modelling

MDPI

Article

Cloud-Edge Collaborative Inference-Based Smart Detection Method for Small Objects

Cong Ye [1], Shengkun Li [2,*], Jianlei Wang [1], Hongru Li [1], Xiao Li [1] and Sujie Shao [2]

[1] Information & Communication Company of State Grid Ningxia Electric Power Co., Ltd., Yinchuan 750001, China; congye2025@163.com (C.Y.); wangjianlei2025@163.com (J.W.); lhongru2025@163.com (H.L.); hainalix60@gmail.com (X.L.)
[2] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100080, China; buptssj@bupt.edu.cn
* Correspondence: lishengkun@bupt.edu.cn

Abstract

Emerging technologies are revolutionizing power system operation and maintenance. Intelligent state perception is pivotal for stable grid operation, with small object detection technology being vital for identifying minor hazards in power facilities. However, challenges like small object size, low resolution, occlusion, and low confidence arise in small object detection for power operation and maintenance. This paper proposes PyraFAN, a feature fusion method designed for small object detection, and introduces a cloud-edge collaborative inference based smart detection method. This method boosts detection accuracy while ensuring real-time performance. Additionally, a graph-guided distillation method is developed for edge models. By quantifying model performance and task similarity, multi-model collaborative training is realized to improve detection accuracy. Experimental results show that compared with standalone edge models, the proposed method improves detection accuracy by 6.98% and reduces the false negative rate by 19.56%. The PyraFAN module can enhance edge model detection accuracy by approximately 12.2%. Updating edge models via cloud model distillation increases the mAP@0.5 of edge models by 2.7%. Compared to cloud models, the cloud-edge collaboration method reduces average inference latency by 0.8%. This research offers an effective solution for improving the accuracy of deep learning based small object detection in power operation and maintenance within cloud-edge computing environments.

Keywords: small object detection; feature fusion; cloud-edge collaboration; knowledge distillation; YOLO; confidence distinction

check for updates

Academic Editor: Wei Gao

Received: 10 August 2025
Revised: 22 September 2025
Accepted: 23 September 2025
Published: 24 September 2025

Citation: Ye, C.; Li, S.; Wang, J.; Li, H.; Li, X.; Shao, S. Cloud-Edge Collaborative Inference-Based Smart Detection Method for Small Objects. *Modelling* **2025**, *6*, 112. https://doi.org/10.3390/modelling6040112

Copyright: © 2025 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

Against the backdrop of the era where information technology profoundly empowers critical infrastructure, the Internet of Things, edge computing, and cloud computing technologies are profoundly transforming the operation and maintenance management modes of power systems. As the core of national energy security, the stable and efficient operation of power grids highly depends on intelligent state perception and defect identification capabilities. Among them, small object detection technology is crucial for discovering tiny yet fatal hidden dangers in power facilities, boasting significant research value and engineering application demands [1,2].

In practical scenarios of power Augmented Reality (AR) operation and maintenance, such as high-voltage transmission line inspection, operation and maintenance personnel

often use intelligent terminals like augmented reality (AR) glasses for auxiliary operations. They need to accurately overlay identified tiny foreign objects (such as fallen insulator fragments, missing pins) or early defects (such as tiny cracks and rust on equipment surfaces) onto the real world in the form of three-dimensional annotations. This places higher requirements on the real-time performance, accuracy, and interactivity of detection algorithms [3–5]. The missed detection or false detection of these small objects not only affects the usability of the AR system but may also lead to serious consequences such as flashovers, wire breakages, equipment failures, and even large-scale power outages, resulting in huge economic losses and social impacts.

However, practical small object detection faces challenges such as small object size, low effective resolution, susceptibility to occlusion, and low detection confidence, which limit detection accuracy in real-world settings [6,7]. Current research mainly tackles these issues through architectural changes, like boosting model capacity for small scale pattern recognition or enhancing robustness to partial occlusions. Yet, these model centered optimizations are not enough for cloud-edge collaborative deployments with resource heterogeneity and contextual differences [8,9]. Edge devices with limited computing resources face high latency, overheating, and power consumption issues when running complex detection models. The conventional cloud centric processing model brings high bandwidth costs and can't utilize edge captured location sensitive data [10–15].

Moreover, there is the problem of edge feature generalization [16–18]. Models at the edge server adapt to local data distributions, such as road features in intelligent transportation systems or regional features in security monitoring networks. While edge tuned models are precise locally, they lack generalizability. Conversely, cloud models trained on multi-source edge data have broad feature representations but lose local specificity. This suggests that using localized features before cloud generalization can improve edge specific accuracy. Thus, a synergistic cloud-edge collaborative framework is promising. Deploying lightweight models (optimized via pruning or knowledge distillation) on edge devices enables real-time, location aware data processing, while the cloud offers intensive computational support for deeper feature analysis or model refinement.

To address small object detection challenges in cloud-edge-end scenarios, we propose a method combining the PyraFAN module with cloud-edge collaborative inference. Raw data from end devices is first streamed to edge servers, where redundant frames are filtered and keyframes are retained. A lightweight local model then performs initial detection on processed frames and delivers real-time results. Low-confidence or occluded detections are offloaded to the cloud for high-precision inference using computationally intensive models, enhancing decision-making robustness. Additionally, cloud servers aggregate data from edge nodes and periodically perform knowledge distillation to transfer refined feature representations to edge models, improving their local detection capabilities. To further optimize edge performance, we introduce a graph-guided distillation method among edge models. By modeling spatial, task, and model similarities, this approach facilitates knowledge sharing between devices, thereby boosting detection accuracy.

This paper makes the following key contributions:

(1) Proposed PyraFAN Module: We introduce PyraFAN, a novel feature fusion module that integrates multi-level feature pyramids with attention mechanisms. This lightweight module is specifically designed to address the challenges of small object detection by enhancing feature representation and improving detection accuracy.
(2) Cloud-Edge Collaborative Framework: We propose an innovative cloud-edge collaborative inference framework that optimizes the balance between computational efficiency and detection accuracy. This framework leverages the strengths of both

cloud and edge computing to improve the robustness and real-time performance of small object detection systems.

(3) Graph-Guided Distillation Method: We develop a graph-guided knowledge distillation method for training edge models. This method enhances the detection capabilities of edge devices by effectively transferring knowledge from cloud models while considering spatial, task, and model similarities among edge nodes.

The following paper is organized as follows. Section 2 describes the related research. Section 3 describes the design of the proposed module (PyraFAN). Section 4 describes the proposed cloud-edge collaboration method. Section 5 provides the experimental evaluation and results. Section 6 concludes and outlines future work.

2. Related Work

Cloud-edge collaboration is a distributed computing model that integrates cloud computing, edge computing, and terminal devices. It aims to optimize resource allocation and data processing workflows to deliver higher efficiency and lower-latency service experiences. In deep learning research on cloud-edge collaboration, Surat Teerapittayanon et al. proposed partitioning neural networks into segments deployed across cloud, edge, and device layers for collaborative inference [19]. By incorporating early exit points, their approach reduces reliance on cloud models and decreases communication costs. Yazhou Yuan et al. applied a cloud-edge-end framework to drone object detection [20]. By introducing decision mechanisms that dynamically trigger cloud collaboration, they balanced real-time detection needs with marginal accuracy tradeoffs. Chuntao Ding et al. designed a dynamic cognitive service framework with hierarchical parameter sharing to address edge-specific challenges like limited data and model overfitting [21]. Their method trains complex cloud models (CloudCNN) and transfers low-level feature parameters to edge models (EdgeCNN), mitigating performance degradation caused by data scarcity at the edge.

However, current collaboration scenarios predominantly focus on vertical synergy (leveraging cloud computational power and edge real-time processing). While effective for general applications, this strategy proves oversimplified for small object detection. It underutilizes the geographical advantages of edge devices and local scene-specific data features. Cloud models, despite larger parameter capacities, train on heterogeneous datasets. Only reliance on cloud-based high-precision detection increases communication overhead and may underperform edge-optimized models due to feature generalization issues.

YOLO, introduced in 2015, revolutionized object detection by framing it as a single-stage regression problem. Its unified neural network for localization and classification significantly accelerated detection speeds, enabling real-time applications. Contemporary deep learning object detectors fall into two categories, single-stage detectors dominate small object detection with their efficiency. For example, YOLOv11 enhances small object accuracy via a multi-dimensional cooperative attention mechanism and multi-scale feature fusion. Its lighter architecture suits edge deployment. Transformer [22]-based detectors (e.g., DETR) improve small object performance through global feature modeling, boosting recall by 14.3% on COCO datasets. AT-YOLO [23] optimizes DenseNet connections (SCDB module) for computational efficiency, enabling a "lightweight edge inference + cloud deep analysis" paradigm in video surveillance.

Current optimization strategies face limitations. Model-centric enhancements (e.g., adding attention modules) increase parameters and inference latency while neglecting data diversity in cloud-edge scenarios. Edge feature generalization remains unresolved. locally trained edge models lose specificity when integrated into cloud-global frameworks, reducing detection robustness. Specialized training strategies (e.g., knowledge distillation,

federated feature refinement) are critically needed to leverage distributed data while preserving privacy and adapting to edge-specific conditions.

3. Design of Pyramid-Fusion Attention Network for Small Object Detection (PyraFAN)

To enhance the model's accuracy in small object detection, we propose the PyraFAN module. The structure is illustrated in Figure 1. The detailed structure of PyraFAN improves the accuracy of small object detection by fusing multi-level features and setting up a dedicated detection head for small objects.

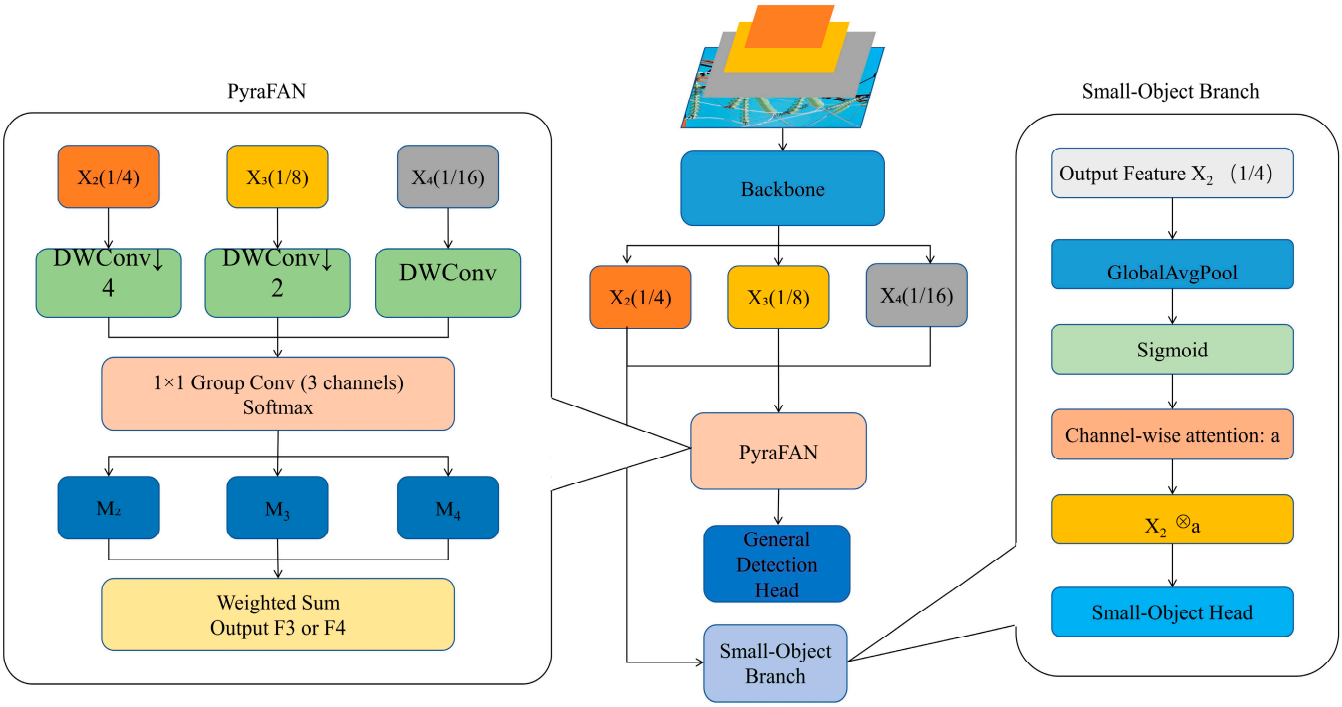


Figure 1. Design of Pyramid Fusion Attention Network for Small Object Detection.

This module (PyraFAN) is inspired by Adaptive Spatial Feature Fusion (ASFF) and redesigns a feature fusion strategy that retains its small object gain while adding virtually zero extra computation. This module keeps a three-scale dynamic fusion and direct P2 bypass with lightweight channel attention is used. The details are as follows:

This module keeps a three-level feature pyramid:

$$\mathcal{X} = \{X_2, X_3, X_4\},\ X_l \in \mathbb{R}^{C_l \times H_l \times W_l},\ H_l = \frac{H_0}{2^l} \tag{1}$$

where C_l, H_l, W_l represent the number of channels, height, and width of the feature map X_l at level l of the feature pyramid. The input size $H_0 = 640$. For object levels $l \in \{3, 4\}$, the fused feature is

$$F_l = \sum_{k=2}^{4} \boldsymbol{M}_{l,k} \odot \mathcal{T}_{l,k}(X_k) \tag{2}$$

where $\odot$ denotes element-wise multiplication. It combines the weight matrix $\boldsymbol{M}_{l,k}$ with the transformed feature map $\mathcal{T}_{l,k}(X_k)$ to fuse features from different levels adaptively. The lightweight alignment $\mathcal{T}_{l,k}(X_k)$ is

$$\mathcal{T}_{l,k}(X_k) = DWConv_{3\times3}^{C/r}(Interp_{\times 2^{l-k}}(X_k)) \tag{3}$$

where *DWConv* stands for Depth-Wise Convolution. It is a lightweight convolution operation that applies a single filter to each input channel, reducing computational complexity. Additionally, r = 4 and *Interp* denotes nearest-neighbor up/down-sampling.

$$\begin{cases} \boldsymbol{Z}_k = \mathcal{T}_{l,k}(X_k) \in \mathbb{R}^{C/r \times H_l \times W_l}, \\ \boldsymbol{W} = Softmax\Big(Conv^3_{1\times1}(\boldsymbol{Z}_2||\boldsymbol{Z}_3||\boldsymbol{Z}_4), dim = 1\Big), \\ \boldsymbol{M}_{l,k} = \boldsymbol{W}_k,\ k = 2,\ 3,\ 4. \end{cases} \tag{4}$$

where the second formula computes the weights for feature fusion. It applies a 1×1 convolution to the concatenated feature maps $\boldsymbol{Z}_2, \boldsymbol{Z}_3, \boldsymbol{Z}_4$, followed by a $Softmax$ function along the channel dimension (dim = 1) to produce normalized weights $\boldsymbol{W}$, ensuring that each feature map contributes meaningfully to the fusion process. After concatenation the tensor has 3C/r channels (e.g., 192). A 1×1 *Conv* with 3 outputs is followed by $Softmax$ to yield per-pixel, three-level weights.

High-resolution feature X_2 bypasses fusion, is fed directly to the detection head and refined by an efficient channel-attention gate. It uses Global Average Pooling (GAP) to aggregate spatial information of the feature map X_2 into channel-wise statistics, which are then scaled between 0 and 1 using a sigmoid function (σ). The result, $\boldsymbol{a}$, is used to refine X_2 through element-wise multiplication ($\otimes$), enhancing important channels and suppressing less relevant ones.

$$\begin{cases} \boldsymbol{a} = \sigma(GAP(X_2)) \in \mathbb{R}^{C_2}, \\ X_2^{attn} = X_2 \otimes \boldsymbol{a} \end{cases} \tag{5}$$

where σ is *Sigmoid* and $\otimes$ denotes broadcast multiplication. Small object branch uses only X_2^{attn} and general branch uses fused features $\{F_3, F_4\}$. Both branches share a 3×3 depth-wise *Conv*; only the final 1×1 convolutions are task-specific:

$$\begin{cases} \boldsymbol{B}_l = DFL(Conv^{4K}_{1\times1}(DWConv_{3\times3}(F_l))), \\ \boldsymbol{C}_l = \sigma(Conv^{|\mathcal{C}|}_{1\times1}(DWConv_{3\times3}(F_l))) \end{cases} \tag{6}$$

where K = 16. $|\mathcal{C}|$ represents the number of object types. This formula defines the detection heads for bounding box regression and classification. The first part, $\boldsymbol{B}_l$, computes the bounding box coordinates using a Distributed Focal Loss (DFL) applied to the output of a 1×1 convolution, which follows a 3×3 depth-wise convolution on the fused feature F_l. The second part, $\boldsymbol{C}_l$, produces classification probabilities for each object category using a sigmoid function applied to the output of another 1×1 convolution, also following a 3×3 depth-wise convolution on F_l.

Through the feature fusion of PyraFAN module, the model can not only effectively recognize normal objects, but also combine multi-level features in the inference of small objects, and use dedicated small object detection heads to improve detection accuracy.

4. Cloud-Edge Collaborative Inference Based Smart Detection Method

Edge models, compared to cloud models, offer the advantage of retaining greater weights for local data features and lower inference latency. However, due to the limited computing power of edge devices, such as commonly used AR devices that struggle to deploy complex network models, so computation must be augmented with cloud resources. To overcome the limitations of traditional fixed cloud-edge division of labor, we propose a cloud-edge collaborative inference based smart detection method. The structure is illustrated in Figure 2.

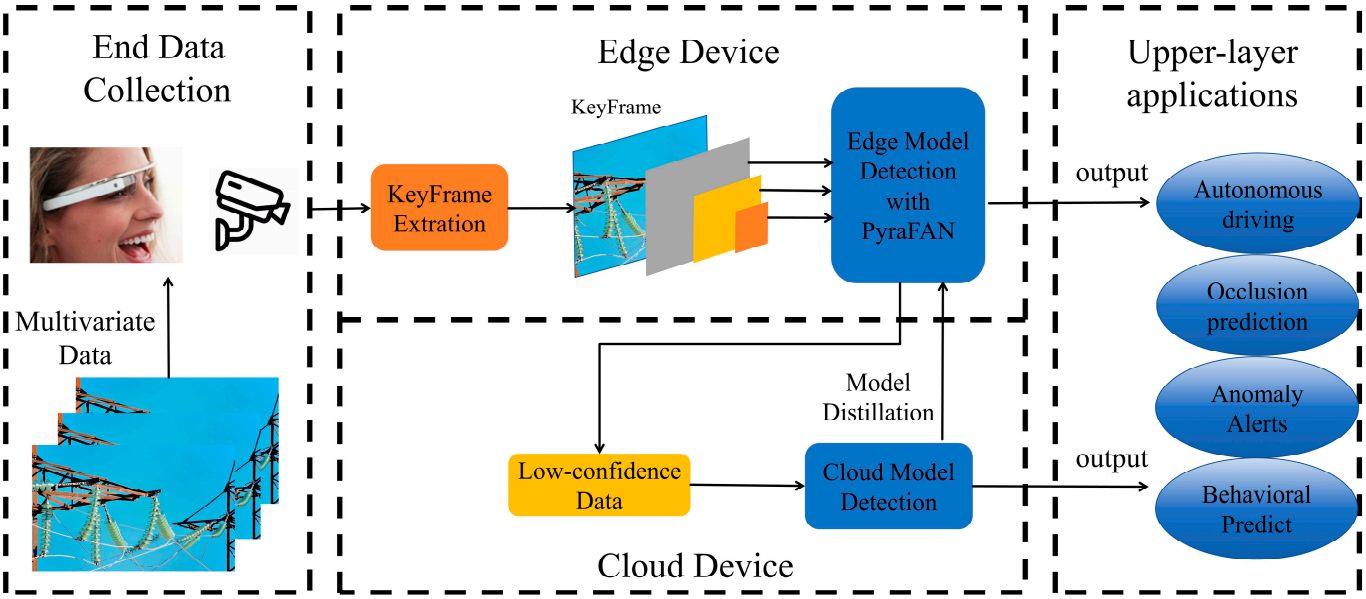


Figure 2. Cloud-Edge Collaborative Inference Method. This method consists of three layers of components: end devices (users), edge devices and cloud devices.

4.1. Design of the Proposed Method

End devices (e.g., cameras) are solely responsible for data acquisition, capturing photo sequences or videos and uploading raw data to edge nodes, such as AR glasses, Microsoft Hololens, Smartphone, and Apple Watch. These devices act as data sources with minimal local processing, optimized for low-power operation in distributed IoT systems.

Edge devices are usually a group of servers that are deployed at the edge of the network, such as sensors, home gateways, and micro servers. These edge devices will utilize these key frames for object detection and then transmit the detection results to the upper-layer applications. Based on this detection, data with confidence scores below the set threshold will be uploaded to the cloud model for re detection, reducing the missed and false detection rates.

Cloud devices are usually AliCloud servers, Amazon Web Service Cloud servers, Microsoft Azure Cloud servers, and Google Cloud servers. Considering that the cloud server has rich computing and storage resources, we make it responsible for computationally intensive tasks, such as training complex models.

The cloud platform utilizes the cloud model for high-precision re-detection. Concurrently, the cloud can construct datasets using the data uploaded by edge nodes and leverage cloud models to perform knowledge distillation on the edge models. This process further enhances the detection accuracy of the edge models.

4.2. The Details of the Cloud-Edge Collaborative Inference Method

This method aims to enhance real-time performance and accuracy for small object detection by optimally scheduling heterogeneous computing resources (cloud, edge, and device endpoints) and enabling adaptive task allocation. The specific workflow and innovations of this architecture are as follows:

1. Dynamic coordination and task diversion for small objects: Instead of static task division, this architecture establishes a dynamic coordination mechanism. End device (like AR glasses and cameras), as sensing tentacles, focus on collecting raw video streams, denoted as $V = \{f_t\}_{t=1}^{T}$. Edge nodes play a key role in real time response, performing temporal correlation analysis on continuous frame data. They dynamically

extract key-frames through inter-frame difference method, effectively filtering out redundant information.

$$\Delta f_t = |f_t - f_{t-1}|_2 \tag{7}$$

where Δf_t is the inter-frame difference calculated by the inter-frame difference method. When the inter-frame difference exceeds a threshold β, the frame is marked as a key-frame, usually with a value of 1.5 or 2.

$$\Delta f_t > \beta \cdot median\left(\{\Delta f_i\}_{i=1}^{t}\right) \tag{8}$$

A temporal data $W_j = [k_1, k_2 \ldots, k_m]$ is constructed where W_j is the sequence of temporal data, and each k_i is a key-frame. To leverage frame to frame correlation information, the temporal data within this window can be used to build trajectories for objects.

Edge models M_e perform preliminary inference on W_j, and the results can directly serve low latency applications, such as real time alert annotation in AR glasses. For ambiguous samples with insufficient edge model confidence, a task offloading mechanism to the cloud is triggered. In this approach, ambiguous samples are determined based on a set of clear and quantifiable multi-dimensional criteria, rather than vague judgments. Specifically, when a detected object meets one or more of the following preset conditions, it is deemed an ambiguous sample:

(a) Low-confidence samples: The confidence score of the detection result output by the edge model is below a preset confidence threshold T_{conf}. This threshold can be adjusted according to the signal to noise ratio of the actual scenario, for example, set at 0.5.

(b) Physically small size samples: The pixel area of the detected object's bounding box is smaller than a preset size threshold T_{size}. This directly corresponds to objects with weak features and difficult to classify on the edge side due to their small physical size.

(c) Unstable detection samples: Combined with temporal analysis, if an object roughly in the same spatial position is intermittently detected and missed in continuous N frames (N is a preset integer, e.g., N = 5), it is also regarded as an ambiguous sample. This indicates unstable detection results, requiring deeper consistency verification by the cloud.

With this refined determination mechanism, this approach accurately filters out objects truly needing the cloud's powerful computing power for refined analysis (especially small objects with tiny size and blurry features). It enables intelligent and efficient task diversion, avoiding unnecessary bandwidth waste and cloud resource occupation.

2. Cloud based in depth analysis and closed loop knowledge gain for small objects: The cloud, with its powerful computing power, conducts in-depth, refined secondary analysis of received data. The cloud model M_c performs high precision reinspection of data:

$$D_c = M_c(W_j) \tag{9}$$

This model can be a model optimized for small object detection, with a larger parameter size and higher detection accuracy.

Most crucially, this method forms a closed loop for small objects knowledge transfer. The cloud model, as a "teacher," continuously empowers and optimizes the "student" models deployed on AR glasses and other devices through response-based knowledge distillation. This enables online incremental learning and continuous evolution of edge intelligence.

4.3. Graph-Guided Distillation Among Edge Models

Aiming at the "data island" and "feature generalization" problems caused by data sparsity and scene limitation of single edge nodes (such as single AR glasses), we propose a graph-guided distillation among edge models method. The structure of the method is shown in Figure 3.

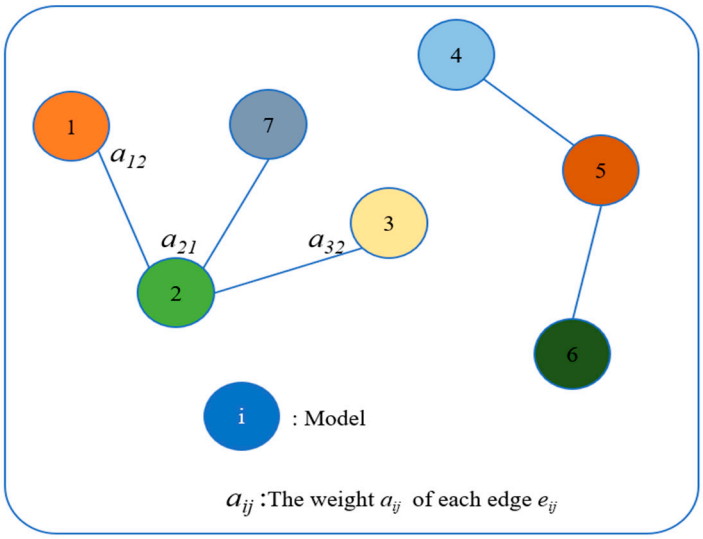


Figure 3. Graph-Guided Distillation among Edge Models.

Each edge model can be used as each node in the graph, and then weights are calculated based on spatial position, task similarity, and model similarity. The degree of each node is calculated and sorted, and higher models are distilled towards lower models. Its core is to enhance the detection ability of the entire edge network for small objects through knowledge sharing among models without compromising privacy data. The specific implementation of this mechanism is as follows:

To organize collaborative training among models in a scientific way, this paper proposes a performance homology quantification method for key small objects. After being trained on their respective private datasets, all edge models are evaluated on a unified public benchmark dataset. Their performance characterization vectors are constructed based on the average precision (AP) of various object detection and the mean average precision (mAP) output by the tests:

$$V_{model} = [mAP, AP_1, AP_2, \ldots, AP_n]^T \tag{10}$$

where AP_i is the detection accuracy of each object category and mAP is the mean of detection accuracy across all categories. Then, each AP_i is normalized as follows:

$$AP_i\prime = \frac{AP_i - \mu_i}{\sigma_i} \tag{11}$$

where μ_i is the mean of AP for category i across all models, and σ_i is the standard deviation of AP for category i across all models.

One of the key innovations of this paper lies in the calculation of the weighted Euclidean distance between models. Higher weights w_k can be assigned to the detection accuracy AP of specific small object categories (such as key bolts and tiny cracks) according to operational and maintenance needs. The weighted Euclidean distance is calculated as follows:

$$D_{weighted}(i,j) = \sqrt{\sum_{k=1}^{n+1} w_k \cdot \left(V_i[k] - V_j[k]\right)^2} \tag{12}$$

where w_k is determined based on whether weighting is needed. If a mean value is used, then $w_k = 1$. If weighting for particularly small objects is required, adjustments are necessary $w_k = \frac{N_k}{\sum N_k}$. This weight design ensures that the collaborative evolution process no longer blindly pursues overall *mAP* improvement but instead focuses on and prioritizes the recognition capability for these high risk, hard to detect key small objects, achieving targeted enhancement of detection performance.

Every edge device is regarded as a node V_i in an undirected weighted graph. The weight a_{ij} of each edge e_{ij} is computed as the normalized product of three complementary factors that jointly characterize the collaboration potential between two nodes:

(1) Spatial proximity:

$$a_{ij}^{(s)} = exp\left(-\frac{\|p_i - p_j\|_2^2}{2\sigma_s^2}\right) \tag{13}$$

where p_i denotes the 2-D geolocation of node i.

(2) Task relevance:

$$a_{ij}^{(t)} = \frac{|\mathcal{C}_i \cap \mathcal{C}_j|}{max(|\mathcal{C}_i|, |\mathcal{C}_j|)} \tag{14}$$

where the task relevance $a_{ij}^{(t)}$ is computed as the ratio of the intersection to the maximum size of the task categories between models i and j.

(3) Model-feature similarity (AP-based performance vector):

$$a_{ij}^{(f)} = \frac{1}{1 + D_{weighted}(i,j)} \tag{15}$$

The final edge weight is normalized across the one-hop neighborhood

$$a_{ij} = \frac{a_{ij}^{(s)} \cdot a_{ij}^{(t)} \cdot a_{ij}^{(f)}}{\sum_{k \in \mathcal{N}_i} a_{ik}^{(s)} \cdot a_{ik}^{(t)} \cdot a_{ik}^{(f)}} \tag{16}$$

Then define the node degree as $d_i = \sum_j a_{ij}$, which reflects the "teaching qualification" of node i. After each training round, the top $-$ ρ percentile of nodes ranked by d_i are designated as the teacher set $\mathcal{T}$.

For every student node i, if its neighbor j is in $\mathcal{T}$, soft-label distillation is performed:

$$L_{total} = \alpha L_{CE} + (1 - \alpha) L_{KD} \tag{17}$$

where L_{CE} measures the difference between the student model's predictions and true labels. L_{KD} is the distillation loss, calculated via KL divergence between the two models' output. It reflects the difference between the student and teacher models' predictions. α is a weighting parameter that balances these two losses. It's usually small to make the teacher model's outputs more influential in distillation.

Distillation loss uses KL divergence (Kullback-Leibler divergence) to measure the difference between the student and teacher models' outputs:

$$L_{KD} = \sum_i KL(p_s(y_i), p_t(y_i)) \tag{18}$$

$$\mathrm{KL}(p \parallel q) = \sum_i p(i) log \frac{p(i)}{q(i)} \tag{19}$$

And a temperature coefficient T is introduced to smooth the probability distribution. This helps the model learn deeper category correlations, not just hard labels:

$$p_s(y_i) = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)} \tag{20}$$

where z_i is the model's unnormalized output score and T is the temperature parameter. When $T > 1$, $softmax$ outputs become smoother, reducing probability difference between categories. Through collaborative evolution, each edge model's generalization ability and detection accuracy for small objects are significantly improved.

To ensure the effectiveness of the collaborative evolution mechanism and efficient utilization of system resources, this paper also designs a flexible clustering triggering and dynamic updating management strategy. This strategy aims to balance model performance improvement and system overhead, ensuring to recalculate the graph when it is most needed. As shown in Figure 4, updating is primarily based on the following two modes:

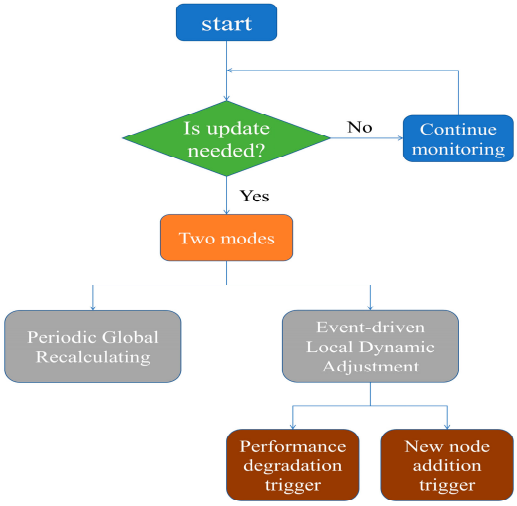


Figure 4. Graph Structure Update Flowchart.

(a) Periodic Global Recalculating: System administrators can set a fixed recalculating period T_{cycle} (e.g., weekly, or monthly) according to operational and maintenance needs. At the end of each period, the system automatically triggers all edge models to undergo performance evaluation on the unified benchmark dataset and re-executes the complete clustering process based on the latest performance characterization vectors. This mode ensures that the entire edge network periodically adjusts collaborative relationships to adapt to slow environmental or device state changes.

(b) Event-driven Local Dynamic Adjustment: When the system detects a significant change in the performance of a single edge node's model, local adjustment is triggered. There are two situations that require adjustment:

Performance degradation trigger: If the mAP of an edge model M_k on its local validation set is below a specific threshold of the historical average for several consecutive monitoring points δ_{down}, the system considers that the model may have encountered new scene challenges or experienced data drift. In this case, the correlation relationship of the model M_k will be recalculated, and the graph structure and correlation values will be update.

New node addition trigger: When a new edge device (e.g., a new AR glass) is deployed and joins the network, the system initializes a model for it. After the new model completes

initial local training, the system immediately triggers an evaluation. This helps the new model quickly integrate into the collaborative update system and absorb existing knowledge.

This management strategy, combining "periodic global optimization" and "event-driven local adjustment," transforms model collaborative evolution from a one-time static configuration into a dynamic process that can self-adapt, incur low overhead, and operate continuously. It ensures long-term stability and efficient improvement of the system's small object detection capabilities.

5. Results and Discussion

The experiments mainly focus on two aspects: detection accuracy and latency. This section presents the configuration of the experimental simulation and some parameter settings. In terms of detection accuracy, the performance of the cloud-edge collaborative model and the original baseline model is compared on the VisDrone2019-DET [24] and VisDrone2019-VID [25] datasets. In terms of latency, the experiment tests the detection latency of the cloud model, the edge model, and other algorithms, as well as the latency of uploading data to the cloud and the edge respectively. The detailed experimental setup is described in Section 5.1. All parameters mentioned above are set as default, except for those specifically mentioned.

5.1. Experimental Design

Datasets: The VisDrone2019 dataset is collected by the AISKYEYE team at Lab of Machine Learning and Data Mining, Tianjin University, China. The benchmark dataset consists of 288 video clips formed by 261,908 frames and 10,209 static images, captured by various drone-mounted cameras, covering a wide range of aspects including location (taken from 14 different cities separated by thousands of kilometers in China), environment (urban and country), objects (pedestrian, vehicles, bicycles, etc.), and density (sparse and crowded scenes). Note that, the dataset was collected using various drone platforms (i.e., drones with different models), in different scenarios, and under various weather and lighting conditions. These frames are manually annotated with more than 2.6 million bounding boxes of objects of frequent interests, such as pedestrians, cars, bicycles, and tricycles.

Experimental Setup: The edge model utilizes YOLOv11n, while the cloud model employs YOLOv11s. The performance of the PyraFAN and collaboration framework, as well as the graph-guided distillation method among cross—domain edge models, are evaluated on the VisDrone2019-DET and VisDrone2019-VID datasets. We trained all models on the NVIDIA GeForce RTX 4090 for 100 epochs, with an input size of 640 $\times$ 640. In the collaborative inference experiments, we tested the collaborative inference accuracy under different confidence thresholds τ. Moreover, the task of small object detection is challenging, so in the distillation process, the temperature is set to 1.

5.2. Detection Performance

In the cloud-edge scenario, the cloud model trains on data from various edges. After training, the weights assigned to each edge's specific scenes decrease. Thus, cloud model detection on certain samples may underperform compared to edge models. As the ablation study shown in Table 1, under the cloud—edge collaboration framework, the collaborative strategy's detection accuracy surpasses both the edge model and the cloud model, with a 6.98% boost over the cloud model on VisDrone-DET. The edge model using the PyraFAN module proposed in this article can improve detection accuracy by about 12.2% (compared with the edge model). This is because PyraFAN incorporates a dedicated small object detection head when processing small objects and achieves better recognition of small objects through feature fusion, thereby improving detection accuracy. This cloud-edge

collaboration framework creates a complementary effect between the cloud and edge models. The edge model YOLOv11n's "high-confidence" results are highly accurate as they represent its most confident predictions, while the cloud model YOLOv11s handles the remaining "challenges".

Table 1. Ablation Study. This table shows the detection accuracy of different strategies on two datasets.

Model	VisDrone-DET (mAP@0.5)	VisDrone-VID (mAP@0.5)
Cloud Model	0.401	0.302
Edge Model	0.328	0.246
Edge Model with PyraFAN	0.368	0.262
Cloud-to-Edge Distillation	0.336	0.249
Edge-to-Edge Distillation	0.336	0.248
Cloud-to-Edge Distillation + Edge-to-Edge Distillation	0.337	0.247
Cloud-Edge Collaboration	0.429	0.302

As shown in Table 2, we conducted a comparison with other models or collaborative algorithms. When all models involved belong to the YOLOv11 series, our collaborative strategy achieves the optimal performance. By comparing the GFLOPs (Giga Floating-point Operations Per Second) and FPS (Frames Per Second) metrics of edge models, our method has yielded good results, enabling lightweight deployment without significantly affecting the detection speed. PyraFAN, as a module, can be added to the object detection series of models without significantly increasing the model's parameter quantity. And, a dedicated small object detection branch has been set up, thereby improving the detection accuracy in small object detection tasks.

Table 2. Performance of different algorithms on the VisDrone-DET dataset. This table displays the detection accuracy and speed of different strategies.

Method	mAP@0.5	GFLOPs	FPS
E^2L [15]	0.375	28.8	110
YOLOv10s (pre-trained)	0.403	21.6	133
YOLOv11n	0.298	7.7	145
AT-YOLO [18]	0.299	4.41	127
Collaboration with PyraFAN (ours)	0.429	15.7	135

As shown in the Tables 3 and 4, under the cloud-edge collaboration framework, not only does the detection accuracy improve, but the model's missed detection rate also decreases. Here, the confidence score indicates that when the confidence threshold is set close to this value, the precision can reach 1.0. FNR (False Negative Rate) can be understood as the proportion of actual positive cases that are incorrectly predicted as negative. On the VisDrone2019-DET dataset, compared to the edge model, this strategy reduces the false detection rate by 19.56%. This indicates that more objects can be detected under cloud-edge collaboration. After distilling knowledge from the cloud model to the edge model, the edge model's detection accuracy on VisDrone-DET improves by approximately 2.7%. This is because when the cloud receives data transmitted from the edge, it can construct a dataset dedicated to that edge model, and the data in this dataset are all targets that are relatively difficult to detect compared to the edge model. Its detection results adjust the edge model's parameters, elevating the edge model's accuracy. However, on VisDrone-VID, results are significantly lower than on VisDrone-DET. This is likely due to VisDrone-VID's larger but less diverse dataset of continuous video frames, offering fewer learning increments.

Table 3. Different Model's Performance on VisDrone-DET.

Model	Confidence	Recall	FNR
Cloud Model	0.952	0.60	0.40
Edge Model	0.972	0.54	0.46
Collaborative Edge Model	0.960	0.54	0.46
Edge Model with PyraFAN	0.967	0.55	0.45
Cloud-Edge Collaboration	0.952	0.63	0.37

Table 4. Different Model's Performance on VisDrone-VID.

Model	Confidence	Recall	FNR
Cloud Model	0.980	0.57	0.43
Edge Model	0.999	0.55	0.45
Collaborative Edge Model	0.974	0.51	0.49
Edge Model with PyraFAN	0.982	0.56	0.44
Cloud-Edge Collaboration	0.98	0.57	0.43

Experiments also contrasted the graph-guided distillation among edge models. As indicated in Table 1, The model distillation from low-degree nodes is performed using the models of high-degree nodes. This enhances detection accuracy of edge model, matching the performance of cloud model assisted distillation. This is because edge models, post training, assign greater weight to specific scene data than cloud models. The cloud model's generalization and the relevance of information exchanged among edge models to edge scene data result in performance on par with cloud model assisted distillation.

After distilling between edge models, using cloud models to distill edge models results in minimal improvement in detection accuracy on both datasets. This is attributed to the limited knowledge increment after the initial cloud model distillation, restricting further gains. Additionally, the effectiveness of knowledge distillation in enhancing detection accuracy is closely tied to the data samples themselves. The dataset's characteristics and limitations are also one of the key factors restricting the further improvement of detection accuracy.

When using the cloud-edge collaboration method based on confidence distinction, setting different confidence thresholds will have varying impacts on the final average detection accuracy. In the experiment, the cloud model adopts the YOLOv11s model, while the edge model uses three different models. The average detection accuracy of the collaborative method under different confidence thresholds is tested respectively. As shown in the Figure 5, this figure shows the average detection accuracy of cloud-edge collaboration under different confidence thresholds when the cloud model remains unchanged and different edge models are used respectively. The average detection accuracy decreases as the threshold increases. This is because with the increase of the confidence threshold, more tasks are completed directly on the edge server. Since the performance of the cloud model is better, in the final statistics, the number of objects detected by the edge model is larger, leading to a slight decrease in the average detection accuracy, but the decline is not significant. Both PyraFAN and the cloud-edge collaborative framework enhance detection accuracy. The standalone edge model's detection accuracy (mAP) is merely 0.328 during inference. However, with collaboration, the accuracy can rise above 0.38. Furthermore, PyraFAN boosts small object recognition, leading to an additional improvement in detection accuracy.

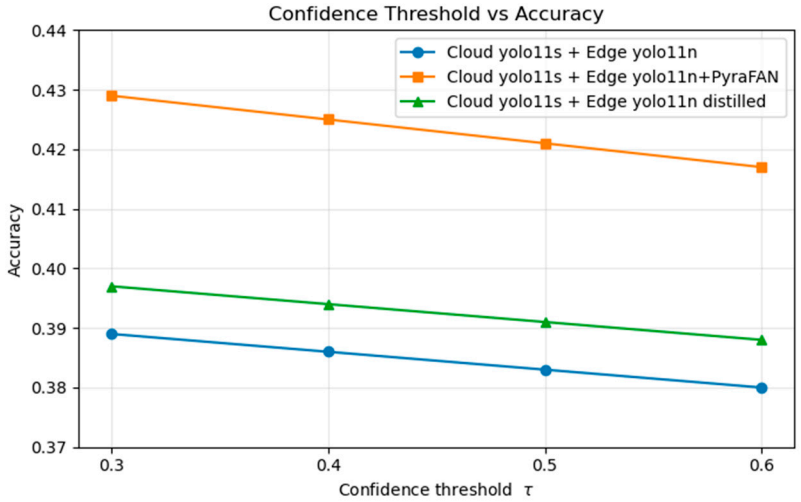


Figure 5. Detection Accuracy of Different Edge Models in Cloud-Edge Collaboration.

We also compared the distillation effects of the edge model with a noisy model added. This noisy model wasn't trained on the VisDrone dataset, so it has low similarity in data scenes, tasks, and other model features. As shown in Figure 6, distilling between models with high similarity improved the detection accuracy, reaching 0.336 (mAP@0.5), compared to the baseline model. However, after incorporating the noisy model, the detection accuracy not only failed to increase but even dropped below the baseline model's, reaching only 0.32. This is because response-based distillation strategies are sensitive to data. Distilling from a teacher model doesn't always boost the student model's detection accuracy, and adding a noisy model can have a similar effect, potentially reducing performance.

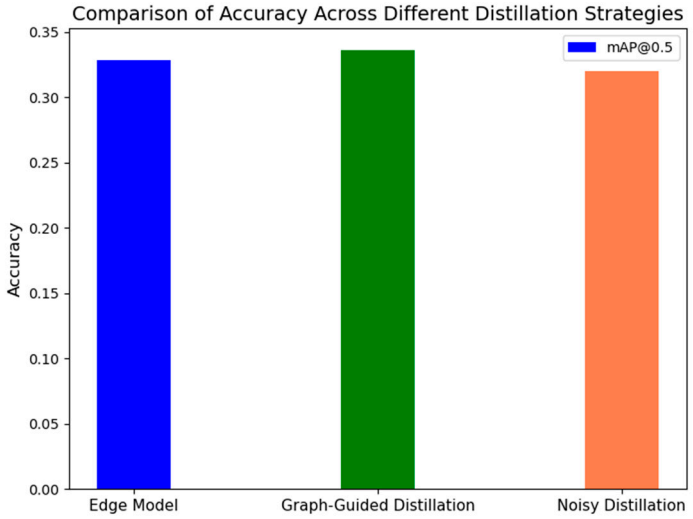


Figure 6. Comparison of Accuracy Across Different Distillation Strategies.

5.3. Response Time

We tested the inference latency of the model on the test set, calculated the average inference time required for one image, and tested the upload latency and model inference latency under different image sizes. As shown in Figure 7, this figure shows the average response time of different models or algorithms on the test set. There are 548 test images. The inference time using the cloud model is the longest, while the inference time using only the edge model is the shortest. This is because the edge model's inference latency was lower than the cloud model's due to its smaller parameter size and lower computational demand. Both YOLOv11n and YOLOv11s are lightweight models with low parameter

quantities, leading to minimal experimental data differences. Meanwhile, as the image size increases, the inference time of the model also increases, and the time required to upload images from the end to the cloud and the edge, respectively, also gradually increases. Figure 8 indicates that uploading images to a cloud server caused significantly higher latency than to an edge server. Cloud servers, typically located in remote data centers, require data to traverse multiple network nodes, accumulating signal and data processing delays. In contrast, edge servers, positioned at the network's edge and closer to users, reduce data transmission distance and the number of network nodes, effectively cutting latency. In cloud-edge-end scenarios, leveraging the edge model's geographical advantage can enhance detection real-time.

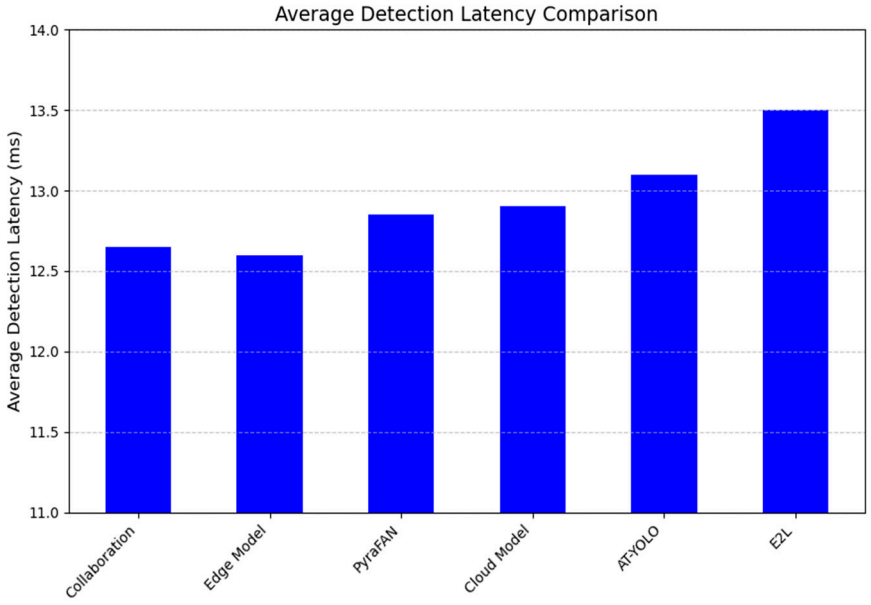


Figure 7. The Average Detection Latency of Different Models.

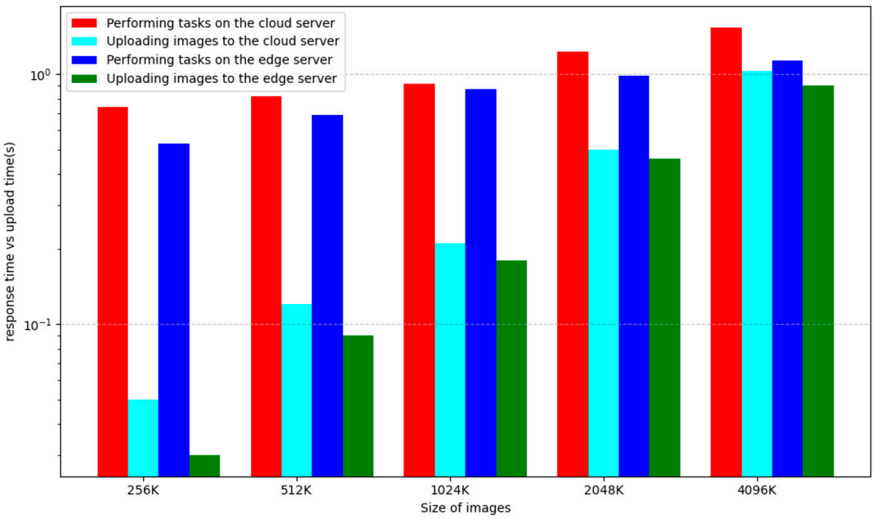


Figure 8. The Response Time vs. Upload Time.

6. Conclusions

In real world applications, small object detection faces challenges like low detection confidence due to the objects' tiny size, low resolution, and susceptibility to occlusion. These tasks also demand high-speed processing to promptly feed results back to applications. The cloud-edge collaboration framework with PyraFAN module and graph-guided distillation

method among edge models proposed in this paper have effectively enhanced detection accuracy and cut false negative rates. Moreover, the cloud-edge collaboration framework has reduced the average inference latency.

For future work, we plan to optimize the graph-guided distillation method, potentially via mutual learning to boost training efficiency. We will also explore model partitioning to further reduce inference latency in the dynamic cloud-edge collaborative reasoning framework. Additionally, we aim to integrate our methods with related techniques like data augmentation and weakly supervised learning. This integration is expected to elevate the performance and robustness of small object detection. We will also investigate applying our approach to more practical fields such as intelligent transportation, security surveillance, and autonomous driving, offering more practical value to edge intelligence.

Author Contributions: C.Y. and S.L. drafted the manuscript and conducted the experiments. J.W. and H.L. designed the experiments and provided guidance. X.L. and S.S. contributed to the critical revision and final approval of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: The authors declare that this study received funding from the State Grid Ningxia Electric Power Co., Ltd. Science and Technology Project under grant 5229XT240002. The funder had the following involvement with the study: Cloud-Edge Collaborative Inference-Based Smart Detection Method for Small Objects. The authors declare that this study received funding from the State Grid Ningxia Electric Power Co., Ltd. Science and Technology Project. The funder was not involved in the study design, collection, analysis, interpretation of data, the writing of this article or the decision to submit it for publication.

Data Availability Statement: The authors declare that the data supporting the findings of this study are available in the article and can be found in Section 5.

Conflicts of Interest: Authors C.Y., J.W., H.L., and X.L. were employed by the company Information & Communication Company of State Grid Ningxia Electric Power Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
2. Zhang, Y.; Xiang, T.; Hospedales, T.M.; Lu, H. Deep Mutual Learning. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4320–4328. [CrossRef]
3. Yang, Z.; Li, Z.; Shao, M.; Shi, D.; Yuan, Z.; Yuan, C. Masked Generative Distillation. *arXiv* **2022**, arXiv:2205.01529. [CrossRef]
4. Ren, J.; Zhang, M.; Yu, C.; Liu, Z. Balanced MSE for Imbalanced Visual Regression. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 7916–7925. [CrossRef]
5. Ji, S.; Zhang, Z.; Ying, S.; Wang, L.; Zhao, X.; Gao, Y. Kullback–Leibler Divergence Metric Learning. *IEEE Trans. Cybern.* **2022**, *52*, 2047–2058. [CrossRef] [PubMed]
6. Wojke, N.; Bewley, A.; Paulus, D. Simple online and realtime tracking with a deep association metric. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 3645–3649. [CrossRef]
7. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
8. Bakke Vennerød, C.; Kjærran, A.; Stray Bugge, E. Long Short-term Memory RNN. *arXiv* **2021**, arXiv:2105.06756. [CrossRef]
9. Ding, C.; Zhou, A.; Liu, Y.; Chang, R.N.; Hsu, C.-H.; Wang, S. A Cloud-Edge Collaboration Framework for Cognitive Service. *IEEE Trans. Cloud Comput.* **2022**, *10*, 1489–1499. [CrossRef]
10. Yang, L.; Han, Y.; Chen, X.; Song, S.; Dai, J.; Huang, G. Resolution Adaptive Networks for Efficient Inference. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 2366–2375. [CrossRef]

11. Lan, G.; Liu, Z.; Zhang, Y.; Scargill, T.; Stojkovic, J.; Joe-Wong, C.; Gorlatova, M. Collabar: Edge-assisted collaborative image recognition for mobile augmented reality. In Proceedings of the 2020 19th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), Sydney, NSW, Australia, 21–24 April 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 301–312. [CrossRef]
12. Deng, D. DBSCAN Clustering Algorithm Based on Density. In Proceedings of the 2020 7th International Forum on Electrical Engineering and Automation (IFEEA), Hefei, China, 25–27 September 2020; pp. 949–953. [CrossRef]
13. Wang, Y.; Yang, C.; Lan, S.; Zhu, L.; Zhang, Y. End-Edge-Cloud Collaborative Computing for Deep Learning: A Comprehensive Survey. *IEEE Commun. Surv. Tutor.* **2024**, *26*, 2647–2683. [CrossRef]
14. Tao, X.; Duan, Y.; Qin, Z.; Huang, D.; Wang, L. Cloud-Edge-End Intelligent Coordination and Computing. In *Wireless Multimedia Computational Communications*; Wireless Networks; Springer: Cham, Switzerland, 2024. [CrossRef]
15. Li, L.; Zhu, L.; Li, W. Cloud–Edge–End Collaborative Federated Learning: Enhancing Model Accuracy and Privacy in Non-IID Environments. *Sensors* **2024**, *24*, 8028. [CrossRef] [PubMed]
16. Zhou, X.; Xu, X.; Liang, W.; Zeng, Z.; Yan, Z. Deep-learning-enhanced multitarget detection for end–edge–cloud surveillance in smart IoT. *IEEE Internet Things J.* **2021**, *8*, 12588–12596. [CrossRef]
17. Zhang, R.; Jiang, H.; Wang, W.; Liu, J. Optimization Methods, Challenges, and Opportunities for Edge Inference: A Comprehensive Survey. *Electronics* **2025**, *14*, 1345. [CrossRef]
18. Yang, L.; Shen, X.; Zhong, C.; Liao, Y. On-demand inference acceleration for directed acyclic graph neural networks over edge-cloud collaboration. *J. Parallel Distrib. Comput.* **2023**, *171*, 79–87. [CrossRef]
19. Teerapittayanon, S.; McDanel, B.; Kung, H.T. Distributed Deep Neural Networks Over the Cloud, the Edge and End Devices. In Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 5–8 June 2017; pp. 328–339. [CrossRef]
20. Yuan, Y.; Gao, S.; Zhang, Z.; Wang, W.; Xu, Z.; Liu, Z. Edge-Cloud Collaborative UAV Object Detection: Edge-Embedded Lightweight Algorithm Design and Task Offloading Using Fuzzy Neural Network. *IEEE Trans. Cloud Comput.* **2024**, *12*, 306–318. [CrossRef]
21. Ding, C.; Ding, F.; Gorbachev, S.; Yue, D.; Zhang, D. A learnable end-edge-cloud cooperative network for driving emotion sensing. *Comput. Electr. Eng.* **2022**, *103*, 108378. [CrossRef]
22. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is All You Need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
23. Liu, Y.; Yu, Z.; Zong, D.; Zhu, L. Attention to Task-Aligned Object Detection for End–Edge–Cloud Video Surveillance. *IEEE Internet Things J.* **2024**, *11*, 13781–13792. [CrossRef]
24. Du, D.; Zhu, P.; Wen, L.; Bian, X.; Lin, H.; Hu, Q.; Peng, T.; Zheng, J.; Wang, X.; Zhang, Y.; et al. VisDrone-DET2019: The Vision Meets Drone Object Detection in Image Challenge Results. In Proceedings of the ICCV Visdrone Workshop, Seoul, Republic of Korea, 27–28 October 2019. [CrossRef]
25. Zhu, P.; Du, D.; Wen, L.; Bian, X.; Ling, H.; Hu, Q.; Peng, T.; Zheng, J.; Wang, X.; Zhang, Y.; et al. VisDrone-VID2019: The Vision Meets Drone Object Detection in Video Challenge Results. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), Seoul, Republic of Korea, 27–28 October 2019; pp. 227–235. [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.