From Spatial to Spectral: An Efficient, Frequency-Guided Representation Learner for Small Object Detection

Anonymous Authors[1]

Abstract

This document provides a basic paper template and submission guidelines. Abstracts must be a single paragraph, ideally between 4–6 sentences long. Gross violations will trigger corrections at the camera-ready phase.

1. Introduction

2. Related Work

We review prior work from three angles that are most relevant to our goal: (i) efficient detector architectures, (ii) small object detection strategies, and (iii) frequency-domain modeling for dense prediction.

2.1. Efficient Detector Architectures

Real-time detection has been driven by architectural efficiency in backbones, feature pyramids, and heads. One-stage YOLO-style detectors optimize the accuracy–latency trade-off through carefully designed blocks and multi-scale prediction, with recent variants continuing to improve both speed and accuracy (Wang et al., 2024; Khanam & Hussain, 2024). Lightweight enhancements for challenging regimes (e.g., cluttered scenes) often rely on stronger feature aggregation or multi-kernel perception to increase representational diversity while keeping inference efficient (Xiao et al., 2025).

In parallel, Transformer-based detectors seek end-to-end set prediction by removing hand-crafted components such as anchors (Carion et al., 2020). Subsequent work improves the practicality of DETR-like models via more efficient attention and training strategies, enabling competitive performance under constrained budgets (Zhao et al., 2024; Zhang et al., 2022). Despite these advances, both CNN- and Transformer-based detectors still face a common tension for tiny/small objects: improving fine-detail sensitivity typically increases computation, memory, or architectural intrusion, making it difficult to deploy a uniformly effective solution across detector families.

2.2. Small Object Detection

Small objects are inherently information-limited: they occupy few pixels, induce weak feature responses, and are easily suppressed by downsampling and coarse fusion. Early two-stage and one-stage frameworks (e.g., Faster R-CNN and SSD) already revealed the difficulty of preserving small-object cues under feature hierarchy and stride growth (Ren et al., 2015; Liu et al., 2016). A large body of work improves small-object performance by strengthening multi-scale feature fusion (e.g., FPN and its variants) (Lin et al., 2017), introducing additional pyramid levels, and designing attention or alignment modules to enhance small-scale features.

Recent methods increasingly emphasize *detail-aware* feature enrichment. For example, HS-FPN highlights tiny objects by generating high-frequency responses as mask weights and complements this with explicit spatial dependency modeling (Shi et al., 2025). Context modeling (e.g., large receptive fields or multi-kernel designs) also helps disambiguate tiny objects from background clutter (Wang et al., 2025; Xiao et al., 2025). However, many of these approaches focus on either spatial fusion or receptive-field engineering, while the *mechanism of how fine details are suppressed and should be reconstructed* is often left implicit, and portability across heterogeneous detector designs is not always validated.

2.3. Frequency-Domain Modeling for Dense Prediction

Frequency-domain analysis offers a complementary lens to understand and manipulate representation learning. A line of work uses Fourier transforms to achieve efficient global interactions. GFNet replaces quadratic self-attention with frequency-domain filtering (FFT–filtering–IFFT), yielding log-linear complexity while maintaining global receptive fields (Rao et al., 2023). Other work links common architectural operations to spectral decomposition: FcaNet interprets channel attention as a frequency-domain compres-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

sion process and generalizes global pooling to multi-spectral channel attention (?).

More recently, frequency-aware modules have been explored for dense prediction. FDConv observes that candidate dynamic convolution kernels often have highly similar frequency responses, and proposes constructing frequency-diverse weights by allocating parameters to disjoint Fourier indices, together with frequency-band/spatial modulation (Chen et al., 2025). Frequency-aware fusion is also studied: FreqFusion explicitly introduces adaptive low-pass/high-pass filtering to improve feature consistency and boundary sharpness during upsampling and fusion (?). Wavelet-based approaches provide multi-resolution decomposition with partial spatial localization; WTConv performs convolutions in wavelet sub-bands to scale receptive fields efficiently and can be used as a drop-in layer in CNNs (Finder et al., 2025).

While these spectral methods demonstrate that frequency-domain techniques can be integrated into modern architectures, existing designs are often *task- or component-specific* (e.g., classification backbones, fusion-only modules, or specific convolution families), and do not provide a unified, plug-and-play operator that can be instantiated across *backbone, neck, and head and* generalize across both CNN- and Transformer-style detectors. Our work fills this gap by introducing a decomposition–reconstruction operator that preserves and re-synthesizes discriminative spectral components with minimal overhead, and systematically validating its cross-architecture generality.

3. Method

3.1. WDG

We introduce Wavelet-Difference Gate (WDG), a lightweight plug-and-play bottleneck that injects frequency-aware modulation into convolutional backbones. Given an input feature map $\mathbf{x} \in \mathbb{R}^{C\times H\times W}$, WDG first applies a 1×1 projection to hidden channels $C' = \lfloor eC \rfloor$ (with expansion ratio e) and then performs a 2D Haar discrete wavelet transform (DWT) to separate low- and high-frequency components. For simplicity, we describe the transform for even H, W; in practice we align sizes by cropping/padding and restore the original resolution after reconstruction.

Projection and wavelet decomposition. We first project $\mathbf{x}$ to a hidden space and decompose it into Haar subbands:

$$\begin{aligned} \mathbf{x}' &= f_{1\times 1}(\mathbf{x}), \\ \big(\mathbf{x}_{LL}, \mathbf{x}_{LH}, \mathbf{x}_{HL}, \mathbf{x}_{HH}\big) &= \mathrm{DWT}(\mathbf{x}'). \end{aligned} \tag{1}$$

Here $\mathbf{x}_{LL}$ is the low-frequency approximation, and $\{\mathbf{x}_{LH}, \mathbf{x}_{HL}, \mathbf{x}_{HH}\}$ capture horizontal/vertical/diagonal high-frequency details. This decomposition explicitly separates coarse structures from fine details, enabling targeted refinement for small objects.

For Haar DWT/IDWT, each spatial 2×2 block is transformed by a 2×2 Haar matrix. For each channel c and location (u, v), define the local block

$$\mathbf{X}^{(c)}_{u,v} = \begin{pmatrix} \mathbf{x}'^{(c)}_{2u,2v} & \mathbf{x}'^{(c)}_{2u,2v+1} \\ \mathbf{x}'^{(c)}_{2u+1,2v} & \mathbf{x}'^{(c)}_{2u+1,2v+1} \end{pmatrix}. \tag{2}$$

Then Haar DWT and IDWT are given by

$$\begin{aligned} \mathbf{S}^{(c)}_{u,v} &= \tfrac{1}{2}\,\mathbf{H}_2\,\mathbf{X}^{(c)}_{u,v}\,\mathbf{H}_2^{\top}, \\ \mathbf{X}^{(c)}_{u,v} &= \tfrac{1}{2}\,\mathbf{H}_2^{\top}\,\mathbf{S}^{(c)}_{u,v}\,\mathbf{H}_2, \qquad \mathbf{H}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \end{aligned} \tag{3}$$

where $\mathbf{S}^{(c)}_{u,v} = \begin{pmatrix} \mathbf{x}^{(c)}_{LL,u,v} & \mathbf{x}^{(c)}_{LH,u,v} \\ \mathbf{x}^{(c)}_{HL,u,v} & \mathbf{x}^{(c)}_{HH,u,v} \end{pmatrix}$ collects the four subbands. This matrix form is exactly equivalent to the element-wise expressions used in our implementation.

RepCDC for low-frequency refinement. To enhance discriminative edges while keeping computation low, we refine the approximation subband at half resolution:

$$\mathbf{y}_{LL} = f_{\mathrm{cdc}}(\mathbf{x}_{LL}). \tag{4}$$

In our implementation, f_{cdc} is RepCDC followed by normalization and activation. RepCDC parameterizes a central-difference convolution by decreasing the center coefficient of a 3×3 kernel with a learnable $\boldsymbol{\theta}$. Concretely, the effective kernel is

$$\mathbf{y}^{(o)}_{p,q} = \sum_{c}\sum_{i=-1}^{1}\sum_{j=-1}^{1} \mathbf{W}^{(o,c)}_{i,j}\,\mathbf{z}^{(c)}_{p+i,q+j} - \sum_{c} \boldsymbol{\theta}^{(o,c)}\,\mathbf{z}^{(c)}_{p,q}, \tag{5}$$

where $\mathbf{z}$ denotes the input to RepCDC (e.g., $\mathbf{z} = \mathbf{x}_{LL}$), and (p, q) indexes spatial locations. This expression is exactly equivalent to subtracting $\boldsymbol{\theta}$ from the center coefficient of a 3×3 kernel. During deployment, the resulting kernel is fused into a single standard convolution, so RepCDC incurs no extra inference branches. Operating on $\mathbf{x}_{LL}$ reduces spatial cost by $4\times$ while strengthening edge sensitivity through the difference term.

High-frequency gated modulation. We use high-frequency responses to predict a content-adaptive gate and modulate the refined low-frequency feature:

$$\begin{aligned} \mathbf{g} &= \sigma\Big(f_g\big(\mathrm{Concat}(\mathbf{x}_{LH}, \mathbf{x}_{HL}, \mathbf{x}_{HH})\big)\Big), \\ \widetilde{\mathbf{x}}_{LL} &= \mathbf{y}_{LL} \odot (\mathbf{1} + \mathbf{g}). \end{aligned} \tag{6}$$

We use additive gating $(1 + \mathbf{g})$ to emphasize informative regions without suppressing the overall magnitude of $\mathbf{y}_{LL}$. f_g is a 1×1 convolution followed by normalization, and $\mathrm{Concat}(\cdot)$ denotes channel-wise concatenation. Since the gate is predicted from high-frequency subbands, it acts as a detail-aware selector that boosts regions with strong edge/texture cues.

Reconstruction and residual output. Finally, we preserve the original high-frequency subbands and reconstruct the feature via inverse Haar transform:

$$\begin{aligned}\widehat{\mathbf{x}}' &= \mathrm{IDWT}(\widetilde{\mathbf{x}}_{LL}, \mathbf{x}_{LH}, \mathbf{x}_{HL}, \mathbf{x}_{HH}),\\ \mathbf{y} &= f^{\mathrm{out}}_{1\times 1}(\widehat{\mathbf{x}}').\end{aligned} \tag{7}$$

When input/output channels match, WDG uses a residual connection $\mathbf{y} \leftarrow \mathbf{x} + \mathbf{y}$. Since the wavelet-domain refinement operates on $H/2 \times W/2$, WDG adds only a small overhead and can be inserted as a generic bottleneck into different backbone designs. Preserving the original high-frequency subbands avoids over-smoothing and helps retain boundary sharpness after reconstruction.

3.2. LGE and LGE-W

We next improve the neck by introducing Log-Gabor Enhancer (LGE), a plug-and-play high-frequency refinement module applied to intermediate feature maps before multi-scale fusion. LGE is instantiated per feature level and is agnostic to the specific fusion topology (e.g., FPN/PAN/decoder-style aggregation).

Log-Gabor filter bank (LGF). Given a feature map $\mathbf{x} \in \mathbb{R}^{C\times H\times W}$, LGF applies a fixed Log-Gabor filter bank using depthwise convolutions. Let K and S denote the number of orientations and scales. For each channel c, orientation k, and scale s, we compute

$$\mathbf{h}^{(c)}_{s,k} = \mathbf{x}^{(c)} * \mathbf{g}_{s,k}, \tag{8}$$

where $\mathbf{g}_{s,k}$ is a non-learnable Log-Gabor kernel and $*$ is convolution. This produces a set of directional subband responses that explicitly emphasize edges and fine textures while introducing no additional learnable filter parameters.

Learnable aggregation and residual enhancement (LGE). LGE aggregates the subbands with learnable orientation/scale importance. Let $\boldsymbol{\alpha} \in \mathbb{R}^S$ and $\boldsymbol{\beta} \in \mathbb{R}^K$ be learnable logits; we obtain normalized weights by softmax and compute the high-frequency summary

$$\mathbf{h}^{(c)} = \sum_{s=1}^{S}\sum_{k=1}^{K} \mathrm{softmax}(\boldsymbol{\alpha})_s \; \mathrm{softmax}(\boldsymbol{\beta})_k \, \mathbf{h}^{(c)}_{s,k}. \tag{9}$$

We further apply a learnable global scale γ (implemented as a sigmoid-gated parameter) and a local mixing operator f_{mix}:

$$\mathbf{y} = \mathbf{x}_{\mathrm{skip}} + f_{\mathrm{mix}}\big(\sigma(\gamma)\,\mathbf{h}\big). \tag{10}$$

Here $\mathbf{x}_{\mathrm{skip}}$ is either the identity mapping (when channels match) or a 1×1 projection. In our implementation, f_{mix} is a 3×3 convolution (depthwise when C is preserved), making LGE a lightweight residual enhancer that strengthens detail sensitivity prior to subsequent fusion.

Wavelet variant (LGE-W). LGE-W follows the same LGF decomposition and aggregation in Eq. (8)–(10), but replaces f_{mix} with a wavelet-transform convolution (WTConv) when the input/output channels match. This variant injects multi-resolution wavelet-domain mixing with minimal architectural intrusion; when channel dimensions do not match, we fall back to the standard 3×3 convolution for stability.

4. Experiment

4.1. Datasets and Metrics

We evaluate our framework on four benchmarks to demonstrate its robustness and cross-domain generalization: VisDrone2019 (Du et al., 2019), TinyPerson (Yu et al., 2020), UAVDT (Du et al., 2018), and DOTA v1 (Xia et al., 2018). **VisDrone2019** is our primary benchmark and is particularly challenging due to dense small objects and severe scale variation, where most targets are smaller than 50×50 pixels.

We report both accuracy and efficiency, including mAP_{50}, the number of parameters, GFLOPs, model size, and FPS.

4.2. Configuration

The experimental configuration is detailed in Table 1.

Table 1. **Configuration of Training and Testing Experiment Environments.** Detailed hardware and software configuration used for all experiments in this study.

Environment	Parameter
CPU	Intel(R) Xeon(R) Gold 5218R CPU @ 2.10GHz
GPU	NVIDIA A100-PCIE-40GB
VRAM	40 GB
RAM	46 GB
Operating System	Rocky Linux 8.5 (Green Obsidian)
Language	Python 3.10.14
Frame	PyTorch 2.1.0
CUDA Version	12.6

For YOLO-style architectures, models are trained for 300 epochs with an input resolution of 640×640 and batch size 16, using SGD optimization. Unless otherwise specified, Mosaic augmentation is enabled throughout training; we use 4 dataloader workers and disable AMP.

5. Main Results

6. Analyses and Discussion

7. Conclusion

References

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. End-to-end object detection with transformers. *arXiv preprint arXiv:2005.12872*, 2020.

Chen, L., Gu, L., Li, L., Yan, C., and Fu, Y. Frequency dynamic convolution for dense image prediction. *arXiv preprint arXiv:2503.18783*, 2025.

Du, D., Qi, Y., Yu, H., Yang, Y., Duan, K., Li, G., Zhang, W., Huang, Q., and Tian, Q. The unmanned aerial vehicle benchmark: Object detection and tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 370–386, 2018.

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 IEEE/CVF International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

Finder, S. E., Amoyal, R., Treister, E., and Freifeld, O. Wavelet convolutions for large receptive fields. In Leonardis, A., Ricci, E., Roth, S., Russakovsky, O., Sattler, T., and Varol, G. (eds.), *Computer Vision – ECCV 2024*, pp. 363–380, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72949-2.

Khanam, R. and Hussain, M. YOLOv11: An overview of the key architectural enhancements. *arXiv preprint arXiv:2410.17725*, 2024.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2117–2125, 2017.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. SSD: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, pp. 21–37. Springer, 2016.

Rao, Y., Zhao, W., Zhu, Z., Zhou, J., and Lu, J. GFNet: Global filter networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 45(9):10960–10973, September 2023. doi: 10.1109/TPAMI.2023.3263824.

Ren, S., He, K., Girshick, R., and Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, 2015.

Shi, Z., Hu, J., Ren, J., Ye, H., Yuan, X., Ouyang, Y., He, J., Ji, B., and Guo, J. HS-FPN: High frequency and spatial perception fpn for tiny object detection. *arXiv preprint arXiv:2412.10116*, 2025.

Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., and Ding, G. YOLOv10: Real-time end-to-end object detection. *arXiv preprint arXiv:2405.14458*, 2024.

Wang, A., Chen, H., Lin, Z., Han, J., and Ding, G. LSNet: See large, focus small. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Xia, G.-S., Bai, X., Ding, J., Zhu, Z., Belongie, S., Luo, J., Datcu, M., Pelillo, M., and Zhang, L. Dota: A large-scale dataset for object detection in aerial images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3974–3983, 2018.

Xiao, Y., Xu, T., Xin, Y., and Li, J. FBRT-YOLO: Faster and better for real-time aerial image detection. *arXiv preprint arXiv:2504.20670*, 2025.

Yu, X., Gong, Y., Jiang, N., Ye, Q., and Han, Z. Scale match for tiny person detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1257–1265, 2020.

Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L. M., and Shum, H.-Y. DINO: DETR with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022.

Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., Liu, Y., and Chen, J. Detrs beat yolos on real-time object detection. *arXiv preprint arXiv:2304.08069*, 2024.

A. You *can* have an appendix here.

You can have as much text here as you want. The main body must be at most 8 pages long. For the final version, one more page can be added. If you want, you can use an appendix like this one.

The `\onecolumn` command above can be kept in place if you prefer a one-column appendix, or can be removed if you prefer a two-column appendix. Apart from this possible change, the style (font size, spacing, margins, page numbering, etc.) should be kept the same as the main body.