ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

Based on LAMP and Channel Pruning and Knowledge Distillation Model Algorithm

Jianhao Liang[1,*]

[1]School of Mechanical Engineering, Xihua University, Chengdu, 611730, China

[*] Corresponding author:(Email: 1466169996@qq.com)

Abstract

To solve the problem of limited computing power of mobile devices for tea picking robots, this study proposed a hybrid compression algorithm based on LAMP pruning and channel pruning, and combined with knowledge distillation technology to optimize the performance of the lightweight model. The LAMP-adaptive normalization strategy of LAMP pruning was used to balance the proportion of pruning at different levels to avoid the over-pruning problem of traditional amplitude pruning. The channel pruning is further introduced to eliminate redundant convolution kernels, reducing the computational effort by 37.3% and the parameter count by 20.3%. The Mimic loss and linear attenuation distillation strategies were proposed, and the reasoning speed was increased by 19.7% at the pruning rate 1.7 through weighted multi-layer feature difference and dynamic adjustment of loss weight. Experiments show that the proposed method reduces the average delay to 8.26ms while maintaining the accuracy of the model, which is significantly better than the traditional pruning methods such as L1 and group_taylor, and provides an efficient solution for real-time tea recognition on edge devices.

Keywords

Model pruning, Distillation of knowledge, Lightweight model.

1. INTRODUCTION

With the acceleration of the intelligent process of agriculture, the visual recognition system relied on by tea picking robots needs to realize high-precision real-time detection on embedded devices. [1]However, the current mainstream detection models such as RT-DETR and YOLOv8 have a large number of parameters and high computational complexity, and are difficult to be directly deployed in mobile terminals with limited computing power. Industry practice shows that the tea picking scenario puts double constraints on the model performance: 1. Field operation requires a detection delay of less than 10ms to meet the real-time response of the robot arm; 2. High mAP50 recognition accuracy should be maintained under complex lighting and branches and leaves occlusion. The existing lightweight methods are faced with serious challenges: the accuracy of traditional structured pruning drops sharply at high pruning rate due to insufficient global sensitivity evaluation; Although the knowledge distillation technique can alleviate the accuracy loss, the large capacity gap between the teacher-student model significantly weakens the feature transfer effect.[2]

To solve the above problems, a hybrid compression framework combining LAMP pruning, channel pruning and dynamic knowledge distillation was proposed in this paper. The lamp-channel joint pruning mechanism was designed. The upper limit of pruning rate of each layer was determined by LAMP pruning, and the redundant convolution kernel was removed

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

combined with channel importance scoring. Based on Mimic loss function, the similarity measure of multilayer feature maps and the divergence of output distribution KL were combined to strengthen the feature alignment ability of the teacher-student model after pruning. The linear attenuation distillation weight strategy is designed to focus on the feature imitation loss in the initial stage of training, and gradually transition to the detection task loss in the later stage to alleviate the optimization conflict caused by the model capacity gap.

2. PRUNING METHOD OF MODEL

2.1. Principle of pruning method

In the past few years, deep learning has broken through many technical problems in computer vision, natural language processing and other fields. However, its training usually involves a large number of parameters and calculations, and its application scope is limited in edge devices and low-power application scenarios. Model compression technology has developed rapidly as a research direction to solve this problem . One approach is to eliminate duplicate weights and neurons by pruning to reduce computation and storage costs, as shown in Figure 1. Pruning directly deletes the whole neuron or channel to adjust the computational structure of the model, which can effectively reduce the computational complexity, speed up the inference time, and adapt to the optimal parallel computation optimization of small-scale hardware.

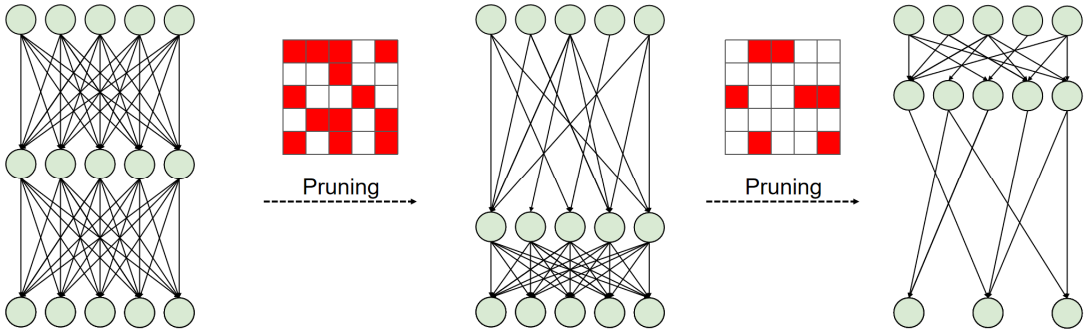


Figure 1. Pruning Schematic Diagram

Magnitude-based Pruning (MP) is a traditional pruning method. Its basic idea is to compress the model by removing smaller absolute weights in a fixed proportion layer by layer. However, this crude and direct approach ignores the difference in weight distribution and sensitivity to the different levels of the neural network, and may destroy the basic architecture of the model during the pruning process. The weights of some levels may be in the smaller absolute range, but they are critical for feature extraction or information transfer, and if pruned at the same ratio as the other levels, the important parameters may be over-pruned, resulting in a sharp decline in model performance.

The core innovation of LAMP is to add a layer by layer adaptive dynamic adjustment mechanism, through the overall L2 norm of the weight of each layer to normalize the absolute weight of each layer, so as to normalize the weight of different layers to the same level, equivalent to creating a "fair playing field" for each layer. There is no problem of excessive pruning parameters in some layers due to low overall weight. The key of this method is to reduce the weight disturbance between the pruned model and the original model, so as to keep

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

the whole function of the pruned model as much as possible. The performance of the pruning model is improved by making the pruning percentages of different layers more consistent with the initial pruning.

2.2. LAMP Compression Model Combined With Channel Pruning

The core idea of LAMP pruning is to dynamically adjust the pruning ratio according to the amplitude of neurons in each layer. Compared with MP pruning, the balance between layers is considered in the pruning process, so that the pruning can carry out dynamic pruning according to the characteristics of different layers, avoid redundant calculation, and preserve the key structure of the network to the maximum extent [3].

Although LAMP pruning can effectively reduce the number of model parameters, it does not directly optimize the computational complexity. For convolutional neural networks, the computational complexity depends heavily on the number of convolutional layer channels. Unlike weight pruning, channel pruning directly eliminates unimportant convolution filters to reduce overhead and memory requirements. Both channel pruning and LAMP try to reduce the compute and memory requirements of the model, but the goals of pruning are different. LAMP pruning evaluates the importance of each neuron by calculating the amplitude of each neuron and dynamically adjusts the pruning rate according to the contribution of neurons, while channel pruning reduces the computational burden of the network by deleting redundant convolutional channels and pruning multiple neurons in the convolutional layer. In order to further improve the efficiency of model compression, an integrated compression algorithm based on weight pruning and channel pruning is proposed to obtain more efficient model compression. Combining the two can optimize the network architecture more comprehensively without reducing model accuracy and computational efficiency. The architecture is shown in Figure 2.

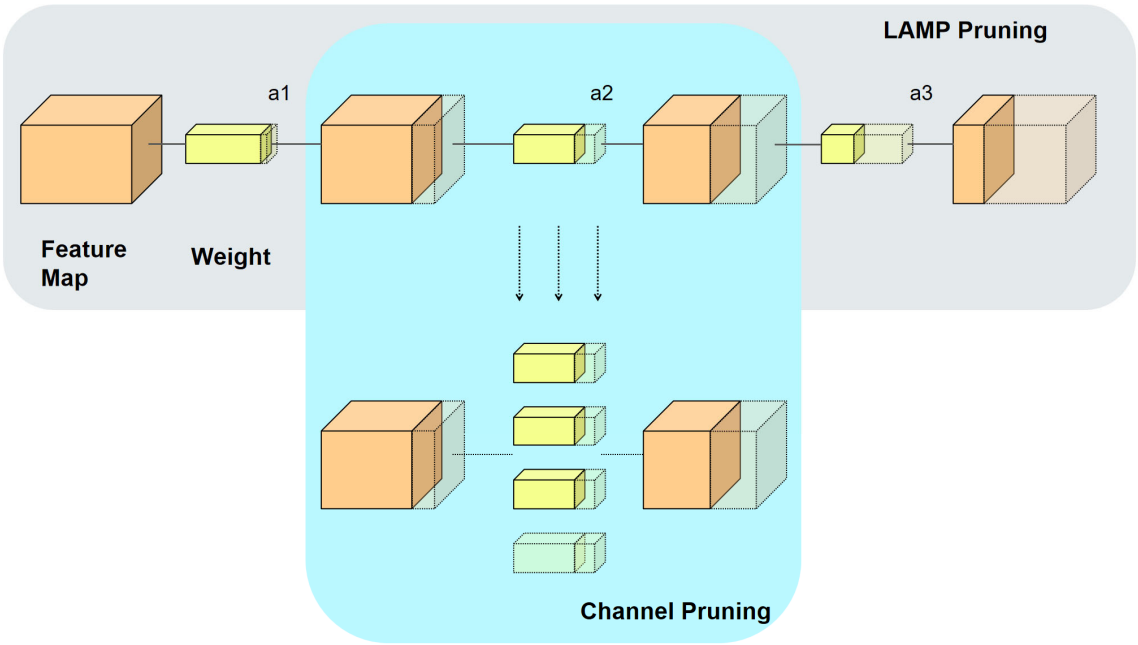


Figure 2. Hybrid Pruning Flowchart

The hybrid pruning method can give full play to the advantages of fine grain pruning of unstructured LAMP and integrate the advantages of structured channel pruning. LAMP pruning filters weights based on layer adaptive strategies to remove redundant connections and reduce the parameter scale. On this basis, channel pruning is further carried out to remove the entire

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

channel with low contribution, and the calculation cost is greatly reduced. It ensures high model accuracy, computational acceleration and memory optimization, making it more suitable for edge computing or low-power device Settings.

Let the weight matrix of layer l of the neural network be represented as W^l, where $W^l \in \mathbb{R}^{C_{out} \times C_{in} \times K_h \times K_w}$, C_{out} and C_{in} represent the number of output channels and the number of input channels respectively, K_h and K_w are the height and width of the convolution kernel. For single weight pruning, the traditional MP pruning method uses L1 or L2 norm to measure the importance of weight, namely:

$$M^l = ||W^l||_p = \left(\sum_{i,j,k,h} \left|W^l_{i,j,k,h}\right|^p\right)^{\frac{1}{p}} \tag{1}$$

LAMP pruning method was improved on this basis, and interlayer normalization factor α^l was introduced to optimize pruning decision.

Given the weight matrix W^l of a layer of the neural network, where each weight $w^l_{i,j,k,h}$ represents the weight of the convolution kernel between the output channel i and the input channel j at position (k, h). Then its amplitude is defined as:

$$|W^l| = \{\left|w^l_{i,j,k,h}\right|\}, \quad \forall i, j, k, h \tag{2}$$

Calculate the sum of the weight ranges for each layer:

$$S^l = \sum_{i,j,k,h} \left|W^l_{i,j,k,h}\right| \tag{3}$$

The weight amplitude of all layers is normalized, and the normalization factor α^l is obtained:

$$\alpha^l = \frac{S^l}{\sum_k S^k} \tag{4}$$

The normalized factor is used to dynamically adjust the pruning ratio of each layer, so that the pruning process can be optimized according to the importance of the layer. The LAMP method uses this factor to calculate the layer adaptive pruning threshold T^l_p:

$$T^l_p = \alpha^l \cdot percentile(W, p) \tag{5}$$

Where, p is the set pruning rate, used to control the weight ratio to be cut off; $percentile(W, p)$ calculates the p quantile of the weight amplitude of the layer;

For convolution layer l, which contains multiple channels $C^l = \{C^l_1, C^l_2, \ldots, C^l_N\}$, the importance of each channel C^l_i is calculated from the cumulative amplitude of all relevant weights:

$$P_i = \frac{\sum_{j \in L_i} \text{Magnitude}(i,j)}{\sum_{j \in L_i} \text{Magnitude}(i,j) + \lambda} \tag{6}$$

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

Where, $Magnitude(i,j)$ calculates the sum of the amplitude of the ownership weight in channel i; λ is the flexibility of the balance factor to adjust the ratio of pruning to each layer.

For all channel scores P_i, calculate a pruning threshold T_c:

$$T_c = percentile(P, p_c) \tag{7}$$

Pruning inevitably leads to a loss of accuracy, and the fine-tuning process is used to reduce the accuracy due to its loss. Fine-tuning keeps the learning rate at a low level to avoid violent perturbations to the pruned architecture, controls the model complexity and adjusts the pruned network architecture by adjusting the regularization terms.

The model was optimized by LAMP pruning and then channel pruning. LAMP pruning can remove duplicate neurons and channel pruning can remove duplicate convolutional channels. The combination of the two can reduce the computational cost and memory usage of the model, and can flexibly respond to different network structures and task requirements.

3. KNOWLEDGE DISTILLATION PRINCIPLE AND METHOD INNOVATION

The previous section describes an efficient compression algorithm combining LAMP and channel pruning, which significantly reduces the computational cost and storage capacity requirements of the model and improves the execution efficiency of the model on low-computing devices. Although the pruning method can greatly reduce the number of model parameters and the amount of calculation, the model accuracy is usually lost in the pruning process [4]. In order to further improve the performance of the model after pruning, knowledge distillation technology is used in this paper, which can balance the accuracy loss during pruning process and improve the accuracy and reasoning ability of the model after pruning.

This section introduces knowledge distillation technology and its application in model pruning optimization, which can not only improve the accuracy of the model after pruning, but also achieve higher accuracy and efficiency when the model is run on resource-limited equipment.

3.1. Theoretical Framework and Application Status of Knowledge Distillation

With the widespread application of deep neural networks in the field of computer vision, the model size and computational cost are increasing, making it particularly difficult to deploy such models on resource-constrained platforms [5]. As a result, knowledge distillation has attracted more and more attention as a robust model compression and acceleration technique, as shown in Figure 3. The core idea is to use the teacher model with better recognition rate to train a small student model, so that the student model can approach the performance of the teacher model without too much complexity. The process of distillation is mainly to narrow the gap between the output of the teacher model and the output of the student model, and optimize the student model with a replacement loss function so that it can imitate the teacher model knowledge [6].

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

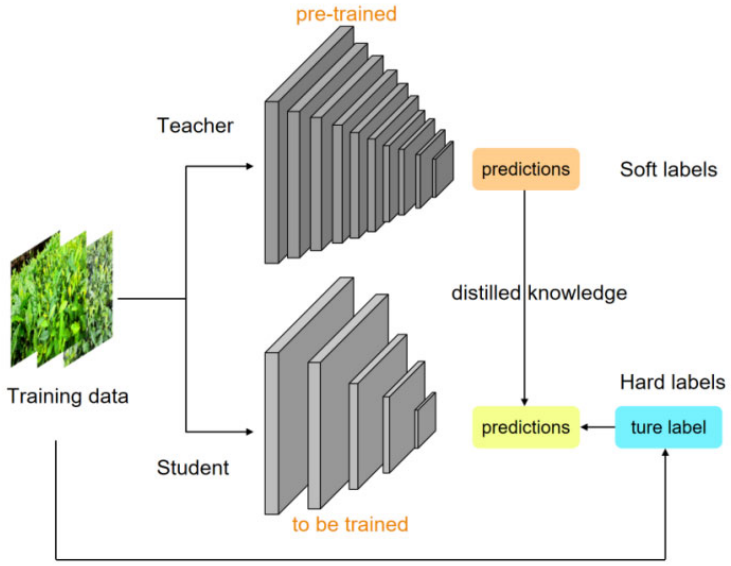


Figure 3. Knowledge Distillation

The distillation loss function usually includes the logical distillation loss and the characteristic distillation loss. The logical distillation mainly guides the training of the student model through the difference between the teacher model and the student model in the final output layer. In the traditional classification task, the teacher model generates the probability distribution through softmax function, and the student model also generates the corresponding probability distribution. The difference between these two distributions is usually measured using KL (Kullback-Leibler) divergence, which measures the relative amount of information in two probability distributions. The formula is:

$$L_{\log} = \mathrm{KL}(p_t \parallel p_s) = \sum_i p_t(i) \log \frac{p_t(i)}{p_s(i)} \tag{8}$$

Where, $p_t(i)$ and $p_s(i)$ are the predicted probabilities of the teacher and student models for the class, respectively. By minimizing the KL divergence, the student model can learn the category prediction of the teacher model and the judgment of similarity between categories.

In order to make the student model smoother in the process of imitating the teacher model, A temperature parameter T is usually introduced to smooth the probability distribution of the teacher model output. The output distributions p_t and p_s after scaling the temperature parameter are:

$$p_t = \frac{\exp(z_t/T)}{\sum_j \exp(z_{t,j}/T)}, \qquad p_s = \frac{\exp(z_s/T)}{\sum_j \exp(z_{s,j}/T)} \tag{9}$$

Where z_t and z_s are the unnormalized outputs of the teacher and student models respectively, and the temperature parameter T controls the smoothness of the output probability distribution. Minimizing the KL divergence after temperature smoothing can help the student model learn the details of the teacher model in dealing with the similarity between categories, and improve the classification ability of the student model.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

3.2. Distillation realization based on Mimic loss and linear attenuation strategy

In the tea recognition task, the model generally has a high degree of feature representation, which is very critical for the object frame regression task. In order to let the student model learn the knowledge of the teacher model in the middle layer, this paper guides the training of the student model by adding Mimic loss to compare the difference between the student model and the teacher model in the middle layer.

In feature distillation, we assume that the feature representation of the teacher model at layer B is C, and the feature representation of the student model at the same layer is D. By calculating the mean square error (MSE) between the two feature representations, the distillation loss of this layer is obtained:

$$L_{\text{fea}}^{(i)} = \| f_s^{(i)} - f_t^{(i)} \|_2^2 \tag{10}$$

In multilayer characteristic distillation, the distillation losses of all layers are treated in the same way. The student model may overlearn low-level features during training, and the contribution of different layers of feature learning to the task is uneven. Therefore, the loss of each layer is weighted, and the learning focus of the student model is adjusted according to the feature importance of different levels during training. The weighting process can facilitate the model to learn more effectively and transfer the expertise of the teacher model, with different levels of features contributing differently to the overall performance of the model in the context of complex tasks. High-level features contain more semantic information, while low-level features are more informative. In order to improve the detection and generalization performance of the model, the definition of weighted characteristic distillation loss can be expressed as:

$$L_{\text{fea}} = \sum_{i=1}^{N} \alpha_i \cdot L_{\text{fea}}^{(i)} = \sum_{i=1}^{N} \alpha_i \cdot \| f_s^{(i)} - f_t^{(i)} \|_2^2 \tag{11}$$

Through this weighting mechanism, the model can pay more attention to the feature layer which has a greater impact on the final task in the distillation process. Higher-level features near the output tend to contain more semantic information, so higher weights can be set for these layers, while lower-level features focus on more detailed information, so their weights can be appropriately reduced. Multi-layer feature weighted distillation is shown in Figure 4.

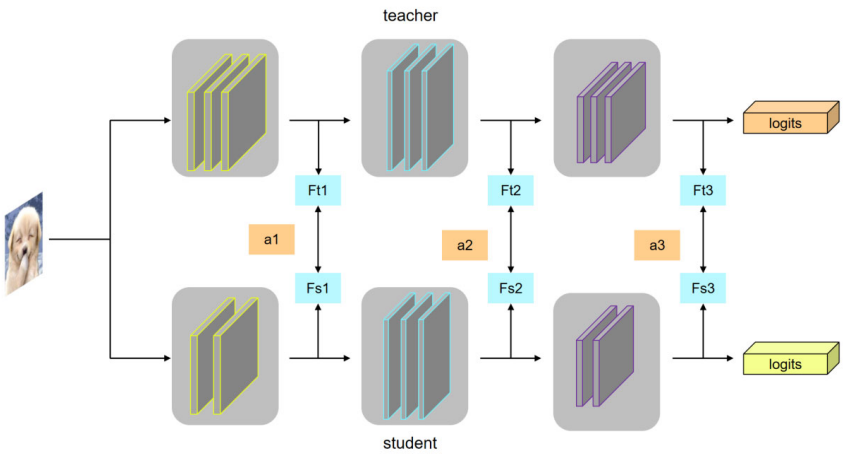


Figure 4. Multi-layer Feature Weighting Distillation

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

In this paper, Mimic loss is selected as a distillation strategy to effectively transfer the features of the middle layer of the teacher model to the student model, and the student model can gradually learn the key information of the teacher model at different levels by continuously minimizing the loss.

In knowledge distillation, the learning process of the student model is often based on the output or feature representation of the teacher model. The teacher model provides complete guidance at the initial stage of training, so that the student model can learn basic knowledge and structure in a limited time. However, with the progress of training, the ability of the student model will gradually improve, and over-reliance on the guidance of the teacher model will limit the student model's independent learning ability. Aiming at the balance between the guidance of the teacher model and the independent learning ability of the student model, this paper introduces a linear attenuation method to gradually reduce the weight of distillation loss, so that the student model can make full use of the knowledge of the teacher model in the early stage of training, and rely more and more on its data learning ability in the later stage of training, so as to improve the autonomy and generalization ability of the student model [7].

The core idea of linear attenuation method is that the contribution of distillation loss decreases with the increase of training, so the student model weakens its dependence on the teacher model with the increase of training, and enhances its ability of independent learning. Usually in the initial stage of learning, the student model has poor learning ability, and the teacher model professor plays an important role. We therefore expect the weight of distillation losses to be greater in the early stages; In the later stages of training, as the student model learns more and more, the weight of distillation loss decreases, and the student model becomes more and more dependent on the acquired features and patterns.

The attenuation factor $D(t)$ is designed to control the weight of distillation losses in each training batch or round, decreasing linearly from 1 to near 0. Assuming that the total training cycle is T and the current training cycle is t, the attenuation factor can be calculated by the following formula:

$$D(t) = 1 - \frac{t}{T} \tag{12}$$

The weight of distillation loss is multiplied by attenuation factor $D(t)$ to adjust its contribution to the total loss function. By introducing attenuation factor, the calculation formula of distillation loss can be expressed as:

$$L_{\text{fea}}^{\text{adjusted}} = D(t) \cdot L_{\text{fea}} \tag{13}$$

Under the influence of the linear attenuation strategy, the student model can learn more knowledge from the teacher model in the early stage, and in the later stage of training, the student model can gradually exert its own learning ability to improve its independence and generalization ability.

4. ANALYSIS OF EXPERIMENTAL RESULTS

4.1. Experimental Environment

The experiment was conducted on the Windows 10 platform, using Python 3.8 as the programming language, and the deep learning library PyTorch to implement the object detection task. In terms of hardware, this paper uses the 13th generation Intel Core i9-13900HX

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

as the central processor, NVIDIA GeForce RTX 4080 Laptop GPU as the graphics card, with 12GB of video memory and 12.0 CUDA version, providing powerful computing power for model training and reasoning. Experimental training parameters are shown in Table 1:

Table 1. Training Parameters

Index	parameter
Image resolution	640×640
Training rounds	300
Learning rate	0.0001
Training batch	4

4.2. Evaluation Index

This paper mainly compares the pruning effect from three aspects: precision, complexity and speed. In terms of accuracy, average precision mean mAP is used. The complexity is measured by FLOPs and the number of model Parameters, while the speed is measured by FPS and average inference time. The experimental results are shown in Table 2.

Table 2. Experimental Results of Different Pruning Methods

method	Ratio	mAP50(%)	GFLOPs	Parameters	FPS	Average delay (ms)
		0.871	54.4	16019172	49.6	12.96
Lamp+Cl	1.3	0.855	38.9	13166188	55.8	10.13
Lamp+Cl	1.5	0.842	35.3	12892196	58.1	9.50
Lamp+Cl	1.7	0.840	34.1	12768324	59.4	8.26
Lamp	1.3	0.861	41.5	14077252	43.3	15.13
Lamp	1.5	0.851	36.2	13163548	51.9	12.27
Lamp	1.7	0.846	35.5	12930188	54.3	11.01
L1	1.3	0.863	41.4	14261764	43.1	15.25
L1	1.5	0.838	35.6	12959028	43.7	14.98
L1	1.7	0.818	33.2	12548732	49.7	12.92
group_taylor	1.3	0.853	40.2	13741844	54.2	11.09
group_taylor	1.5	0.828	35.6	12943396	48.2	13.24
group_taylor	1.7	0.830	35.5	12922844	50.0	13.50

The experimental results show that the calculation efficiency of LAMP (Lamp+Cl) method with channel pruning is greatly improved. Although mAP50(%) is lower than that of traditional LAMP method, the overall performance is more balanced, and it is suitable for inference tasks requiring high computational efficiency. Lamp+Cl is superior to all other methods in reducing computational complexity and increasing reasoning speed. When the pruning ratio is 1.7, Lamp+Cl has 1,2768324 parameters, 34.1 GFLOPs and 59.6 FPS, which means that the model has more advantages in reducing the computing overhead, which can greatly save the running cost of the model in practical use, and can complete the inference in a shorter time. Meets the needs of low-latency applications and is more suitable for edge devices and low-resource Settings.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

Figure 5 shows the mAP value of the experimental distillation method when the pruning ratio is 1.7. Both Lamp+Cl and Lamp methods can maintain a good mAP, which is at a higher level than other pruning schemes. Although LAMP combined with channel pruning will lose less accuracy due to greatly reducing parameters, it can sacrifice a small part of detection capability in exchange for faster reasoning speed and calculation cost, which is closer to the requirements of rapid reasoning tasks. In the process of pruning, the network structure is better optimized and the network computing load is more balanced. The problem of unbalanced computing load that may be caused by traditional LAMP pruning is avoided.

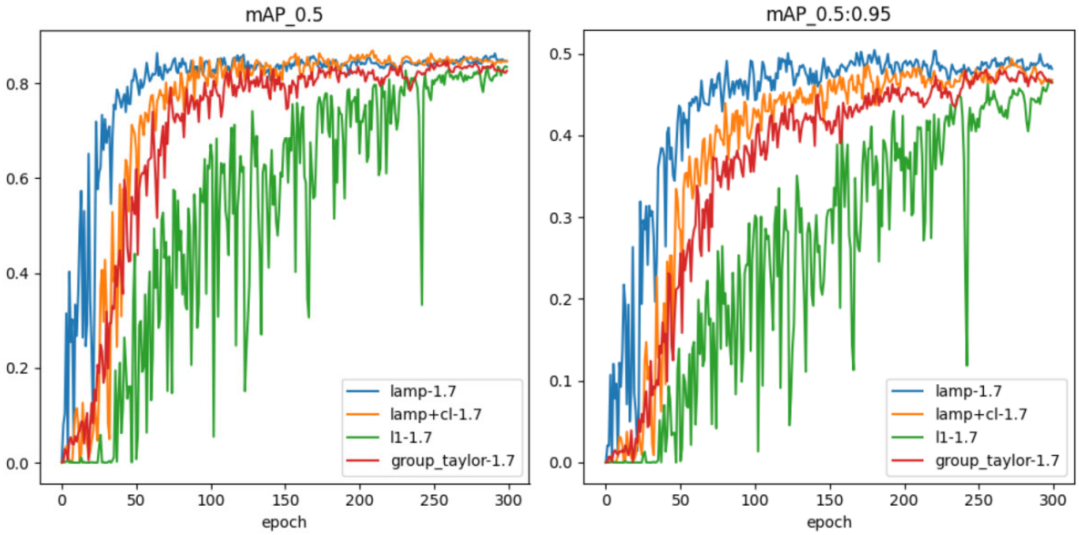


Figure 5. Comparison of Pruning mAP Results

In the distillation experiment, the distillation feature methods of Mimic, Mgd and Cwd were combined with the distillation of Constant, Cosine and Linear loss attenuation on the Lamp+Cl benchmark model with a pruning ratio of 1.7. The experimental results are shown in Table 3 and Figure 6. The effects of these methods on model accuracy, inference rate (FPS) and other indexes of model accuracy and computational efficiency were compared.

Table 3. Experimental Results of Different Distillation Methods

Characteristic method	loss attenuation	mAP50(%)	FPS
Cwd	Constant	0.800	56.7
Cwd	Cosine	0.824	55.2
Cwd	Linear	0.829	51.0
Mgd	Constant	0.820	59.2
Mgd	Cosine	0.837	58.5
Mgd	Linear	0.834	58.3
Mimic	Constant	0.840	56.7
Mimic	Cosine	0.852	57.8
Mimic	Linear	0.857	58.1

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

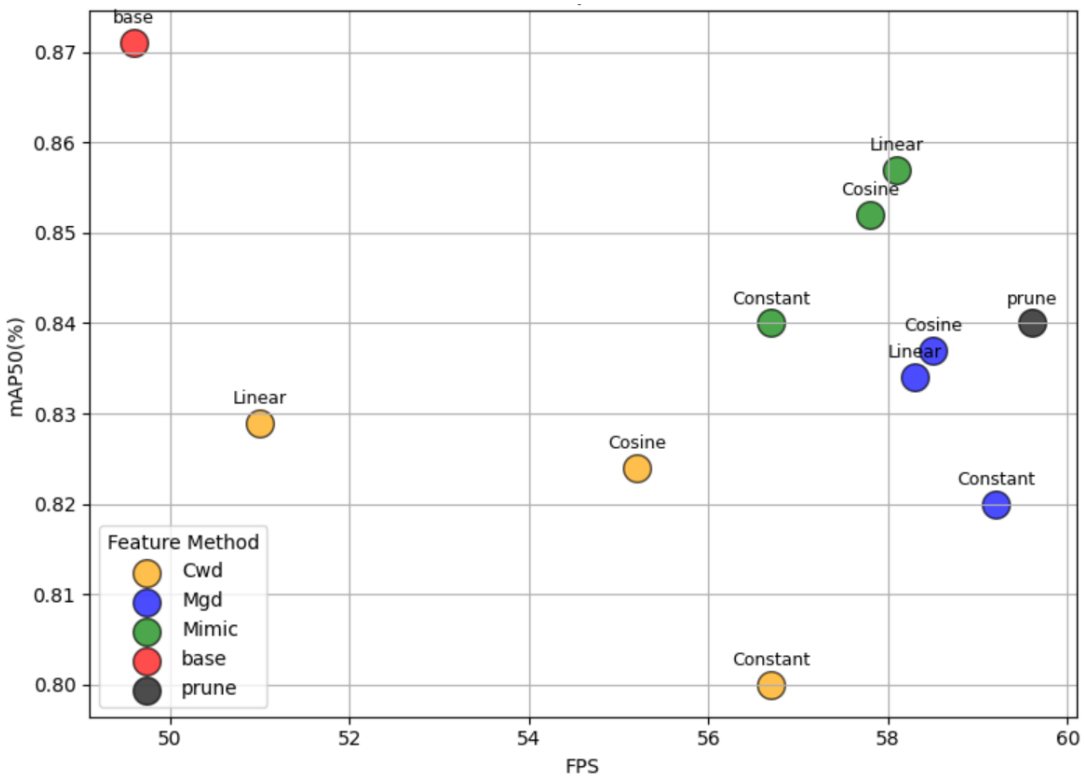


Figure 6. Experimental Results of Different Distillation Methods

The red circle in Figure 6 indicates that the basic model mAP50 is 0.871 and the FPS is 49.6, showing high accuracy and low inference speed. 6. Although the pruned model represented by the black circle has been pruned, FPS is 59.4, showing a good reasoning speed, but the accuracy mAP50 is reduced to 0.840. The loss of model expression ability was mainly due to the reduction of some parameters and calculation amount in the process of pruning.

5. CONCLUSION

Mimic+Linear can achieve high-speed inference and high detection accuracy, and the method is significantly superior to the pre-pruning model and other distillation techniques on the same pruning model. In practical application, this scheme will be more suitable for the situation of limited computing resources, and play a key role in the deployment of tea picking robots.

REFERENCES

[1] C. Zhang, J. Wang, T. Yan, X.H. Lu, G.D. Lu, X.L. Tang, B.C. Huang, An instance-based deep transfer learning method for quality identification of Longjing tea from multiple geographical origins, Complex Intell. Syst. 9(3) (2023) 3409-3428. https://doi.org/10.1007/s40747-023-01024-4.

[2] J. Yang, Y. Chen, Tender Leaf Identification for Early-Spring Green Tea Based on Semi-Supervised Learning and Image Processing, Agronomy-Basel 12(8) (2022) 13. https://doi.org/10.3390/agronomy12081958.

[3] M. Zhu, S. Gupta, To prune, or not to prune: exploring the efficacy of pruning for model compression, (2017).

[4] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, International Conference on Learning Representations, 2021.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202504_11(4).0018

[5] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin Transformer: Hierarchical Vision Transformer using Shifted Windows, (2021).

[6] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, S. Zagoruyko, End-to-End Object Detection with Transformers, 2020.

[7] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, J. Dai, Deformable DETR: Deformable Transformers for End-to-End Object Detection, International Conference on Learning Representations, 2021.