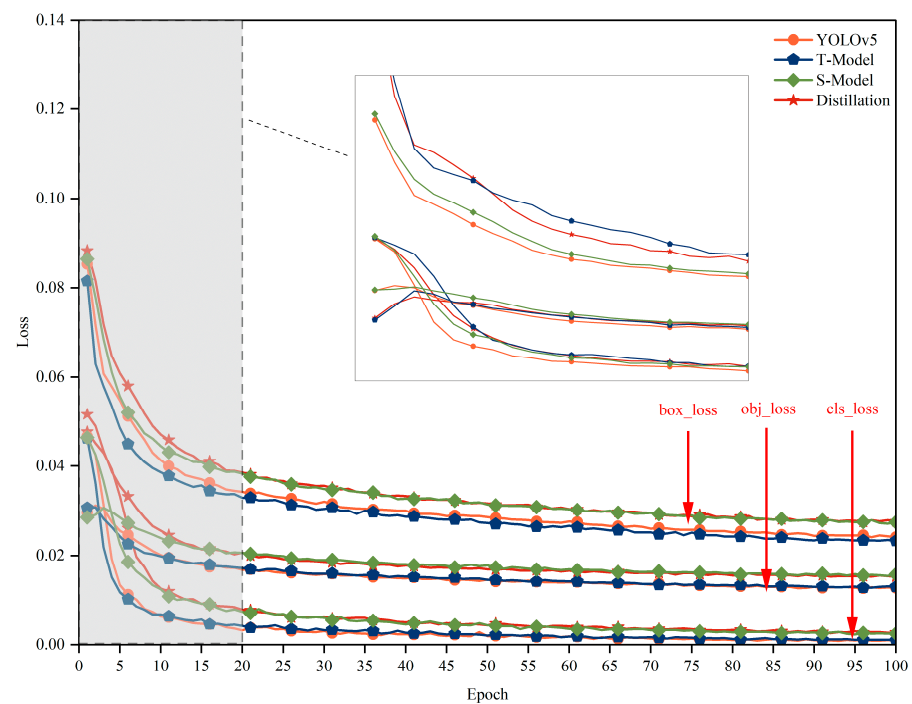


Figure 8. Comparison of the loss function curves of the four models, where orange is the original model, purple is the improved teacher model, green is the student model and red is the multidimensional knowledge distillation model.

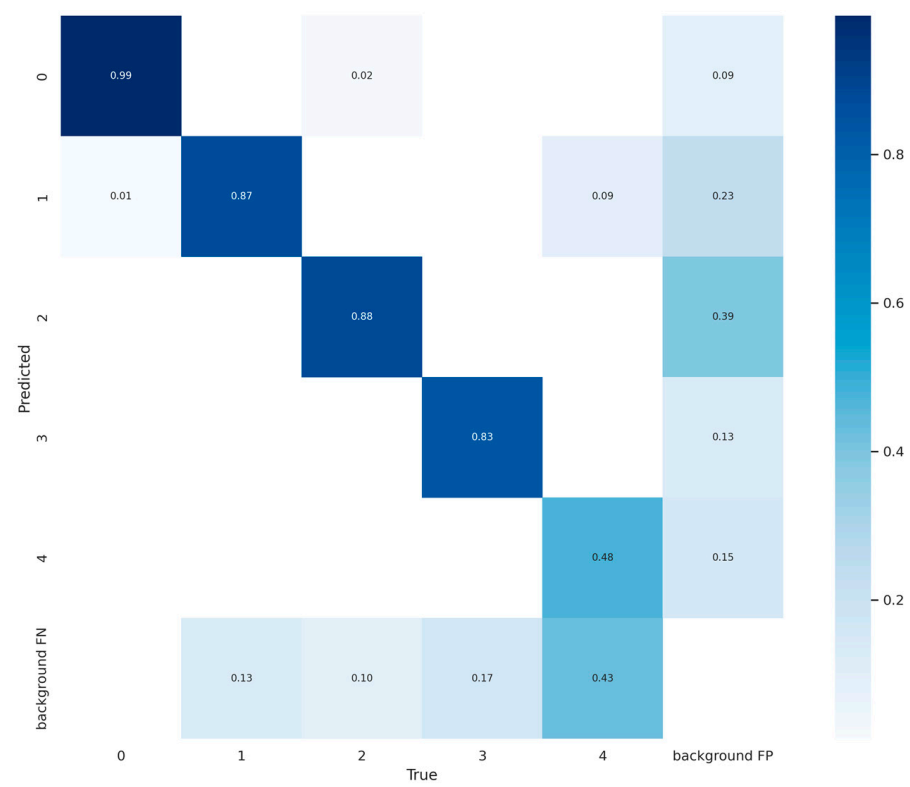
To assess the model classification performance and compare the model prediction results with the actual label differences, the original model was compared with the confusion matrix of the multidimensional distillation model as shown in Figure 9. It can be seen that the improved model has improved the prediction accuracy for all types of defects, the most obvious of which is the case of uncompacted defect prediction, which has improved from 48% to 94%.

Also, to verify the effectiveness of each model for various defects and highlight the importance of each module, the study tallied the detection results of each model on different defect types, as shown in Figure 10 and Table 5. The results showed that the multidimensional distillation model proposed in this study outperformed the original model for various tunnel roadbed defect types, which further validated the effectiveness of the model in tunnel roadbed defect detection.

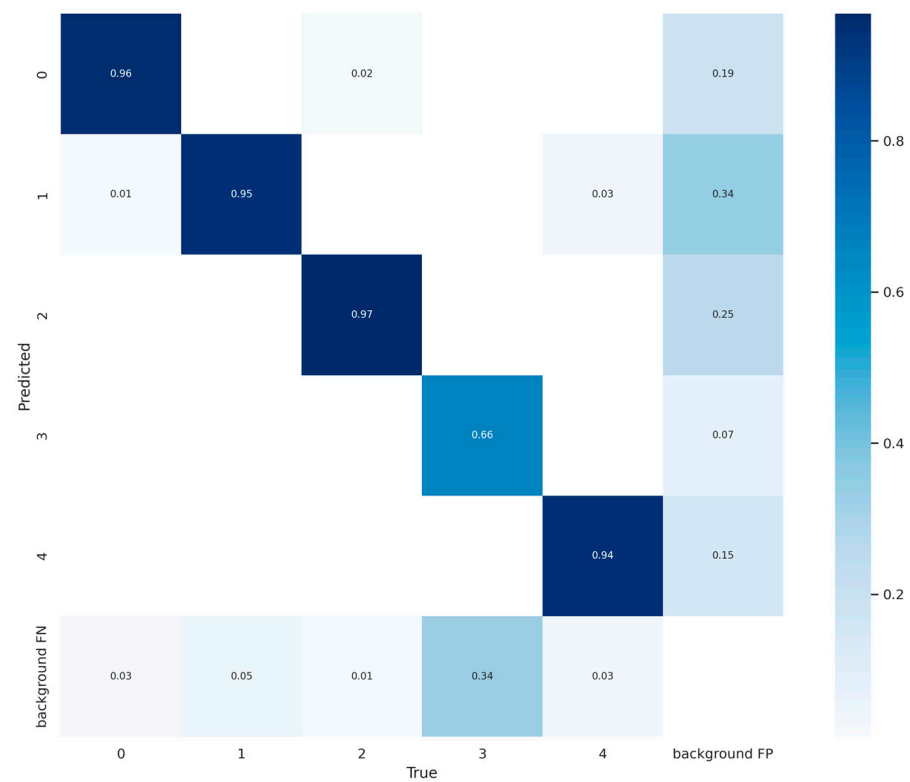
Table 5. Comparison of experimental results of the models in different categories of defects.

Type	Average Accuracy Rate/%				
	BM	TK	KD	CS	YBM
Yolov5	95.6	87.9	88.4	76.1	69.1
T-Model	92.4	88.1	91.6	72.4	82.5
S-Model	89.6	85.8	93.1	67.4	80.2
Distillation	93.5	89.8	93.6	69.0	83.5

To more accurately assess the performance of the multidimensional distillation model proposed in this study, further ablation tests were conducted, and experimental results were obtained as shown in Figure 11 and Table 6. The results show that with the improved distillation technique, an accuracy of 87.1% of the parameter file size was achieved. These results show that the distillation model can significantly improve the lightweight and accuracy of the model in complex tunnel roadbed defect detection scenarios.



(a)



(b)

Figure 9. Confusion_matrix, where (a) denotes the original model confusion matrix and (b) denotes the improved model confusion matrix.

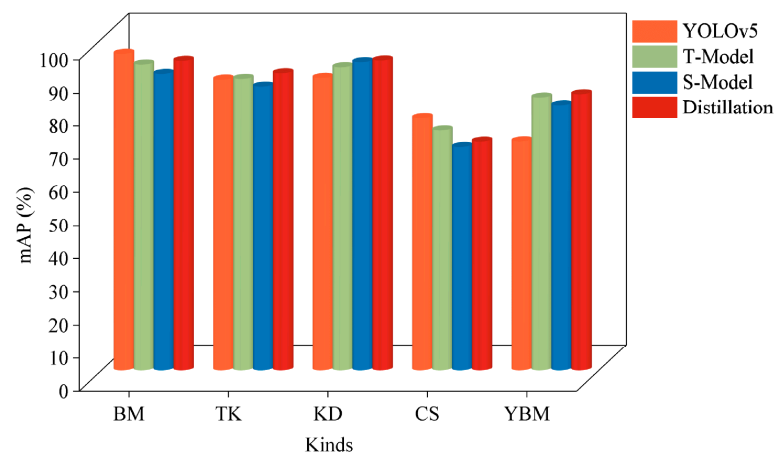


Figure 10. Comparison of different models for different categories of detection.

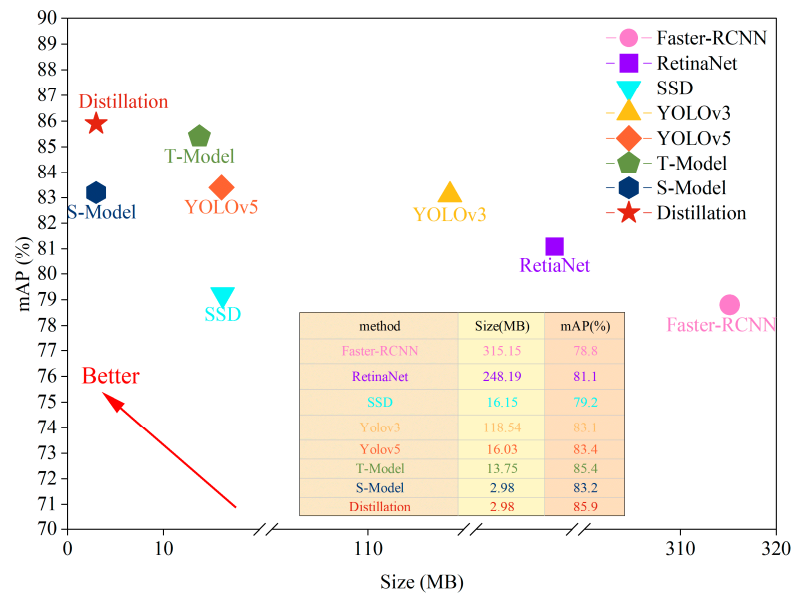


Figure 11. Comparison of model accuracy and number of parameters.

Table 6. Model performance comparison.

Model	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Precision (%)	Recall (%)	GFLOPS (G)	Size (MB)
Faster-RCNN	78.8	48.8	78.6	80.2	88.2	315.15
RetinaNet	81.1	49.8	79.7	81.3	70.3	248.19
SSD	79.2	47.0	75.2	82.1	15.2	16.15
Yolov3	83.1	49.1	80.3	81.6	154.6	118.54
Yolov5	83.4	47.9	75.4	82.4	16.8	16.03
T-Model	85.4	49.6	76.3	80.3	15.8	13.75
S-Model	83.2	46.7	74.8	80.9	2.3	2.98
Distillation	85.9	50.3	77.4	83.1	2.3	2.98

The loss function and prediction correctness, recall, and accuracy data during the training of the distilled model are recorded as shown in Figure 12, which shows that the model converges very quickly.

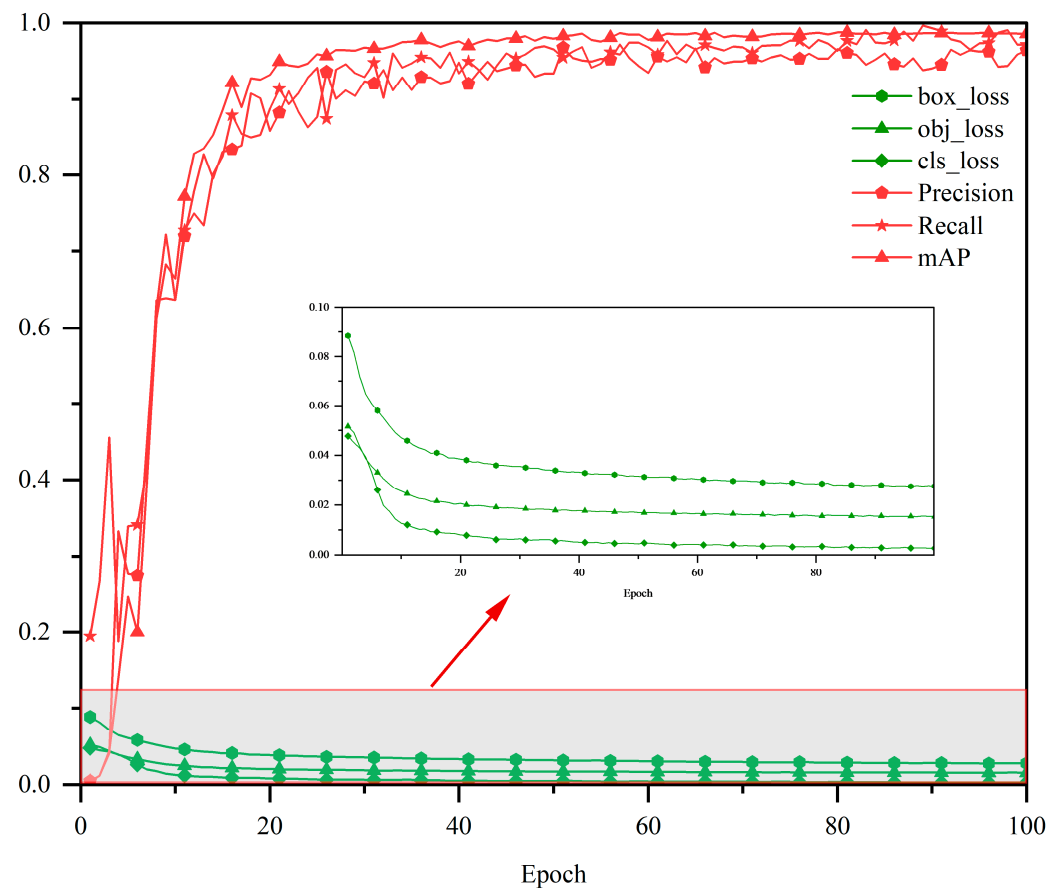


Figure 12. Performance of a multidimensional knowledge distillation compression model, with the red portions representing accuracy, recall, and prediction rates; the higher they are, the better, and the green portions representing loss functions; the lower they are, the better.

The dataset was tested in this study, and the test results are presented through a visual display in Figure 13. An in-depth analysis of Tables 5 and 6 and Figures 7–12 reveals that the detection efficiency of the improved algorithm relative to the original algorithm has been significantly improved, with the number of its references reduced by 81.4%, from 16.03 MB to 2.98 MB, while the accuracy has been improved by 2.5%, from 83.4% to 85.9%; the prediction accuracy for various types of defects has also been significantly improved, and the matching between the detection frame and the target to be detected has also been significantly improved; at the same time, the algorithm proposed in this study can detect small targets that were missed or wrongly detected by the original algorithm. These improvements highlight the superiority and effectiveness of the proposed algorithm compared to existing algorithms.

Figure 14 depicts the experimental findings from this study's use of the publicly available VOC data set, which demonstrated that the improved model is In the VOC data set, the multi-location knowledge distillation model's detection accuracy In the VOC data set, the multi-location knowledge distillation model's detection accuracy increased by 3.6% from the original model's 59.9% to 63.5%.

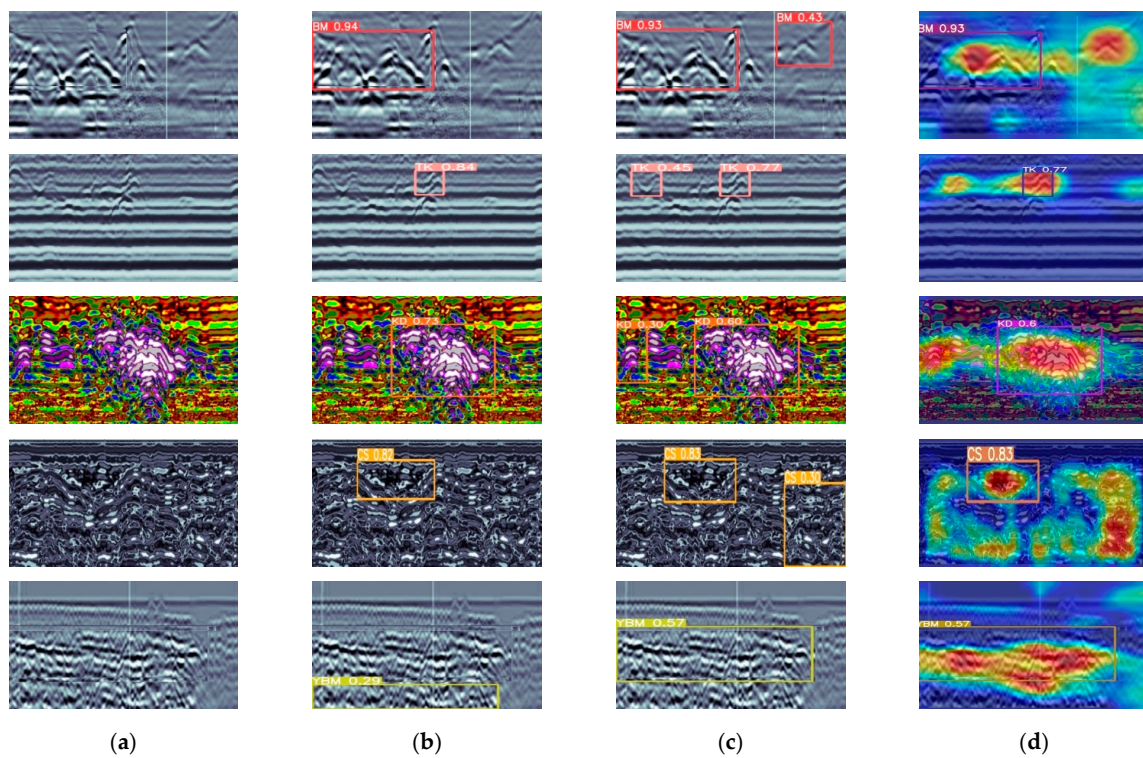


Figure 13. Detection result graphs where (a) is the original defect map, (b) is the original model detection result map, (c) is the improved multi-dimensional knowledge distillation model detection result map, and (d) is the result visualization map.

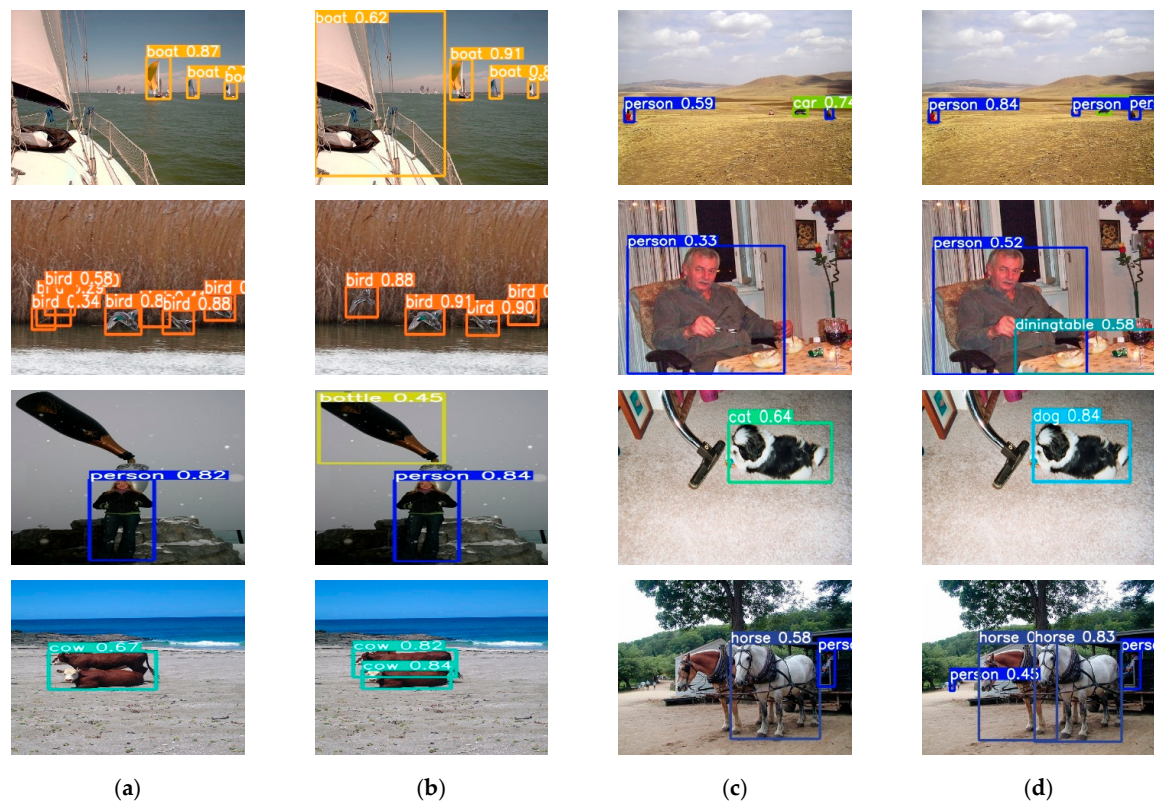


Figure 14. Performance results of the multidimensional knowledge distillation model and the Yolo model on the VOC dataset Where (a,c) are the original model detection results map and (b,d) are the multidimensional knowledge distillation results map.

6. Conclusions

In this study, we suggest a multidimensional knowledge distillation and detection approach. First, we increase the detection accuracy of the teacher model using pooling and weighting operations using a deep pooling residual structure; next, we introduce the MobileNetv3 lightweight backbone network into the student model; and finally, we learn how to extract feature information from the feature layer and the output layer in order to transfer the teacher model. The experimental results show that the improved model achieves 85.9% accuracy in tunnel defect detection, an improvement of 2.5%, and the model size is 2.98 MB, a volume compression of 83.2%, which meets the current demand for lightweight tunnel defect detection engineering. This accuracy improvement was achieved by testing the model on the radar detection dataset.

In order to further improve the detection efficiency and robustness of the algorithms, future work will explore how to combine transfer learning with knowledge distillation techniques to achieve simultaneous migration across domains and models, and combining these two techniques can effectively improve the generalization ability and robustness of the models while reducing the training time and computational cost; in terms of lightweight, we will continue to investigate how to combine pruning quantization techniques with knowledge distillation techniques to further reduce the number of parameters and computation of the model, such as quantization-aware training, weight clustering, etc.; in terms of accuracy, we will focus on designing more effective model structures to improve model performance and loss functions.

Author Contributions: Conceptualization, A.Z., B.W. and C.M.; Methodology, A.Z., B.W. and C.M.; Software, B.W.; Validation, B.W.; Formal analysis, J.X.; Investigation, J.X.; Resources, A.Z. and C.M.; Data curation, A.Z. and J.X.; Writing—original draft, B.W.; Writing—review & editing, A.Z. and B.W.; Project administration, A.Z.; Funding acquisition, A.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Key Science and Technology Project of Henan Province (222102210135).

Data Availability Statement: The data used to support the findings of this study are included in the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Minardo, A.; Catalano, E.; Coscetta, A.; Zeni, G.; Zhang, L.; Di Maio, C.; Vassallo, R.; Coviello, R.; Macchia, G.; Picarelli, L. Distributed fiber optic sensors for the monitoring of a tunnel crossing a landslide. *Remote Sens.* **2018**, *10*, 1291. [\[CrossRef\]](#)
2. Monsberger, C.M.; Lienhart, W. Distributed fiber optic shape sensing of concrete structures. *Sensors* **2021**, *21*, 6098. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Afshani, A.; Kawakami, K.; Konishi, S.; Akagi, H. Study of infrared thermal application for detecting defects within tunnel lining. *Tunn. Undergr. Space Technol.* **2019**, *86*, 186–197. [\[CrossRef\]](#)
4. Dong, Y.; Wang, J.; Wang, Z.; Zhang, X.; Gao, Y.; Sui, Q.; Jiang, P. A deep-learning-based multiple defect detection method for tunnel lining damages. *IEEE Access* **2019**, *7*, 182643–182657. [\[CrossRef\]](#)
5. Xue, Y.; Li, Y. A fast detection method via region-based fully convolutional neural networks for shield tunnel lining defects. *Comput.-Aided Civ. Infrastruct. Eng.* **2018**, *33*, 638–654. [\[CrossRef\]](#)
6. Zhou, Z.; Zhang, J.; Gong, C. Automatic detection method of tunnel lining multi-defects via an enhanced You Only Look Once network. *Comput.-Aided Civ. Infrastruct. Eng.* **2022**, *37*, 762–780. [\[CrossRef\]](#)
7. Liu, Z.; Wu, W.; Gu, X.; Li, S.; Wang, L.; Zhang, T. Application of combining YOLO models and 3D GPR images in road detection and maintenance. *Remote Sens.* **2021**, *13*, 1081. [\[CrossRef\]](#)
8. Zhu, A.; Chen, S.; Lu, F.; Ma, C.; Zhang, F. Recognition Method of Tunnel Lining Defects Based on Deep Learning. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 9070182. [\[CrossRef\]](#)
9. Zhu, A.; Ma, C.; Chen, S.; Wang, B.; Guo, H. Tunnel Lining Defect Identification Method Based on Small Sample Learning. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 1096467. [\[CrossRef\]](#)
10. Liu, M.; Liu, C.; Chen, Y.; Yan, Z.; Zhao, N. Radio frequency fingerprint collaborative intelligent blind identification for green radios. *IEEE Trans. Green Commun. Netw.* **2022**, *7*, 940–949. [\[CrossRef\]](#)

11. Liu, M.; Zhang, Z.; Chen, Y.; Ge, J.; Zhao, N. Adversarial attack and defense on deep learning for air transportation communication jamming. *IEEE Trans. Intell. Transp. Syst.* **2023**, 1–14. [\[CrossRef\]](#)
12. Huang, Q. Weight-quantized squeezenet for resource-constrained robot vacuums for indoor obstacle classification. *AI* **2022**, 3, 180–193. [\[CrossRef\]](#)
13. Hu, W.; Che, Z.; Liu, N.; Li, M.; Tang, J.; Zhang, C.; Wang, J. Channel Pruning via Class-Aware Trace Ratio Optimization. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, 1–13. [\[CrossRef\]](#)
14. Zhang, S.; Qiang, B.; Yang, X.; Wei, X.; Chen, R.; Chen, L. Human Pose Estimation via an Ultra-Lightweight Pose Distillation Network. *Electronics* **2023**, 12, 2593. [\[CrossRef\]](#)
15. Zhao, Z.; Su, S.; Wei, J.; Tong, X.; Gao, W. Lightweight Infrared and Visible Image Fusion via Adaptive DenseNet with Knowledge Distillation. *Electronics* **2023**, 12, 2773. [\[CrossRef\]](#)
16. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531.
17. Romero, A.; Ballas, N.; Kahou, S.E.; Chassang, A.; Gatta, C.; Bengio, Y. Fitnets: Hints for thin deep nets. *arXiv* **2014**, arXiv:1412.6550.
18. Zagoruyko, S.; Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv* **2016**, arXiv:1612.03928.
19. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 779–788.
20. Zhang, J.; Chen, H.; Yan, X.; Zhou, K.; Zhang, J.; Zhang, Y.; Jiang, H.; Shao, B. An Improved YOLOv5 Underwater Detector Based on an Attention Mechanism and Multi-Branch Reparameterization Module. *Electronics* **2023**, 12, 2597. [\[CrossRef\]](#)
21. de Moraes, J.L.; de Oliveira Neto, J.; Badue, C.; Oliveira-Santos, T.; de Souza, A.F. Yolo-Papaya: A Papaya Fruit Disease Detector and Classifier Using CNNs and Convolutional Block Attention Modules. *Electronics* **2023**, 12, 2202. [\[CrossRef\]](#)
22. Liu, Y.; Chu, H.; Song, L.; Zhang, Z.; Wei, X.; Chen, M.; Shen, J. An improved tuna-YOLO model based on YOLO v3 for real-time tuna detection considering lightweight deployment. *J. Mar. Sci. Eng.* **2023**, 11, 542. [\[CrossRef\]](#)
23. Xiao, P.; Xu, T.; Xiao, X.; Li, W.; Wang, H. Distillation Sparsity Training Algorithm for Accelerating Convolutional Neural Networks in Embedded Systems. *Remote Sens.* **2023**, 15, 2609. [\[CrossRef\]](#)
24. Hou, S.; Tuerhong, G.; Wushouer, M. UsbVisdaNet: User Behavior Visual Distillation and Attention Network for Multimodal Sentiment Classification. *Sensors* **2023**, 23, 4829. [\[CrossRef\]](#)
25. Wang, L.; Zhang, Y.; Xu, Y.; Yuan, R.; Li, S. Residual Depth Feature-Extraction Network for Infrared Small-Target Detection. *Electronics* **2023**, 12, 2568. [\[CrossRef\]](#)
26. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
27. Wu, X.; Shi, H.; Zhu, H. Fault Diagnosis for Rolling Bearings Based on Multiscale Feature Fusion Deep Residual Networks. *Electronics* **2023**, 12, 768. [\[CrossRef\]](#)
28. Muhammad, W.; Bhutto, Z.; Ansari, A.; Memon, M.L.; Kumar, R.; Hussain, A.; Shah, S.A.R.; Thaheem, I.; Ali, S. Multi-path deep CNN with residual inception network for single image super-resolution. *Electronics* **2021**, 10, 1979. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.