Parameter-Efficient Image Denoising for Noise-Robust Object Detection

Takamichi Miyata
Faculty of Advanced Engineering
Chiba Institute of Technology
Chiba, Japan
takamichi.miyata@it-chiba.ac.jp

***Abstract*—Object detectors deployed in real-world environments often suffer severe performance degradation when input images are corrupted by noise. While denoise-then-recognize strategies are known to be effective for image classification, their impact on object detection and the choice of suitable denoising front-ends remain unclear. This study presents an efficient noise-robust object detection pipeline that inserts a Gated Texture CNN (GTCNN) as a trainable preprocessing module in front of a Single-Shot multi-box Detector (SSD). GTCNN reduces noise by separately modeling texture and context and fusing them through a gating mechanism. Experiments on noisy images show that denoising consistently improves detection performance over using SSD alone. Our GTCNN front-end achieves detection accuracy comparable to or higher than the conventional heavy-weight denoising network, while requiring only about 3% additional parameters relative to the SSD backbone, demonstrating extremely high parameter efficiency and practical suitability for resource-constrained edge environments.**

***Index Terms*—Object detection, Image denoising, Noise robustness, Efficient neural networks, Edge devices, Gated Texture CNN (GTCNN), U-net**

I. Introduction

Object detection systems are vulnerable to noise in the input image, and their detection accuracy can degrade severely when the input is corrupted. Since detection relies on extracting features from image patterns to localize and classify objects, pixel-level perturbations easily propagate into erroneous high-level representations. A natural way to mitigate this issue is to insert an image denoising network before the detector, and a variety of CNN-based denoisers [1]–[5] have in fact been shown to recover a significant portion of the lost accuracy in noisy conditions [6], [7]. At the same time, however, the image denoising community is increasingly adopting large transformer-based models [8]–[10], where improved restoration accuracy is often achieved at the cost of dramatically increased parameter counts and computational complexity.

While large-scale denoising networks offer impressive performance, they are poorly suited for edge AI scenarios such as smartphones or IoT cameras, where real-time execution, low latency, and energy efficiency are critical [11]. Deploying heavyweight denoisers as a pre-processing stage on these devices would incur substantial computational and memory costs, undermining the latency budget of the downstream detection pipeline. Consequently, there is a strong demand for denoising approaches that achieve a favorable accuracy-efficiency trade-off, enabling robust object detection under noise without relying on large, resource-hungry models.

To address this need, GTCNN [5] (Gated Context CNN) has been proposed as a lightweight and accurate image denoiser. It disentangles noise removal and context extraction into separate components, allowing for tailored architectures and achieving competitive denoising performance with substantially fewer parameters than conventional models. This design is particularly effective at suppressing noise while preserving fine textures and structural details that standard CNN denoisers often over-smooth. However, prior work on GTCNN has focused on signal-level quality measures like PSNR, and has not explored training strategies to explicitly optimize its performance as a preprocessor for high-level vision tasks, such as object detection.

We adapt an intermediate-feature-based training objective, previously used to connect restoration to recognition, to the setting of GTCNN+SSD for noise-robust object detection. The objective encourages the denoiser to preserve detector-relevant representations under noise, emphasizing efficient front-end system design for edge deployment.

As a result, our approach improves detection accuracy in noise and realizes an efficient, end-to-end pipeline of a lightweight denoiser and an object detector suitable for resource-constrained edge platforms.

The major contributions of this paper are as follows:

- We evaluate GTCNN as a parameter-efficient denoising front-end for SSD-based object detection and show that it improves robustness under additive noise in our experimental setting.
- We conduct an empirical sensitivity study on the choice of SSD intermediate layers and distance norms for feature-based training, providing practical guidance for configuring a lightweight denoising front-end.
- Our GTCNN-based front-end achieves detection accuracy comparable to or higher than conventional heavy-weight denoisers, while requiring only about 3% additional parameters relative to the SSD backbone, highlighting its high parameter efficiency and practical suitability for edge environments.

TABLE I: Comparison of denoising (or enhancement) methods for high-level vision tasks. The notation meanings are as follows: "Feature Loss": use of intermediate feature losses; "Task Loss-Free": training without task losses; "Frozen HLN": high-level vision task network kept frozen; "Obj. Det. Task": target task is object detection.

Method	Year	Feature Loss	Task Loss-Free	Frozen HLN	Obj. Det. Task	Denoiser
Liu et al. [6]	2018			✓		U-net [2] like
Liu et al. [12]	2020	✓		✓		U-net [2] like
Mamiya and Miyata [7]	2020	✓	✓	✓		DnCNN [1]
Tran et al. [13]	2024				✓	NAFNet [10]
Ours	2026	✓	✓	✓	✓	GTCNN [5]

II. Related Work

A. Object Detection Methods

Early CNN-based object detectors such as R-CNN [14] (Regions with CNN features) and its successors [15], [16] achieved high accuracy by adopting a two-stage architecture that first generated region proposals and then classified the objects within them. However, the computational overhead of these methods made them unsuitable for real-time applications on embedded or resource-limited edge devices. To address this limitation, SSD [17] (Single Shot Multi-Box Detector) and YOLO [18] (You Only Look Once) introduced a representative one-stage detection framework that simultaneously regresses bounding boxes and predicts class scores directly from the input image, enabling real-time performance without relying on expensive proposal generation. While recent transformer-based approaches [19] have achieved even higher accuracy by leveraging global attention and end-to-end optimization, SSD remains widely used in practical deployments owing to its simplicity, efficiency, and ease of integration into real-world applications [20], [21].

B. Image Denoising Methods

Image denoising aims to recover a clean image from a noisy observation. Early CNN-based denoisers such as DnCNN [1] learned mappings from noisy images to noise components, significantly outperforming traditional model-based methods. U-net [2] introduced an encoder-decoder architecture with skip connections that enables effective multiscale feature extraction and has since become a strong baseline for various image restoration tasks, including denoising. GTCNN [5] further disentangles texture and context information into separate branches and fuses them via a gating mechanism, allowing effective noise suppression while preserving important structures. More recently, transformer-based denoisers such as Restormer [8], SwinIR [9], and NAFNet [10] have leveraged self-attention to capture long-range dependencies, and GTCNN has been shown to remain highly competitive in terms of the accuracy-parameter trade-off even against these state-of-the-art models [22].

C. Robust High-Level Vision Network to Degraded Conditions

As mentioned in Section I, object detectors are vulnerable to input noise, and their performance can degrade significantly under noisy conditions. Existing approaches improve the robustness of high-level vision networks under degraded conditions by coupling a front-end denoising or enhancement module with a task network, yet they differ in how high-level information is exploited and which components are trained.

Liu et al. [6] froze a classification or segmentation network and optimized a denoiser using the task loss, and Liu et al. [12] additionally introduced intermediate feature losses as an additional term in the loss function, thereby encouraging the denoiser to preserve internal representations. These methods employ U-net-inspired multiscale architectures as denoisers. Mamiya and Miyata [7] employ DnCNN and relied solely on intermediate feature discrepancies, removing the task loss and enabling few-class training without retraining the classifier. Tran et al. [13] addressed low-light object detection by placing pre-trained enhancement modules before detectors and treating them as task-agnostic black boxes that are not updated by feature or task losses. They use NAFNet as the part of their enhancement module.

These methods are compared in Table I with respect to four aspects: (i) whether intermediate feature losses are used, (ii) whether training is performed without task losses, (iii) whether the high-level vision task network is kept frozen, and (iv) whether the target task is object detection.

III. Our proposed Pipeline

Our goal is to achieve robust object detection under noise with a parameter-efficient architecture that can be deployed in edge environments. To this end, we adopt GTCNN as a denoising front-end, instead of relying on heavy networks such as U-Net or NAFNet. Furthermore, inspired by prior work [7], we construct our training objective from intermediate feature maps of the VGG-16 backbone in SSD rather than directly using the high-level detection loss of the object detection task directly. In our proposed pipeline, the high-level detection network remains completely frozen and only the GTCNN parameters are updated.

Based on the above design policy, Fig. 1 illustrates the overall pipeline from input to output. During inference, a noisy input image is first fed into the GTCNN, which produces a denoised image. This denoised output is then passed to an off-the-shelf SSD detector consisting of a VGG-16 backbone and its detection heads, from which object classes and bounding boxes are obtained. Although this architecture is conceptually simple, it remains unclear how the GTCNN should be trained so that its denoising is optimally tailored for the downstream high-level vision task.

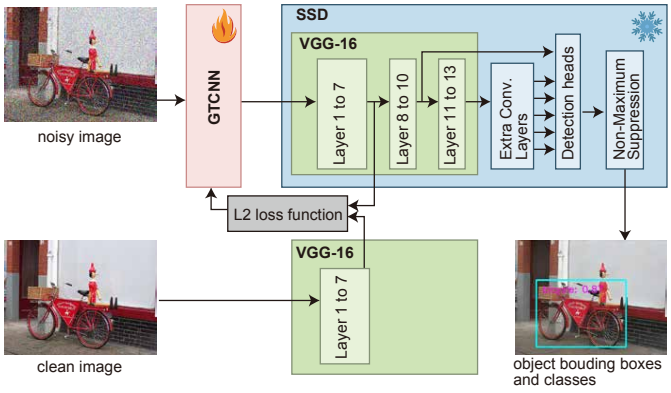

Fig. 1: Overview of our noise-robust object detection pipeline using GTCNN as a denoising front-end for SSD. The training strategy using intermediate feature losses from the SSD backbone are also depicted. Note that we freeze the SSD parameters and update only the GTCNN parameters during training.

In order to train the GTCNN for SSD, we design a learning strategy that leverages intermediate feature representations of the VGG-16 backbone. VGG-16 consists of 16 layers: features from the early layers tend to capture low-level information such as edges and textures, whereas deeper layers provide more abstract, object-level representations. Therefore, an important design choice in our method is which feature maps from these candidate layers are used to construct the feature-based loss function.

Furthermore, another design issue is how to measure the discrepancy between the feature maps at the selected layer, extracted from the denoised image and from the corresponding clean image. In conventional training of denoising networks, the L_2 norm, which is directly related to the widely used metric mean square error (MSE), has predominantly been adopted. However, it has also become increasingly common to employ the L_1 norm [23], [24], which is more robust to outliers. Therefore, in this work we consider these two options, the L_2 norm and the L_1 norm, as candidates for defining the feature-space discrepancy.

Based on the above considerations, in our pipeline we instantiate the feature-space loss using a single intermediate layer and the L_2 norm. Specifically, we adopt the output feature map of layer 7 of the VGG-16 backbone as the intermediate representation. Let $\mathbf{x}$ denote a clean input image and $\hat{\mathbf{x}}$ the corresponding output of the GTCNN, and let $\phi_7(\cdot)$ be the mapping that extracts the layer-7 feature map from VGG-16. The training loss function L for our pipeline is then defined as

$$L(\hat{\mathbf{x}}, \mathbf{x}) = \left\| \phi_7(\hat{\mathbf{x}}) - \phi_7(\mathbf{x}) \right\|_2, \tag{1}$$

where $\| \cdot \|_2$ denotes the L_2 norm. The choice of layer 7 and the L_2 norm is motivated by empirical comparisons, and will be discussed in detail in Section IV-D.

IV. Experimental Results

A. Experimental Setup

Dataset. For the training set, we randomly selected 25 classes from the ImageNet training dataset. As shown in [7], even when using images from a few classes, it is possible to effectively train a denoising front-end for image recognition by utilizing feature loss in the training process. In this study, we evaluate whether a similar trend can be observed in the object detection task. We use the PASCAL VOC 2007 test set as our evaluation dataset. The PASCAL VOC 2007 dataset consists of approximately 5,000 images containing 20 object categories and is widely used as a standard benchmark dataset for object detection tasks.

Evaluation Metric. We use mean Average Precision (mAP) as the evaluation metric for object detection. Following the PASCAL VOC 2007 evaluation protocol, a detection is considered true positive if its Intersection over Union (IoU) with a ground-truth bounding box is at least 0.5. The Average Precision (AP) for each class is computed from its precision–recall (PR) curve using the 11-point interpolation method, and mAP is obtained by averaging the AP values over all classes. Higher mAP indicates better overall detection performance.

Baselines. To evaluate the effectiveness of our proposed pipeline, we compare it against the following baselines. First, we use the case without any denoising ("None") as the simplest baseline. Second, we adopt a denoising method based on U-Net trained with an MSE loss (U-Net w/ MSE). Third, we consider a U-Net-based denoising method that employs a feature-based loss (U-Net w/ VGG). For fairness, the intermediate feature layer and norm in U-Net w/ VGG are fixed to the combination of layer 10 and the L_1 norm, which yielded the best performance among the design choices examined in Section IV-D. By comparing these baselines with our GTCNN-based denoising method (GTCNN w/ VGG (ours)), we quantitatively assess the effectiveness of our approach.

Noise type and levels. Noisy input images are generated by adding i.i.d. Gaussian noise to the clean images. We consider three noise levels with standard deviations σ = 40, 60, and 80.

Implementation details. For the object detector, we use the open-source PyTorch SSD implementation [25] and adopt SSD300 with an input resolution of 300 × 300 pixels. We use the pre-trained weights on the PASCAL VOC 2007+2012 training set provided by the implementation. For the denoising networks, we use the official PyTorch implementations of GTCNN [5].

B. Quantitative Evaluation

To quantitatively evaluate the effectiveness of the proposed method, we compare it with the baseline methods summarized in Table II. We report the mean Average Precision (mAP) of SSD under different noise conditions and denoising configurations, focusing on how each method mitigates the performance degradation caused by additive Gaussian noise.

We first examine the case without any denoising ("None"). When no noise is added (σ = 0), SSD achieves a high

TABLE II: Comparison of the proposed denoising method and baselines at various noise levels. Best results are highlighted in **bold**.

	mAP ↑			
Denoiser	$\sigma = 0$	$\sigma = 40$	$\sigma = 60$	$\sigma = 80$
None	77.26	56.52	37.71	21.44
U-net w/ MSE	-	70.70	64.94	58.53
U-net w/ VGG	-	73.55	**70.18**	65.59
GTCNN w/ VGG (ours)	-	**73.78**	70.16	**66.36**

TABLE III: Comparison of the proposed method and baselines in terms of mAP and number of parameters at noise level $\sigma = 60$. We also show additional parameter ratio (APR), the ratio of additional parameters for denoising network relative to the SSD network (26M parameters).

Denoiser	mAP ↑	Parameters ↓	APR [%] ↓
None	21.44	N/A	N/A
U-Net w/ VGG	**70.18**	17M	63
GTCNN w/ VGG (ours)	70.16	**0.85M**	**3**

mAP of 77.26, indicating that the detector itself performs well on clean images. However, as the noise level increases to $\sigma = 40, 60$, and 80, the mAP drops sharply, showing that the detection performance is severely deteriorated by noise. This confirms our problem setting: SSD is highly vulnerable to input noise, and a suitable denoising front-end is essential for robust detection.

Next, we compare the two U-Net-based baselines, U-Net w/ MSE and U-Net w/ VGG. For all noise levels, both methods significantly improve mAP compared to the "None" case, demonstrating that denoising before detection is effective. Moreover, U-Net w/ VGG consistently outperforms U-Net w/ MSE, indicating that using feature loss derived from the SSD backbone is more beneficial than relying solely on pixel-wise MSE. This result is consistent with our design principle that leveraging intermediate representations of the high-level network as supervision can better preserve task-relevant information for object detection.

We then compare the proposed method, GTCNN w/ VGG, with U-Net w/ VGG. Across the considered noise levels, the proposed method achieves comparable or better mAP. In particular, at $\sigma = 40$ and $\sigma = 80$, GTCNN w/ VGG attains similar or slightly higher mAP than U-Net w/ VGG, while at $\sigma = 60$ it is slightly inferior, but the gap remains very small. These results indicate that the lightweight GTCNN can achieve almost the same level of detection accuracy as the much heavier U-Net when both are trained with the same feature-based loss.

Finally, we discuss parameter efficiency using Table III. U-Net w/ VGG has 17M parameters, whereas GTCNN w/ VGG uses only 0.85M parameters, which corresponds to roughly a 3% increase relative to SSD (26M parameters) alone. In contrast, U-Net w/ VGG increases the parameter count by about 63% compared to SSD, meaning that a substantial portion of the overall model capacity would have to be devoted solely to the preprocessing stage. From the viewpoint of deploying object detection in resource-constrained edge environments, it is undesirable to allocate such a large number of parameters to the denoising front-end, and the proposed GTCNN-based approach is therefore much more attractive in terms of parameter efficiency.

C. Qualitative Evaluation

Fig. 2 presents qualitative results before and after denoising using the proposed method. In this figure, we focus on the noise level of $\sigma = 60$. The comparison between the ground-truth (GT) labels in Fig. 2 (a) and the noisy-image detections in Fig. 2 (b) highlights how severely noise degrades SSD's performance. In the first row, an object that should be labeled as a cat is misclassified as a dog. In the second row, a small car located in the upper-right corner is completely missed, illustrating a typical case of false negatives under noise. In the third row, an object that is not a chair is incorrectly detected as a chair, showing an example of false positives.

In contrast, the detection results on the denoised images in Fig. 2 (c) demonstrate that the proposed method can largely recover the performance of SSD under noisy conditions. Many of the mistakes observed in the noisy case are corrected: the cat is correctly classified, the small car in the corner is successfully detected, and spurious chair detections are suppressed. Overall, the detections on the denoised images are much closer to the ground-truth annotations, indicating that the proposed denoising front-end effectively restores the original capability of SSD.

Regarding the visual appearance of the denoised images themselves, we observe that the outputs of the proposed method sometimes exhibit checkerboard artifacts, especially when zoomed in. Such artifacts are commonly reported in methods that use feature-based or perceptual losses to improve the perceptual quality of images [24]. However, for high-level vision tasks, these patterns, although perceptually noticeable to humans, may still encode information that is beneficial for the detector, as they preserve or even emphasize structures that are important for recognition.

D. Design Choices for Proposed Method and Baseline

In this subsection, we investigate the design choices for the proposed GTCNN w/ VGG, focusing on which intermediate feature layer and norm to use in the feature loss. The backbone VGG-16 in SSD consists of 13 convolutional layers followed by 3 fully connected layers, that is, 16 layers in total. Among these layers, we consider the output feature maps of layers 2, 4, 7, 10, 13, and 15 as candidates for defining the feature loss. Note that layer 15 corresponds to the feature map used in the prior work FDnN [7]. For each of these candidate layers, we also consider two options for measuring feature-

(a) Original image + GT labels (b) Noisy image (c) Denoised image (ours)

Fig. 2: Comparison of object detection results. Noise level $\sigma = 60$. In the noisy image, the detector misses object classes, fails to detect objects, or detects unnecessary objects. In contrast, in the denoised image using the proposed method, the detector accurately recognizes the objects.

space discrepancy, namely the L_1 and L_2 norms, so that the overall design space is given by all combinations of layer index and norm.

To analyze the effect of these design choices on detection performance, we conduct a systematic exploration for the proposed GTCNN w/ VGG. We fix the noise level to $\sigma = 60$ and, for each combination of the all candidate layers and norms, train the GTCNN using the corresponding feature loss. The denoised images produced by each trained model are then fed into SSD, and the resulting mAP is reported in Table IV. As shown in the table, the L_2 norm generally achieves equal or better performance than the L_1 norm for most layers, indicating that L_2 is preferable for the GTCNN in this setting. In particular, the combination of layer 7 and the L_2 norm attains the highest mAP of 70.16. Based on these results, we adopt layer 7 with the L_2 norm as the default loss configuration for the proposed method.

An interesting observation is that the layer that works best for the GTCNN does not coincide with the layers most heavily used by SSD itself. In SSD, intermediate feature maps from layer 10 and layer 13 of VGG-16 are directly exploited for detection. The feature map at layer 13 is fed into an extra feature extractor specific to SSD, which produces five additional feature maps that are used to estimate object locations and classes at multiple scales. From this architecture, one might expect that using layer 10 or layer 13 as the supervision for the denoiser would be the most appropriate choice. However, Table IV shows that a shallower intermediate layer, layer 7, yields the best performance for the GTCNN, suggesting that using slightly earlier, mid-level features as a loss target can be more suitable for a lightweight denoising front-end than directly matching the task-critical layers of the detector.

TABLE IV: Design choices for the proposed method. Noise level $\sigma = 60$.

	mAP ↑					
Norm	layer 2	layer 4	layer 7	layer 10	layer 13	layer 15
L_1	65.05	64.88	67.84	68.03	68.95	67.45
L_2	66.99	68.70	**70.16**	69.32	68.02	67.25

We also perform a similar design-space exploration for the U-Net-based denoiser with feature loss, U-Net w/ VGG, and summarize the results in Table V. Under the same noise level $\sigma = 60$, we train U-Net w/ VGG for all combinations. In this case, the combination of layer 10 and the L_1 norm achieves the highest mAP. All U-Net w/ VGG baseline results in the quantitative comparison are obtained using this best-performing loss configuration.

V. Conclusion

We presented an efficient system design for noise-robust object detection by combining an off-the-shelf SSD detector with

TABLE V: Design choices for the baseline method. Noise level $\sigma = 60$.

	mAP ↑					
Norm	layer 2	layer 4	layer 7	layer 10	layer 13	layer 15
L_1	68.42	68.18	69.07	**70.18**	69.54	66.69
L_2	69.91	69.69	69.94	69.97	69.60	67.69

a lightweight GTCNN front-end trained using intermediate feature supervision. Experiments on the PASCAL VOC 2007 dataset with additive Gaussian noise showed that our GTCNN w/ VGG achieves almost the same mAP as parameter-heavy baseline methods, while requiring only about a 3% increase in parameters relative to the SSD backbone. Furthermore, we quantitatively showed that performance is highly sensitive to the design of the loss function, in particular to which intermediate feature of VGG-16 is used and whether the L_1 or L_2 norm is employed. Even small changes in the selected layer can alter the detection accuracy, and the optimal combination of layer and norm differs between GTCNN and U-Net.

We also found that, for GTCNN, the most effective supervision is obtained by using a higher-resolution, more image-like intermediate feature map from layer 7 with the L_2 norm, whereas for U-Net the best configuration is to use the layer 10 feature map, which SSD mainly relies on, together with the L_1 norm. The fact that the smaller GTCNN benefits more from such "raw" mid-level features is somewhat counterintuitive; one possible hypothesis is that, due to its limited capacity, GTCNN learns more effectively when it is directly encouraged to preserve local, low-to-mid-level structures that indirectly sustain higher-level representations in the downstream detector.

As future work, in addition to a deeper analysis of the behavior in feature space, we plan to measure latency and throughput on actual edge-AI devices, extend the framework to other detectors such as YOLO and Transformer-based models, and investigate robustness under other degradation conditions (e.g., low illumination and motion or defocus blur). Further efficiency improvements, for example by incorporating depth-wise separable convolutions [22] or knowledge distillation, are also promising directions.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number 23K03871, research grant from the Telecommunications Advancement Foundation. The author would like to thank Wang Zheng and Satoshi Nakano for their assistance with the experiments.

References

[1] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Proceedings of the Medical Image Computing and Computer-Assisted Intervention*, 2015.

[3] Y. Tai, J. Yang, X. Liu, and C. Xu, "MemNet: A Persistent Memory Network for Image Restoration," in *Proceedings of the International Conference on Computer Vision*, 2017.

[4] Y. Zhang, K. Li, K. Li, B. Zhong, and Y. Fu, "Residual non-local attention networks for image restoration," in *Proceedings of the International Conference on Learning Representations*, 2019.

[5] K. Imai and T. Miyata, "Gated Texture CNN for Efficient and Configurable Image Denoising," in *Proceedings of the European Conference on Computer Vision Workshop*, 2021.

[6] D. Liu, B. Wen, X. Liu, Z. Wang, and T. Huang, "When image denoising meets high-level vision tasks: A deep learning approach," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2018.

[7] K. Mamiya and T. Miyata, "Few-class learning for image-classification-aware denoising," in *Proceedings of the IEEE International Conference on Image Processing*, 2020.

[8] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, and M.-H. Yang, "Restormer: Efficient transformer for high-resolution image restoration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5718–5729.

[9] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte, "SwinIR: Image restoration using swin transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2021, pp. 1833–1844.

[10] L. Chen, X. Chu, X. Zhang, and J. Sun, "Simple baselines for image restoration," in *Proceedings of the European conference on computer vision*. Springer, 2022, pp. 17–33.

[11] X. Wang and W. Jia, "Optimizing Edge AI: A Comprehensive Survey on Data, Model, and System Strategies," *arXiv preprint arXiv:2501.03265*, 2025.

[12] D. Liu, B. Wen, J. Jiao, X. Liu, Z. Wang, and T. S. Huang, "Connecting image denoising and high-level vision tasks via deep learning," *IEEE Transactions on Image Processing*, vol. 29, pp. 3695–3706, 2020.

[13] D. Q. Tran, A. Aboah, Y. Jeon, M. Shoman, M. Park, and S. Park, "Low-light image enhancement framework for improved object detection in fisheye lens datasets," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024, pp. 7056–7065.

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[15] R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[16] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[17] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proceedings of the European Conference on Computer Vision*, 2016, pp. 21–37.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[19] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proceedings of the European conference on computer vision*. Springer, 2020, pp. 213–229.

[20] H. Zhao, Y. Gao, and W. Deng, "Defect detection using shuffle Net-CA-SSD lightweight network for turbine blades in IoT," *IEEE Internet of Things Journal*, vol. 11, no. 20, pp. 32 804–32 812, 2024.

[21] Y. Wei, Z. Zhao, and K. Gu, "The defect detection method for automotive parts based on improved single shot multibox detector using feature enhancement," in *Proceedings of the International Conference on Signal Processing and Intelligent Computing*, 2024, pp. 891–895.

[22] K. Udoh and T. Miyata, "Parameter efficient image denoising by combining depthwise separable convolution and gating mechanism," *Nonlinear Theory and Its Applications, IEICE*, vol. 16, no. 4, pp. 860–877, 2025.

[23] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 47–57, 2017.

[24] T. Miyata and Y. Aoki, "Perceptual JPEG artifact removal using weighted sum of IQAs as loss function," *Nonlinear Theory and Its Applications, IEICE*, vol. 15, no. 4, pp. 725–736, 2024.

[25] Q. Gao, "pytorch-ssd," https://github.com/qfgaohao/pytorch-ssd, accessed: 2025-12-11.