Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 22, No. 1, 2026

A Lightweight Feature Transformation Network with Progressive Training for Rainy Object Detection

Tianming Wang*, Jian Wei

North China Institute of Computing Technology, Beijing 100083, China
*** Corresponding author:** Tianming Wang (Email: 504999591@qq.com)

Abstract: Rainy weather causes occlusions and blur in images, severely degrading the performance of object detectors deployed in scenarios such as autonomous driving. Existing methods mostly adopt pixel-level deraining preprocessing or adversarial domain adaptation. The former introduces additional computational overhead and error accumulation, while the latter suffers from training instability and tends to forget clear-weather knowledge. This paper proposes a lightweight feature transformation module embedded in the detector backbone, which maps "rainy features" to "rain-free features" in the feature space, allowing the subsequent detection network to adapt to rainy conditions without any modification. To achieve both lightweight design and high accuracy, a knowledge distillation framework based on intermediate feature maps is introduced, where a large-capacity teacher network guides a small student network to learn the feature mapping. Meanwhile, a three-step progressive training strategy is designed to sequentially train the detector, distill the transformation module, and fine-tune the overall network, ensuring training stability and avoiding catastrophic forgetting. Extensive experiments show that our method has essentially reached the rain-free baseline level, while significantly reducing computational cost compared to preprocessing approaches. This work provides a new perspective for the efficient and robust deployment of vision systems under adverse weather conditions.

Keywords: Adverse weather object detection; Knowledge distillation; Joint training.

1. Introduction

Object detection is a cornerstone task in computer vision and has been widely deployed in practical systems such as autonomous driving, intelligent surveillance, and industrial quality inspection. In recent years, deep learning-based detectors (e.g., two-stage Faster R-CNN[1], one-stage FCOS[2], and the YOLO series[3-9]) have achieved remarkable performance on standard academic benchmarks such as MS-COCO[10] and PASCAL VOC[11]. However, these detectors generally assume that the training and test data are independent and identically distributed and are collected under clear, high-visibility weather conditions. In the real world, weather conditions are highly variable; adverse weather such as rain, snow, and fog is unavoidable, causing a drastic degradation in detector performance.

The impact of rain on image quality is multi-dimensional. First, rain streaks are spatially non-uniform, locally occluding the edges, textures, and color information of objects and destroying discriminative features. Second, the refraction and scattering of light by raindrops reduce the overall image contrast, making objects less distinguishable from the background. In particular, for small distant objects in autonomous driving, superimposed rain streaks can easily lead to missed or false detections, posing serious risks to driving safety. Therefore, studying how to make existing detectors effectively robust to rain interference is of significant academic value and practical importance.

To address this problem, several technical routes have been explored in the literature. One is domain adaptation, which aligns rainy image features to the clear domain through adversarial training or style transfer. However, such methods suffer from training instability and tend to lose discriminative semantic information during alignment, resulting in performance degradation on the clear domain (catastrophic forgetting [12,13]). The second is preprocessing-based deraining [14], where an independent deraining network first restores a clean version of the image, which is then fed into a fixed detector. This scheme offers high modularity but has two inherent drawbacks: first, pixel-level reconstruction often introduces over-smoothing or artifacts, and these errors are amplified in the subsequent detection; second, the additional network increases inference latency, making it difficult to meet real-time requirements.

Different from the above schemes, this paper tackles the problem from a "feature-level adaptation" perspective and proposes an embedded feature transformation module. The core idea is to no longer pursue generating visually perfect rain-free images, but instead to directly eliminate rain streak responses in the shallow feature maps of the detector backbone, so that the transformed feature distribution approximates that of clear-weather images. The advantages are evident: the transformation process is directly or indirectly supervised by the detection loss, optimizing for task performance rather than pixel fidelity, and the module can be designed to be extremely lightweight.

However, embedding a feature transformation module into a detector poses two major difficulties. The first concerns how to design a module that is both efficient and accurate. Although shallow feature maps have a lower resolution than the original image, they still possess large spatial dimensions and channel numbers, and stacking complex networks directly would severely slow down the overall inference speed. To tackle this, we adopt knowledge distillation for model compression. Specifically, we construct variants of different capacities based on the convolutional sub-network proposed in denseformernet [15]. Using a large-capacity network as the teacher and the distance between intermediate feature maps as the distillation loss, we guide the small-capacity student network to learn the teachers feature

representation capability. Distillation not only compresses the parameter count but also enables the student to inherit the teachers ability to suppress rain streak responses. The second difficulty lies in training the module to collaborate seamlessly with the detector. If the detector and the transformation module are jointly trained end-to-end from scratch, they can easily fall into local optima and converge slowly because their optimization objectives differ—the detector pursues classification and regression accuracy, while the transformation module pursues feature distribution alignment. To address this, we design a three-step progressive training strategy: Step 1, pre-train the detector on clear images to establish stable backbone feature extraction capability; Step 2, freeze the detector backbone and independently distill the transformation module using rainy/rain-free feature pairs; Step 3, freeze the transformation module and the preceding backbone parameters, and only fine-tune the subsequent network layers, allowing the entire cascaded system to adapt to the transformed feature space. This strategy effectively decouples the optimization objectives at different stages, ensuring training stability and avoiding the forgetting problem.

Experiments demonstrate that the proposed method brings rainy detection performance close to the clear-weather baseline while guaranteeing real-time inference, providing a feasible path for the efficient adaptation of vision systems under adverse weather conditions.

2. Methodology

2.1. Problem Formulation and Overall Architecture

We define the input rainy image as a three-dimensional feature matrix with spatial resolution (H×W), and the corresponding rain-free clean image serves as the ground-truth reference. This method adopts ResNet-50[16] as the detection backbone, which outputs five multi-scale feature maps through a series of convolutional downsampling operations. The spatial resolutions of these feature levels are 1/4, 1/8, 1/16, 1/32, and 1/64 of the original image, respectively. To eliminate the interference of rain-streak features on detection accuracy, we embed a custom feature transformation module at a shallow position in the backbone. This module takes the intermediate rainy-image features extracted by the backbone as input, learns a feature mapping, and produces purified rain-free features, with the optimization objective being the minimization of the distance between the purified features and the real clean features. The optimized clean features then replace the original rainy features and continue to flow through the remaining backbone layers, the feature pyramid network, and the detection head to accomplish object classification and bounding box regression proposed in FCOS[2]. This method accomplishes rain-streak removal in the feature space without processing the raw pixel images, thus being able to reuse the semantic priors learned by the pre-trained detector and to reduce computational overhead by operating on lower-resolution feature maps. The overall lightweight detection model with integrated transformation is named B-FCOS, and its architecture is illustrated in Figure 1.

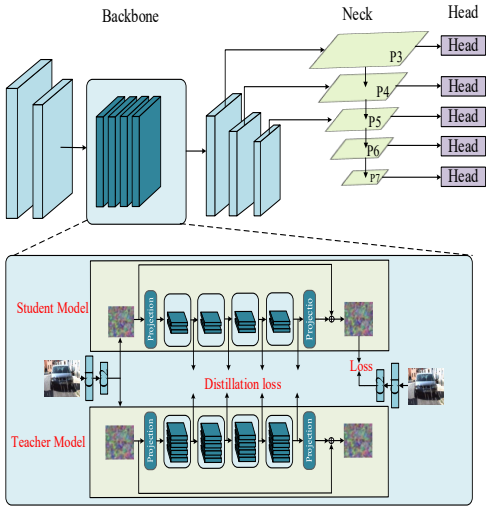

Figure 1. Overall architecture of rainy object detection

2.2. Instantiation Design of the Feature Transformation Module

The feature transformation module designed in this paper is built upon the deraining network proposed in denseformernet[15]. Its basic building block is the CU convolutional block, which consists of a convolutional layer, a batch normalization layer, and a ReLU activation function stacked sequentially. Multiple CU blocks are concatenated to form the core feature extraction pathway of the module. To adapt to the dimensional interfaces of the preceding and subsequent networks, an input projection layer and an output projection layer are placed at the beginning and the end of the module, respectively. The input projection layer regularizes the channel dimension of the input features, while the output projection layer maps the transformed features back to the original channel dimension, ensuring the coherence of the feature flow. The performance and parameter count of the module are jointly determined by two hyperparameters: the internal number of channels and the number of basic layers in each CU block. By varying these two hyperparameters, network variants with different capacities can be constructed. In this work, we set the network with 4 basic layers per CU block and 60 channels as the teacher model, and the network with 1 basic layer per CU block and 36 channels as the student model.

Since the input to the module is an intermediate feature map rather than the raw RGB image, similar to FitNet[17], we set the stride of the input projection layer to 1 to avoid losing feature resolution. Meanwhile, under the default setting, we keep the input channel dimension unchanged and only strengthen the feature representation capacity through nonlinear transformations, thereby preserving the structural information of the original image to the greatest extent.

2.3. Knowledge Distillation Framework Based on Intermediate Feature Maps

Limited by limited capacity, lightweight feature transformation modules struggle to accurately fit complex rain-streak interference mapping relationships, resulting in suboptimal clean feature restoration. To solve this problem, we propose an intermediate feature-based knowledge distillation framework, where a high-capacity teacher network guides the training of a lightweight student network.

The distillation training includes two core stages.

In the first teacher training stage, the detector backbone is frozen. Rainy and clean image pairs are input to extract paired rainy and clean features. The teacher network transforms rainy features and minimizes the L2 distance between restored features and ground-truth clean features, with the feature loss defined as:

$$L_{feature}=\left\|C_{xt}-C_x^{clean}\right\|^2$$

where C_{xt} denotes transformed rainy features, and C_x^{clean} represents ground-truth clean features. The trained teacher network parameters are fixed after convergence.

The second stage is student distillation training. The student and teacher networks share the same rainy feature input. To eliminate channel dimension mismatch, a 1×1 convolution layer $g_s(\cdot)$ aligns student feature channels with teacher features, retaining complete spatial information. The distillation loss is formulated as:

$$L_{Distill}=\left\|F_t-g_s(F_s)\right\|^2$$

where F_t and F_s are teacher and student output features, respectively, and $g_s(\cdot)$ is the channel alignment adapter.

The student network is optimized by a weighted combination of feature restoration loss and distillation loss:

$$L_s=L_{feature}+\delta\cdot L_{Distill}$$

where δ is the balancing coefficient. Ablation experiments verify the optimal δ=5, which effectively balances feature restoration and imitation learning, avoiding overfitting or insufficient distillation effects.

2.4. Three-Step Progressive Joint Training Strategy

Direct end-to-end fine-tuning of the embedded transformation module easily disrupts pre-learned feature mapping and causes training oscillation. To address this issue, we design a three-step progressive joint training strategy for stable module-detector integration.

Step 1: Clear-domain detector pre-training. The original FCOS detector is pre-trained on clean rain-free images with standard detection loss:

$$L_{pre}=L_{cls}+\lambda L_{reg}+L_{cnt}$$

where L_{cls}, L_{reg}, and L_{cnt} denote focal classification loss, GIoU regression loss, and centerness loss, respectively, and λ is the balance weight. This step provides a stable clean feature baseline for subsequent tasks.

Step 2: Frozen-backbone module distillation. The pre-trained backbone is fixed, and only the student transformation module is optimized via joint feature and distillation loss:

$$L_{dist}=L_{feature}+\delta\cdot L_{Distill}$$

where all loss terms and δ=5 are consistent with Section 2.3. The fixed backbone stabilizes feature distribution, accelerating convergence and enabling accurate rain feature removal mapping learning.

Step 3: Cascaded network fine-tuning. The trained transformation module and front backbone are fixed, while the rear backbone, FPN, and detection head are fine-tuned on rainy images with detection loss:

$$L_{fine}=L_{cls}+\lambda L_{reg}+L_{cnt}$$

with consistent loss definitions and weights. This strategy compensates for minor feature transformation deviations, adapts the detector to rainy scenarios, preserves module deraining capability, and avoids catastrophic forgetting.

3. Results and Discussion

3.1. Dataset Construction and Experimental Setup

To support both feature transformation training and detection evaluation, we adopt the RainKITTI dataset[14]. Training set contains approximately 5,000 clear images, and the test set contains about 400 images.Detection performance is measured using the mean Average Precision (mAP, IoU thresholds 0.50:0.95) and mean Average Recall (mAR) from the COCO standard[18].

All models are implemented in PyTorch on a single NVIDIA RTX 3090 GPU. All models are trained using the Adam optimizer. For the detector training stage, we employ a batch size of 4, an initial learning rate of 1×10^{-3}, and train for 30 epochs, with the learning rate multiplied by 0.1 at epochs 22 and 27. For the transformation module, due to GPU memory limitations, the batch size is reduced to 1, the learning rate is set to 1×10^{-4}, and the network is trained for 30 epochs with the learning rate halved at epoch 20. In the final fine-tuning stage, we use a batch size of 4 and a learning rate of 1×10^{-5} for 10 epochs.

3.2. Comprehensive Comparison with Mainstream Preprocessing Methods

We compare our embedded method (B-FCOS, after fine-tuning) with several state-of-the-art deraining methods (SAPNet[19], MARD[20], DCSFN[21], HiNet[22], JDNet[23], MPRNet[24], and the student model itself) used as preprocessing. The results are shown in Table 1. All preprocessing methods first restore the rainy images to rain-free images and then feed them into the original FCOS detector.

Table 1. Performance comparison with object detection methods.

Method Type	Configuration	mAP (%)	mAR (%)	Test Time (ms)
Baseline (rainy)	Original FCOS	44.71	51.34	20.1
Baseline (rain-free)	Original FCOS	57.92	66.59	20.1
Preprocessing	SAPNet	55.93	65.57	86.5
Preprocessing	MARD	56.70	65.97	92.3
Preprocessing	DCSFN	56.66	65.71	78.1
Preprocessing	HiNet	56.80	66.11	95.0
Preprocessing	JDNet	56.88	65.94	88.7
Preprocessing	MPRNet	56.82	65.75	120.5
Preprocessing	denseformernet	57.05	66.25	107.8
Embedded (Ours)	B-FCOS (Net-C_T)	57.59	66.01	31.53

From Table 1, it can be seen that rain causes a significant drop of 13.21 percentage points in mAP for FCOS, highlighting the necessity of rain adaptation. Our embedded method (B-FCOS) achieves 57.59 mAP with an inference time of only 31.53 ms, more than twice as fast as the fastest preprocessing method while also being more accurate. Compared with the rain-free baseline, the mAP gap is reduced to only 0.33, essentially eliminating the negative impact of rain.

Figure 2 presents the object detection results of the original FCOS and B-FCOS on rainy days. The top row shows the results from the original FCOS detector, and the bottom row

shows the results from B-FCOS. It can be observed that the performance improvement of FCOS with the embedded deraining module is reflected not only in fewer missed detections but also in higher confidence scores of the output bounding boxes.

Figure 2. The object detection results of the original FCOS and B-FCOS on rainy days.

3.3. Ablation Studies

To verify the effectiveness of key designs, we conduct ablation studies.

(1) Effect of Feature Transformation Position

We insert the distilled transformation module at the C1 (resolution 1/4) and C2 (resolution 1/8) positions of the backbone, respectively. The detection results are shown in Table 2.

Table 2. Effect of feature transformation position on detection performance

Position	mAP (%)	mAR (%)
C1	56.14	65.35
C2	55.75	65.31

Transformation at the C1 position yields higher accuracy because shallow features preserve the most complete rain-streak details, and early intervention can effectively block rain streaks from propagating to deeper layers. Therefore, all subsequent experiments are conducted at the C1 position.

(2) Knowledge Distillation for Feature Transformation Network

We compare the effects of direct training and distillation training. The results are shown in Table 3.

Table 3. Effect of Knowledge Distillation for Feature Transformation Network performance

Training Method	mAP (%)	mAR (%)
Direct training	55.26	64.40
Distillation training	55.56	64.93

After distillation, mAP increases by 0.30% and mAR increases by 0.53%. This gain indicates that the student network, by imitating the teachers intermediate feature distribution, effectively acquires the teachers ability to suppress rain streaks, demonstrating the effectiveness of feature-level distillation.

4. Conclusion

This paper proposes a lightweight feature transformation module based on knowledge distillation and a joint training method with an object detector to address the problem of detection performance degradation in rainy weather. Unlike traditional pixel-level deraining preprocessing, this method performs rain streak suppression in the feature space and avoids additional preprocessing overhead through an embedded design. Specifically, we design a scalable Net-C transformation architecture that supports transferring knowledge from a large model to a small model via distillation, significantly reducing computational cost while maintaining high accuracy. We also propose a three-step progressive training strategy that effectively decouples detection pre-training, distillation learning, and task fine-tuning, ensuring stable convergence of the joint training. Experiments show that with the smallest model embedded, detection accuracy approaches the rain-free baseline, and inference efficiency is significantly better than that of preprocessing schemes. This work provides a feasible technical path for the efficient and robust deployment of vision tasks under adverse weather conditions.

References

[1] Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6), 1137-1149. https://doi.org/10.1109/TPAMI.2016.2577031

[2] Tian, Z., Shen, C., Chen, H., & He, T. (2019). FCOS: Fully convolutional one-stage object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 9627-9636). IEEE. https://doi.org/10.1109/ICCV.2019.00972

[3] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 779-788). IEEE. https://doi.org/10.1109/CVPR.2016.91

[4] Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 6517-6525). IEEE. https://doi.org/10.1109/CVPR.2017.690

[5] Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. arXiv, arXiv:1804.02767. https://doi.org/10.48550/arXiv.1804.02767

[6] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv, arXiv:2004.10934. https://doi.org/10.48550/arXiv.2004.10934

[7] Li, C. Y., Li, L. L., Jiang, H. L., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., & Wei, X. (2022). YOLOv6: A single-stage object detection framework for industrial applications. arXiv, arXiv:2209.02976. https://doi.org/10.48550/arXiv.2209.02976

[8] Wang, C. Y., Bochkovskiy, A., & Liao, H. Y. M. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. arXiv, arXiv:2207.02696. https://doi.org/10.48550/arXiv.2207.02696

[9] Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., & Ding, G. (2024). YOLOv10: Real-time end-to-end object detection. arXiv, arXiv:2405.14458. https://doi.org/10.48550/arXiv.2405.14458

[10] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), Computer Vision – ECCV

2014 (Vol. 8693, pp. 740-755). Springer. https://doi.org/10.1007/978-3-319-10602-1_48

[11] Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) challenge. International Journal of Computer Vision, 88(2), 303-308. https://doi.org/10.1007/s11263-009-0275-4

[12] Goodfellow, I. J., Mirza, M., Xiao, D., Courville, A., & Bengio, Y. (2013). An empirical investigation of catastrophic forgetting in gradient-based neural networks. arXiv, arXiv:1312.6211. https://doi.org/10.48550/arXiv.1312.6211

[13] French, R. M. (1999). Catastrophic forgetting in connectionist networks. Trends in Cognitive Sciences, 3(4), 128-135. https://doi.org/10.1016/S1364-6613(99)01294-2

[14] Wang, K., Wang, T., Qu, J., Jiang, H., Li, Q., & Chang, L. (2022). An end-to-end cascaded image deraining and object detection neural network. IEEE Robotics and Automation Letters, 7(4), 9541-9548. https://doi.org/10.1109/LRA.2022.3191937

[15] Wang, T., Wang, K., & Li, Q. (2023). Denseformer for single image deraining. International Journal of Applied Mathematics and Computer Science, 33(4), 651-661. https://doi.org/10.34768/amcs-2023-0044

[16] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770-778). IEEE. https://doi.org/10.1109/CVPR.2016.90

[17] Romero, A., Ballas, N., Kahou, S. E., Chassang, G., Gatta, C., & Bengio, Y. (2014). FitNets: Hints for thin deep nets. arXiv, arXiv:1412.6550. https://doi.org/10.48550/arXiv.1412.6550

[18] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), Computer Vision – ECCV 2014 (Vol. 8693, pp. 740-755). Springer. https://doi.org/10.1007/978-3-319-10602-1_48

[19] Zheng, S., Lu, C., Wu, Y., & Yang, Y. (2022). SAPNet: Segmentation-aware progressive network for perceptual contrastive deraining. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) (pp. 52-62). IEEE. https://doi.org/10.1109/WACV51458.2022.00012

[20] Chen, X., Huang, Y., & Xu, L. (2020). Multi-scale attentive residual dense network for single image rain removal. In Proceedings of the Asian Conference on Computer Vision (ACCV). Springer.

[21] Wang, C., Xing, X., Wu, Y., Su, Z., & Chen, Y. (2020). DCSFN: Deep cross-scale fusion network for single image rain removal. In Proceedings of the 28th ACM International Conference on Multimedia (pp. 1643-1651). ACM. https://doi.org/10.1145/3394171.3413562

[22] Chen, L., Lu, X., Zhang, J., Chu, X., & Chen, C. (2021). HINet: Half-instance normalization network for image restoration. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 182-192). IEEE. https://doi.org/10.1109/CVPR46437.2021.00025

[23] Zhao, H., Yap, K. H., Kot, A. C., & Duan, L. (2020). JDNet: A joint-learning distilled network for mobile visual food recognition. IEEE Journal of Selected Topics in Signal Processing, 14(4), 665-675. https://doi.org/10.1109/JSTSP.2020.2980852

[24] Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., & Yang, M. H. (2021). Multi-stage progressive image restoration. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 14821-14831). IEEE. https://doi.org/10.1109/CVPR46437.2021.01458