ThinFormer: Channel Sparse Transformer for Efficient HRW Object Detection

Wenxi Li[3,4], **Kunpeng Liu**[1,2], **Moran Liu**[1,2], **Shuyang Liu**[3], **Chenyang Lyu**[5], **Haozhe Lin**[1] and **Yuchen Guo**[1†]
[1]Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, China,
[2]Department of Automation, Tsinghua University, Beijing, China,
[3]KLATASDS-MOE, School of Statistics, East China Normal University, Shanghai, China,
[4]Zhuoxi Lab, Hangzhou, China,
[5]Alibaba Group, Hangzhou, China
wxli@sfs.ecnu.edu.cn, liukp22@mails.tsinghua.edu.cn, yuchen.w.guo@gmail.com

Abstract

Object detection in high-resolution wide (HRW) shots presents unique challenges due to the extreme sparsity of objects and the variability in sparsity ratios across images. Conventional detectors, designed for close-up settings like MS COCO, struggle to generalize to these scenarios, leading to inefficient computation and degraded performance. While prior work has focused on spatial sparsity, the potential of sparsity in the channel dimension remains largely underexplored. In this paper, we propose ThinFormer, a dynamic and efficient framework tailored for object detection in HRW shots. ThinFormer leverages sparsity in both the spatial domain and the channel domain, the latter enabled by a novel local feed-forward network (local FFN) that exploits the duality between spatial and channel representations. To adapt to varying levels of scene complexity, we introduce a dynamic sparsity ratio estimator, trained in a self-supervised manner, which guides selective activation of channels based on semantic richness. Extensive experiments on the PANDA benchmark demonstrate that ThinFormer significantly reduces computational cost while maintaining competitive detection performance compared to state-of-the-art methods. Our results highlight the effectiveness of embracing multi-domain sparsity for scalable and robust object detection in complex, real-world HRW imagery. Code and supplementary material are available at https://github.com/LiuKunpeng03/Thinformer.

1 Introduction

Object detection is a fundamental yet challenging topic in computer vision. Close-up settings such as MS COCO [Lin *et al.*, 2014] have witnessed impressive performance with successful real-world applications. Nowadays, object detection in high-resolution wide (HRW) shots is attracting more and more attention [Wang *et al.*, 2020; Li *et al.*, 2024a; Li *et al.*, 2024b; Liu *et al.*, 2024]. As trivially adopting vanilla close-up detectors results in unsatisfying performance, difficulties unique to such setup come into view. The extreme sparsity of objects and the variability of the sparsity ratio among shots embarrass conventional close-up detectors.

As found in PANDA [Wang *et al.*, 2020] and DOTA [Xia *et al.*, 2018], the most significant feature of HRW shots lies in the sparsity of information, as no more than 5% area of the image contains some objects. Such characteristic imposes challenges on backbones, which are supposed to effectively extract key features out of inundating background noise. On the other hand, the sparsity ratio varies violently among images. Conventional backbones tend to suffer from deteriorated performance and high computational expense. They lack the ability to distinguish between foreground and background, not to mention the adaptivity to variable sparsity ra-

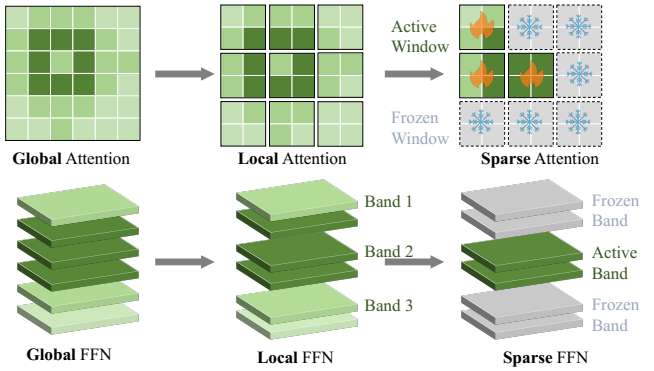

Figure 1: Illustration of the sparsification techniques in Thinformer. Local attention is derived by restricting the queries and keys in the same spatial group *window*. Similarly, local FFN is derived by restricting the calculation in the same channel group *band*. Additionally, weights of local FFN in this paper is modified to be dynamic, which separate it from a trivial generalization of FFN. Furthermore, only a proportion of windows/bands undergo attention/FFN layer. It dramatically reduces the computational expense.

†Corresponding author.

tio.

To address the difficulties above, a dynamic sparsification mechanism is necessary. Previous works mainly focus on taking advantage of the spatial sparsity of HRW shots by local techniques [Li *et al.*, 2024a]. However, there is still a large gap from optimality. Inspired by the spectral sparsity of images and videos [Hassanieh *et al.*, 2012], we notice the sparsity in channel dimension has long been ignored. This motivates us to generalize existing sparsification method on spatial positions to channels (Fig. 1).

We propose a dynamic framework named ThinFormer to take full advantage of the inherent sparsity of HRW shots, both in the spatial and the channel domain. In the spatial domain, Thinformer inherits existing local attention method; in the channel domain, it adopts a novel approach called local FFN, which is designed according to the delicate duality between spatial domain and channel domain. Furthermore, the weights of local FFN are dynamic since they are calculated from the inputs, distinct from the trivial generalization of the static weights of FFN. Owing to the sparsity of both domains, ThinFormer obtains significantly economic computational expense, while maintaining comparable performance with state-of-the-arts. Global attention and FFN are still preserved but in a coarse form to facilitate understanding of global features. Additionally, we implement a dynamic sparsity ratio estimator to address the variability of semantic complexity in images.

In summary, our main contributions are as follows:

- We propose a dynamic architecture for object detection in HRW shots, called ThinFormer. Our methodology takes full advantage of sparsity inherent in HRW shots in both spatial positions and channels. In pursuit of sparsity of channels, we propose a novel dynamic sparse feedforward block, named local FFN.
- We propose a dynamic sparsity ratio generator to flexibly deal with shots of different levels of complexity. The generator is trained in a self-supervised way for the sake of self-consistency. A series of accessory score nets is incorporated. The effectiveness of Thinformer is evaluated under a unified theoretical framework for sparsity in models.
- We extensively validate our method on a large-scale HRW-shot benchmark PANDA. Our model gains significant improvement in efficiency and maintains comparable performance compared with state-of-the-arts.

2 Related Works

Close-up shot detection models. Close-up object detection primarily focuses on megapixel-level datasets like PASCAL VOC [Everingham *et al.*, 2010] and MS COCO [Lin *et al.*, 2014], leading to the development of two main detection architectures: single-stage [Redmon *et al.*, 2016; Redmon and Farhadi, 2018; Ge *et al.*, 2021; Liu *et al.*, 2016] and two-stage [Girshick *et al.*, 2014; Girshick, 2015; Ren *et al.*, 2015; He *et al.*, 2017; Cai and Vasconcelos, 2018]. In recent years, there has been a growing interest in anchor-free approaches [Zhou *et al.*, 2019; Tian *et al.*, 2022], especially with the introduction of the Transformer-based detection architecture, DETR [Carion *et al.*, 2020]. However, these methods are generally designed for megapixel-level detection. Even those claiming real-time capabilities fail to meet the real-time requirements in gigapixel scenarios.

High-resolution wide shot detection models. The benchmark PANDA [Wang *et al.*, 2020] is the first gigapixel-level dataset for HRW shots detection and has recently attracted significant attention. Some prior works focus on reducing latency through techniques like patch selection or arrangement [Najibi *et al.*, 2019; Fan *et al.*, 2022; Chen *et al.*, 2022; Li *et al.*, 2024b; Li *et al.*, 2024c]. GigaHumanDet [Liu *et al.*, 2024] leverages corner modeling for gigapixel-level full-body detection, especially in crowded scenes. While these methods address speed issues in relatively straightforward ways without deeply considering the architectural aspects, we found that the architecture of the detection model itself is the key factor influencing inference speed.

Efficient computation in vision transformers. Some prior works have explored sparse execution strategies, including dynamic selection of patches [Rao *et al.*, 2021], self-attention heads [Meng *et al.*, 2022], dimensions [Li *et al.*, 2021; Li *et al.*, 2022], and Transformer blocks [Meng *et al.*, 2022], primarily for image classification tasks. PnP-DETR [Wang *et al.*, 2021] proposed a poll-and-pool sampler to abstract image features from the backbone and feed sparse tokens to the attention encoder, demonstrating effectiveness across object detection, panoptic segmentation, and image recognition. However, sparse sampling strategies applied directly on the backbone have not been thoroughly investigated. DGE [Song *et al.*, 2021] is a general-purpose plugin for Vision Transformers, but it is not compatible with ConvNet-based models and cannot handle arbitrary input sizes. While the challenge of designing a flexible and model-agnostic architecture for object detection in high-resolution wide (HRW) shots remains largely unexplored, SparseFormer [Li *et al.*, 2024a] proposed a spatial sparsification strategy. However, it fails to account for the redundancy in the channel dimension.

3 Methodology

3.1 Overall Architecture

The overall architecture of our model is shown in Fig. 2. ThinFormer follows the conventional paradigm of attention-based backbones like Swin Transformer. After position embedding phase, the input image is split into non-overlapping patches. Such division exists in both spatial and channel domains: neighboring positions (in spatial domain) are grouped into *windows* and channels are grouped into *bands*. In this paper, a window consists of 7x7 neighboring spatial positions and a band consists of 8 contiguous channels.

The feature-extracting procedure mainly consists of 4 stages, each of which begins with a window-merging layer that concatenate the features of 2x2 neighboring windows. The concatenated features are then projected to half of the previous dimension using a fully connective layer. Then the feature map is sent to the core components: *global blocks* and *local blocks*. A global block consists of a global multi-head

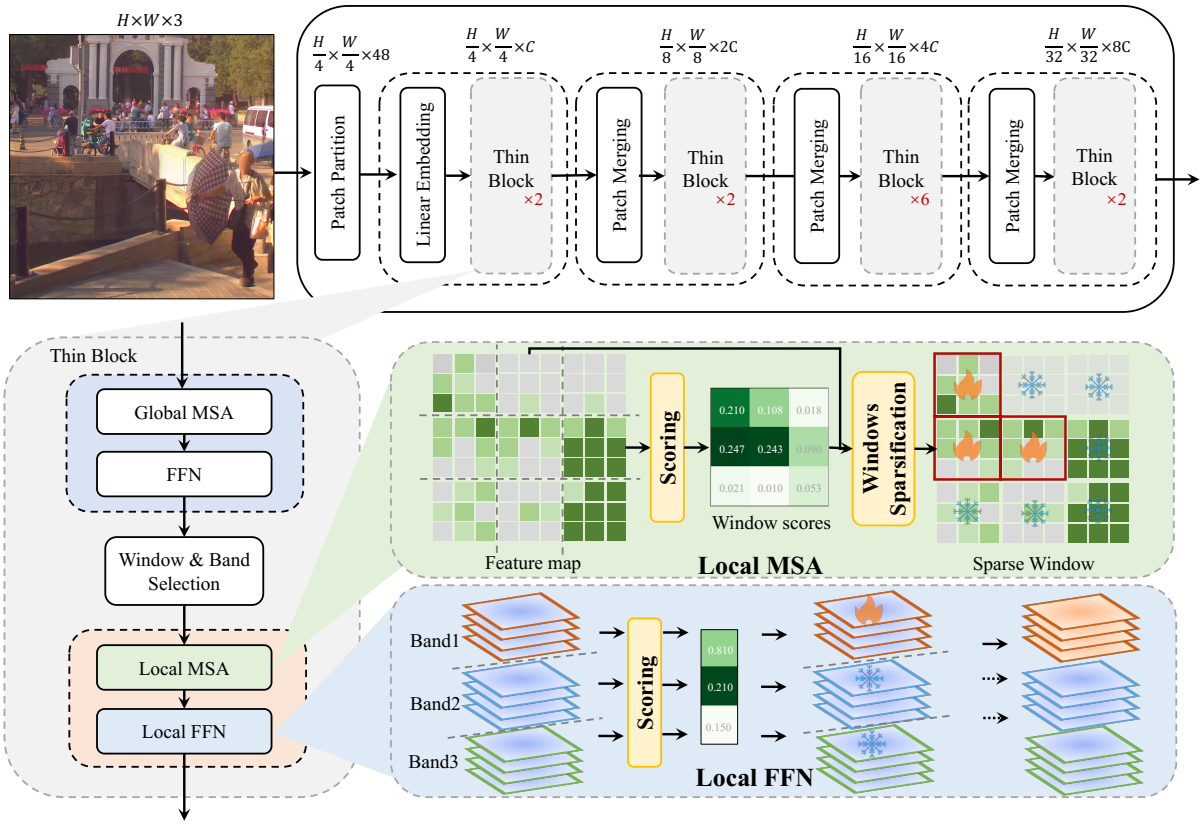

Figure 2: Overall architecture of Thinformer. Thinformer consists of a patch embedding block and 4 sequential stages. Each stage comprises several global blocks and local blocks. Aggregated feature map undergoes global MSA and FFN first. An adaptive generator decides on the sparsity ratio for each stage accordingly. Then a corresponding proportion of windows and bands is selected out for local blocks. There the selected ones go through local MSA and local FFN. Finally, the output of local blocks are merged back to that of global blocks, and is sent to the next stage.

self-attention (MSA) layer and a feedforward network (FFN). Similarly, a local block consists of a local MSA layer and a local FFN. The global blocks are aimed at extracting global features coarsely, while the local blocks focus on fine-grained analysis within a certain window and band. In this way, we strike the balance between performance and efficiency.

Taking into account the potential variability of sparsity of different pictures, we implement a dynamic sparsity ratio generator and a corresponding rating network for each stage. On entering a stage, each window or band is rated according to the richness of information it carries. Meanwhile, the generator performs a comprehensive evaluation on the feature map and decides on a reasonable sparsity ratio accordingly. The sparsity ratio and the scores altogether decide whether a certain window or bands should be sent to local blocks for further fine-grained feature extraction.

3.2 Sparsity in both Spatial and Channel Domain

The main concern of the backbone is how to extract meaningful features as efficiently as possible. In this section, we shall introduce in detail our key solution: a sequence consists of global blocks and local blocks.

Global attention on aggregated features. Some semantics of an image can only be interpreted by taking global information into account. That is why the ability to understand high-level global features is an essential competence of a backbone. We adopt the global information extracting pipeline proposed in (Li et al. 2024). We take the input feature map z and divide it into windows of size M. The left-top location of a window is denoted (x, y), and other locations is determined by its relative deviation $(\Delta x, \Delta y)$. The aggregated feature can be derived as follows:

$$\bar{z}_{x',y'} = \frac{\sum_{\Delta x,\Delta y} \alpha_{\Delta x,\Delta y} \cdot z_{x+\Delta x,y+\Delta y}}{\sum_{\Delta x,\Delta y} \alpha_{\Delta x,\Delta y}}, \tag{1}$$

where $x' = \frac{x}{M}, y' = \frac{y}{M}$. In this paper, we simply let $\alpha_{\Delta x,\Delta y} \equiv 1$, which indicates that the aggregated feature is the algorithmic average of information within the window. This aggregation dramatically cuts computational expense, as it is actually equivalent to $M\times$ downsampling. These aggregated feature map proceeds to be sent to trivial multi-head self-attention blocks, which completes the extraction of global features. We shall rewrite it in a formula form:

$$\bar{z}^{l+1} = f(\bar{z}^l, \Theta), \tag{2}$$

where z^l represents the feature map after the l-th global block. Finally the features are decompressed to the shape of the input of current stage, using the following inverse formula:

$$z_{x+\Delta x,y+\Delta y} = \alpha_{\Delta x,\Delta y} \cdot \bar{z}_{x',y'}, \tag{3}$$

where $x' = \frac{x}{M}, y' = \frac{y}{M}$, as before.

Local attention within selected windows and bands. Global blocks offer only coarse semantics. To extract fine-grained features while avoiding expensive computational cost, prior works exploit spatial sparsity but neglect channel sparsity. We therefore propose local blocks that leverage sparsity in both domains. Inspired by signal processing, where filtering is applied to retain informative frequency bands, we treat channels as a transform domain and introduce a local block equipped with a joint selection mechanism over windows and bands.

We partition the feature map z of size $H \times W \times C$ into $N_S = \frac{H}{M} \times \frac{W}{M}$ windows and $N_T = \frac{C}{D}$ bands, i.e. a window consists of $M \times M$ spatial position and a band contains D channels. Be aware that such a partition is parallel, i.e. a window is of size $M \times M \times C$ and a band is of size $H \times W \times D$. Given an input feature map z, we allocate each window and each band a score based on their information richness and generate a sparse ratio k based on holistic statistics (Details in the following section). Taking into account that the sparsity ratio may be different between windows and bands, we introduce a hyperparameter μ to indicate such difference. In other words, $K_S = \lfloor kN_S \rfloor$ windows and $K_T = \lfloor \mu kN_T \rfloor$ bands will undergo further feature extraction. Windows or bands with higher scores are preserved for subsequent processes. Technically, we maintain two binary decision masks,$A_S \in \{0,1\}^{N_S}$ and $A_T \in \{0,1\}^{N_T}$, subject to $\mathbf{1}^T A_S = K_S$ and $\mathbf{1}^T A_T = K_T$, to indicate whether a window or a band should be preserved. Suppose the patched feature map is $Z \in \mathbb{R}^{N_S \times N_T \times N}$, then the sparse form of it can be derived as follows:

$$\hat{Z}_S = (A_S \otimes A_T^T) \odot Z, \tag{4}$$

where $\otimes$ denotes Kronecker product and $\odot$ denotes Hadamard product, and $N = M \times M \times D$ is the amount of entries in the intersection of a window and a band.

Inspired by spatial-channel duality [Han *et al.*, 2022], we generalize local MSA (a sparsified global MSA) to its channel counterpart, local FFN (a sparsified FFN). Local MSA performs MSA *within each window* with shared Q, K, V matrices, yet produces distinct window-wise weights via dynamic self-attention. Analogously, performing FFN *within each band* with shared weights gives a form of local FFN, but the static nature of FFN yields identical weights across bands, which we term *static local FFN*. The name *local FFN* is reserved for our proposed dynamic form, referred to as *dynamic local FFN* when necessary.

It is widely known that one-layer FFN can be viewed as point-wise convolution:

$$\text{FFN}(x;\Theta) \equiv \text{Conv}(x;\Theta). \tag{5}$$

To allow the weight to vary according to the input, we calculate the weight by the formula

$$\tilde{\Theta}(x) = \text{Conv}(\text{BatchNorm}(\text{Conv}(\text{AvgPool}(x)))). \tag{6}$$

Hence local FFN can be presented as

$$\text{LocalFFN}(x) = \text{Conv}(x;\tilde{\Theta}(x)), \tag{7}$$

where x is the intersection of a window and a band, i.e. $x \in \mathbb{R}^{M \times M \times D}$.

In summary, the sparse feature map $\hat{Z}_S$ undergoes the following operations:

$$\hat{Z}_S^{l+1} = \text{LocalFFN}(\text{LocalMSA}(\hat{Z}_S^l)), \tag{8}$$

where Z_S^l denotes sparse feature map after the l-th local block. The output of local blocks is merged back to the output of global blocks, in the following manner:

$$\hat{Z} \leftarrow (A_S \otimes A_T^T) \odot \hat{Z}_S + (\mathbf{1} \otimes \mathbf{1}^T - A_S \otimes A_T^T) \odot Z, \tag{9}$$

where Z is the output of global blocks, $\hat{Z}_S$ is the output of local blocks, and $\hat{Z}$ is the output of an entire stage.

3.3 Dynamic Sparsity Ratio and Rating Network

The sparsity ratio may vary sharply among shots. Using a pre-defined sparsity ratio regardless of the specific case of individual shot inevitably leads to inefficiency. Therefore, we implement a sparsity ratio generator and a rating network for each stage.

When a feature map z is proceeded to the next stage, each band or window of it is rated according to the variance of information it carries:

$$\text{BandsRating}(z) = \text{Softmax}(\text{MLP}(\tilde{z})), \tag{10}$$

$$\text{WindowsRating}(z) = \text{Softmax}(\text{MLP}(\tilde{z})), \tag{11}$$

where $\tilde{z} = \text{MaxPool}(z) - \text{AvgPool}(z)$ somehow indicates the variance.

In analogous to Nyquist Sampling Theorem [Nyquist, 1928], having estimated the importance of each window and band, we are able to come up with a reasonable sparsity ratio, by

$$\text{kGenerator}(z) = \text{Sigmoid}(\text{MLP}(\tilde{z})). \tag{12}$$

Given that vanilla Sigmoid function tends to push input towards 0 or 1, which is not desired, we insert an affine transformation before Sigmoid, i.e.

$$\text{kGenerator}(z) = \text{Sigmoid}(\alpha(\text{MLP}(\tilde{z}) + \beta)), \tag{13}$$

where α and β are learnable parameters.

To make end-to-end optimization possible, we introduce a self-supervised style loss term $\mathcal{L}_k$ for training sparsity ratio generator:

$$\mathcal{L}_k = \left| k - \frac{S_{union}}{S} \right|, \tag{14}$$

where k denotes the predicted sparsity ratio, S_{union} denotes the area of the union of all predicted bounding boxes and S denotes the area of the entire shot. Heuristically, k and S_{union}/S should be as close as possible, which means the model is able to estimate the sparsity ratio reasonably and consistently. Hence, the total loss is as follows:

$$\mathcal{L} = \lambda_{cls}\mathcal{L}_{cls} + \lambda_{bbox}\mathcal{L}_{bbox} + \lambda_k\mathcal{L}_k. \tag{15}$$

Detector	Backbone	GFLOPs	Param.	AP	AP_S	AP_M	AP_L
FasterRCNN [Ren *et al.*, 2015]	ResNet-101 [He *et al.*, 2016]	140.97	42.50M	-	19.0	55.2	74.4
FasterRCNN [Fan *et al.*, 2022]	ResNet-50 [He *et al.*, 2016]	73.97	23.51M	70.5	20.3	71.2	76.0
RetinaNet [Lin *et al.*, 2017]	ResNet-101	140.97	42.50M	-	22.1	56.1	74.0
CascadeRCNN [Cai and Vasconcelos, 2018]	ResNet-101	140.97	42.50M	-	22.7	57.9	76.5
YOLOv13 [Lei *et al.*, 2025]	CSPNet	111.58	27.60M	58.5	30.2	62.6	66.8
ClusDet [Yang *et al.*, 2019]	ResNet-50	73.97	23.51M	71.8	21.9	69.6	78.2
DMNet [Li *et al.*, 2020]	ResNet-50	73.97	23.51M	54.0	11.9	37.1	71.4
PAN [Fan *et al.*, 2022]	ResNet-50	73.97	23.51M	71.5	25.6	71.9	76.8
GigaHumanDet [Liu *et al.*, 2024]	Hourglass-52 [Newell *et al.*, 2016]	271.65	54.22M	79.6	48.5	78.8	84.8
DINO [Zhang *et al.*, 2023]	Swin-T [Liu *et al.*, 2021]	81.20	27.52M	60.6	36.7	61.2	64.9
DINO	DEG [Song *et al.*, 2021]	40.73	28.25M	58.2	33.9	57.8	62.4
DINO	SparseFormer [Li *et al.*, 2024a]	33.84	53.53M	78.0	50.8	78.1	82.3
DINO	Ours	7.24	36.34M	80.6	52.6	79.8	86.0

Table 1: Comparison with the SoTAs on PANDA. GFLOPs and Param. only consider that of the backbone.

Method	GFLOPs	Param.	AP	AP_S	AP_M	AP_L
Dynamic FFN (Dynamic k)	7.24	36.34M	80.6	52.6	79.8	86.0
Dynamic FFN (k=.1)	7.10	36.34M	80.6	52.7	80.0	85.8
Dynamic FFN (k=.5)	18.34	36.34M	80.2	51.1	79.0	85.5
Static FFN (Dynamic k)	7.25	36.29M	80.1	50.4	79.8	85.9
Static FFN (k=.1)	7.10	36.29M	80.7	52.0	81.1	85.6
Static FFN (k=.5)	18.35	36.29M	80.5	51.5	80.3	85.7

Table 2: Ablation Study on Dynamic Mechanisms. For simplicity, the sparsity ratio in the spatial domain is referred to as k.

To train the rating network, we adopts the Gumbel-Softmax trick [Maddison *et al.*, 2017] to bridge the gap of gradient back-propagation between soft values and binary values through re-parameterization. In other words, before the output of global blocks Z are sent to local blocks, the estimated scores of windows s_S and scores of bands s_T are merged into it, by

$$Z \leftarrow (1 - s_S)(1 - s_T)Z, \tag{16}$$

where s_S and s_T denotes the scores of windows and bands, respectively.

4 Theoretical Analysis

In this section, we establish a theoretical foundation for ThinFormer, focusing on the relationship between dynamic sparsity (as used in ThinFormer) and classical static sparsity. We show that while dynamic sparse networks can be approximated by static ones, there exists a fundamental gap in representation efficiency that explains the empirical advantages of our approach.

4.1 Unified Framework for Sparsity

We begin by formulating a unified framework that encompasses both static and dynamic sparsity, which are also referred to as *model sparsity* and *ephemeral sparsity* in the literature [Hoefler *et al.*, 2021]. Let $f(\mathbf{x}; \mathbf{w}, \mathbf{m})$ denote a neural network with parameters $\mathbf{w}$ and binary mask $\mathbf{m}$ indicating active parameters. The most general sparse learning objective can be written as:

$$\begin{aligned} \min_{\theta} \quad & \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\mathcal{D}}[\ell(f(\mathbf{x}; \mathbf{w}(\mathbf{x};\theta), \mathbf{m}(\mathbf{x};\theta)), \mathbf{y})] \\ \text{s.t.} \quad & \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[\gamma(\mathbf{w}(\mathbf{x};\theta), \mathbf{m}(\mathbf{x};\theta), \mathbf{x})] \le C, \end{aligned} \tag{17}$$

where γ measures computational complexity (e.g., FLOPs) and C is a budget. Both $\mathbf{w}$ and $\mathbf{m}$ can depend on the input $\mathbf{x}$ through parameters θ. This formulation captures:

- **Model (Static) Sparsity**: $\mathbf{w}$ and $\mathbf{m}$ are input-independent.
- **Ephemeral (Dynamic) Sparsity**: $\mathbf{w}$ is input-independent but $\mathbf{m}$ depends on $\mathbf{x}$.
- **ThinFormer's Dynamic Sparsity**: Both $\mathbf{w}$ and $\mathbf{m}$ depend on $\mathbf{x}$.

One should be aware that the loss function (15) is exactly with the form of the Lagrangian dual of (17).

4.2 Dynamic-to-Static Approximation

We now analyze the relationship between dynamic and static sparse layers in ThinFormer. Consider a simplified model where channels are partitioned into N_T bands (blocks), each of dimension D. Let $x_i \in \mathbb{R}^D$ denote the i-th band of input x, i.e. $x = [x_1, \cdots, x_{N_T}]^\top$.

Definition 1 (Dynamic Sparse Layer)**.** *A dynamic sparse layer is defined as $f_{dyn}(x) := [y_1^{dyn}, \cdots, y_{N_T}^{dyn}]^\top$, where*

$$y_i^{dyn} = x_i + m_i(x) \cdot \sigma(W_i'(x)x_i), \quad i = 1, \ldots, N_T,$$

$m_i(x) \in \{0, 1\}$ is a dynamic mask and $W_i'(x) \in \mathbb{R}^{D\times D}$ are dynamic weights. σ is the activating function.

Definition 2 (Static Sparse Layer)**.** *A static sparse layer is defined as $f_{static}(x) := [y_1^{static}, \cdots, y_{N_T}^{static}]^\top$, where*

$$y_i^{static} = x_i + m_i \cdot \sigma(W_i x_i), \quad i = 1, \ldots, N_T,$$

$m_i \in \{0, 1\}$ is a fixed mask and $W_i \in \mathbb{R}^{D\times D}$ are fixed weights. σ is the activating function.

Assumption 1 (Boundedness)**.** *There exist constants* $R, B, B' > 0$*, s.t.* $\forall x \sim \mathcal{D}, i = 1, \cdots, N_T$*: (1)* $\|x\|_2 \leq R$ *and* $\|x_i\|_2 \leq R/\sqrt{N_T}$*; (2) The expected weights* $\bar{W}_i = \mathbb{E}[m_i(x)W_i'(x)]/\mathbb{E}[m_i(x)]$ *satisfy* $\|\bar{W}_i\|_F \leq B$*; (3)* $\|W_i'(x)\|_F \leq B'$ *when* $m_i(x) = 1$*.*

Assumption 2 (Concentration)**.** $\forall i = 1, \cdots, N_T$*,* $m_i \sim$ *Ber*(p_i) *and* $W_i'(x)$ *follows a sub-Gaussian distribution.*

Assumption 3 (Activating Function)**.** σ *is* L_σ*-Lipschitz with* $\sigma(0) = 0$*.*

Theorem 1. *Under Assumption 1,2,3, given a dynamic sparse layer* f_{dyn}*,* $\exists \epsilon_0 > 0, \forall \epsilon > 0$*, there exists a static sparse layer* f_{static} *such that with probability at least* $1 - \delta(\epsilon)$*:*

$$\|f_{static}(x) - f_{dyn}(x)\| \leq \epsilon_0 + \epsilon,$$

where $\delta(\epsilon) = O(\exp(-c \cdot \epsilon^2))$ *for some constant* $c > 0$*.*

Proof Sketch. The construction chooses a threshold τ and activates bands with $p_i \geq \tau$. The error comes from two sources: (1) approximation error in active bands, bounded by concentration of $W_i'(x)$ around $\bar{W}_i$; (2) residual activation in inactive bands ($p_i < \tau$), bounded by τ. We can further show that $\tau = O(\epsilon)$ for this construction. The full proof is provided in the supplementary material. □

Theorem 1 shows that dynamic sparse layers can be approximated by static ones, but with crucial caveats.

To achieve error ϵ, the static model must activate all bands with $p_i \geq \tau(\epsilon) = O(\epsilon)$. For small ϵ, $\tau(\epsilon)$ becomes small, requiring activation of most bands. This means: (i) A static model with fixed sparsity ratio cannot achieve arbitrarily small error; and (ii) To achieve a given error bound, a static model needs lower sparsity (more computation) than the dynamic model.

The inability to drive ϵ arbitrarily small without losing sparsity reveals a *fundamental representation gap* between dynamic and static sparsity. Dynamic sparsity implements a form of *conditional computation* that cannot be efficiently simulated by static sparsity with comparable computational budget.

Our theoretical analysis validates key design choices in ThinFormer:

- **Dynamic FFN**: The input-dependent weight generation enables richer representations than static FFN, justifying the additional complexity.
- **Channel Sparsity**: The band structure allows exploiting sparsity in the channel domain, complementing spatial sparsity.
- **Adaptive Sparsity Ratio**: The dynamic estimation of k adapts to input complexity, achieving better sparsity-error trade-offs than fixed ratios.

5 Experiments

5.1 Experimental Setup

Dataset and Evaluation. We conduct our experiments on the PANDA dataset, a large-scale benchmark for object detection in gigapixel-level images, featuring 18 scenes and over 15.9 million annotated bounding boxes. Following the official protocol, 13 scenes are used for training and 5 for testing. We adopt the standard COCO evaluation protocol, reporting Average Precision (AP) as the main metric, along with AP for small (AP_S), medium (AP_M), and large (AP_L) objects. To assess computational efficiency, we report the GFLOPs and parameter count of the backbone when processing a 1344×672 image patch.

μ	GFLOPs	AP	AP_S	AP_M	AP_L
∞	7.25	79.7	50.7	79.2	85.4
3	7.24	80.0	50.8	78.8	85.9
2	7.24	80.6	52.6	79.8	86.0
1	7.24	80.4	52.7	80.1	85.5
.5	7.24	80.1	50.3	79.9	85.8

Table 3: Ablation Study on the Hyperparameter μ. $\mu = \infty$ means all bands are preserved for local FFN. Since μ has no influence on the amount of parameters, this item is omitted.

Implementation Details. We implement all detectors using MMDetection toolbox [Chen *et al.*, 2019]. To ensure a fair comparison, all models are configured with identical architectural hyperparameters when paired with the same detection head. Thinformer is trained from scratch for 60 epochs in an end-to-end style. Unless otherwise specified, some of the crucial hyperparameter are as follows: we adopt the dynamic local FFN and dynamic sparsity ratio generator; a band consists of 8 channels and a window consists of 7x7 spatial positions; $\mu = 2, \lambda_k = .1$. The code is available in the supplementary materials.

5.2 Main Results

We benchmark Thinformer against state-of-the-art methods on the PANDA dataset, with results detailed in Table 1. Conventional close-up detectors like FasterRCNN and RetinaNet show limited performance, underscoring the unique challenges of gigapixel imagery.

To ensure a fair comparison of backbone architectures, we integrate Thinformer into the DINO detector framework. Our method achieves the performance of **80.6 AP.50**, surpassing all other backbones. Notably, compared to the highly efficient SparseFormer, Thinformer not only obtains performance improvement but also reduces computational cost from 33.84 GFLOPs to **7.24 GFLOPs** - a **78.6%** reduction. This demonstrates that Thinformer establishes a novel and significantly better trade-off between accuracy and efficiency for object detection in high-resolution wide imagery.

5.3 Ablation Study

Effectiveness of Dynamic Mechanisms. In Table 2, we analyze the core dynamic components of our model. We compare our full model (with *dynamic local FFN* and dynamic sparsity ratio k) against variants with fixed sparsity ratio or *static local FFN*. The results confirm that our dynamic mechanisms are crucial, achieving the best overall performance while maintaining low computational cost. Furthermore, even with static local FFN Thinformer still achieves non-trivial performance, which validates our conjecture about

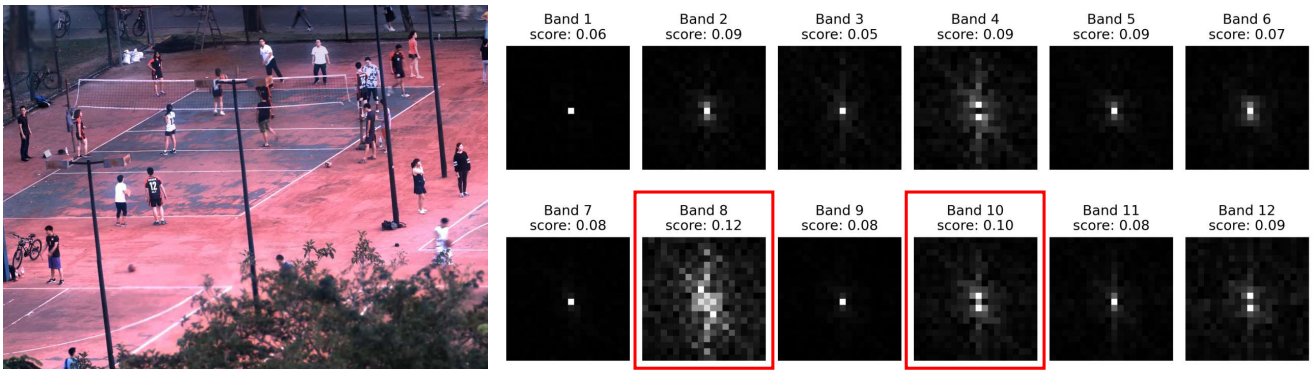

Figure 3: Visualization of the scores in the channel domain. We record the score allocated to each band in stage 1 and perform Fourier Transform on the average feature map within each band. Presented are the obtained amplitude spectra. The selected bands are highlighted using red bounding boxes. In this case, 2 bands are determined to be preserved for local FFN, according to the scores and the generated sparsity ratio. The leftmost picture stands for the original shot.

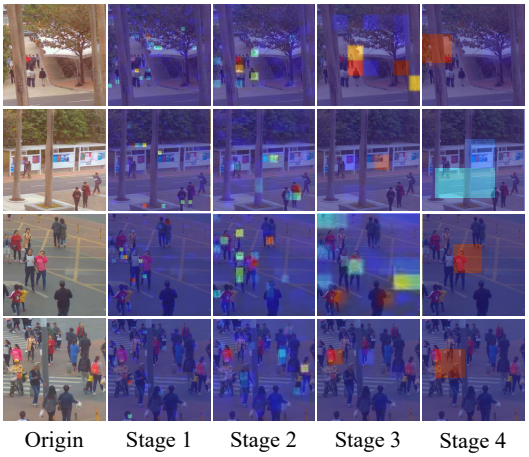

Figure 4: Visualization of the scores in the spatial domain in each stage of Thinformer. The leftmost column stands for the original shots and the following columns exhibit the scores of the feature map in each stage. Highlighted pixels represent higher scores.

the sparsity in the channel domain and our group-and-select method on channels.

Sparsity Ratio Distribution. Table 3 investigates the impact of the hyperparameter μ, which tunes the relationship between the sparsity ratios of the spatial and channel domains. Performance peaks at $\mu = 2$, with degradation for both higher and lower values. Notably, simply preserving all the bands (i.e., $\mu = \infty$) does not lead to improvement. This suggests that an optimal balance between spatial and frequency sparsification is key to success. Performance remains robust across a range of μ values, indicating our method is not overly sensitive to this hyperparameter.

Impact of Band Size. We explore the influence of band size on performance in Table 4. The results show that ThinFormer shows little sensitivity to this parameter, as performance remains stable across different configurations. A band size of 8 yields the best result and is adopted for all other experiments.

Band Size	GFLOPs	Param.	AP
16	7.25	37.13M	78.9
12	7.24	36.55M	79.4
8	7.24	36.34M	80.6
4	7.24	36.28M	79.7

Table 4: Ablation on Band Size. ThinFormer is insensitive to the number of channels per band.

5.4 Visulization

As shown in Fig. 4, we visualize the learned scores that guide spatial sparsification at each stage of the backbone. The model effectively learns to assign higher scores (highlighted regions) to areas containing objects, thereby preserving salient information while discarding background.

Fig. 3 offers a glimpse into the transform-domain sparsification. We visualize the amplitude spectra of the average feature map within different bands and highlight the bands selected by our model (red bounding boxes). The selected bands clearly correspond to spectra with richer structural information, whereas bands with dominating low-frequency coefficients or much redundant information are pruned. This confirms that Thinformer learns to identify and process the most informative frequency components to some extent, which is critical for its efficiency and effectiveness.

6 Conclusion

In this paper, we present ThinFormer, a dynamic and efficient object detection framework tailored for HRW shots. By jointly exploring sparsity in both spatial and channel domains, ThinFormer addresses the challenges of extreme object sparsity and variable scene complexity. The proposed local FFN leverages channel-domain sparsity through dynamic activation, while a self-supervised sparsity ratio estimator adapts to diverse image conditions. Extensive experiments demonstrate that ThinFormer achieves competitive accuracy with significantly reduced computational cost.

Acknowledgments

This work is supported by National Science and Technology Major Project (No. 2022ZD0119402), National Natural Science Foundation of China (No. 62572268, No. 82441013), "Pioneer" and "Leading Goose" R&D Program of Zhejiang (No. 2024C01142), Tsinghua University – Migu Xinkong Culture Technology (Xiamen) Co., Ltd. Joint Research Center for Intelligent Light Field and Interaction Technology, Tsinghua University - Joint R&D Project (R2511DKW) – Phase 2, and Ministry of Science and Technology of China (No. 2024YFB2809103). We also acknowledge the KLATASDS-MOE.

Contribution Statement

Wenxi Li and Kunpeng Liu made equal contribution to this work.

References

[Cai and Vasconcelos, 2018] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018.

[Carion *et al.*, 2020] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[Chen *et al.*, 2019] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Mmdetection: Open mmlab detection toolbox and benchmark, 2019.

[Chen *et al.*, 2022] Kai Chen, Zerun Wang, Xueyang Wang, Dahan Gong, Longlong Yu, Yuchen Guo, and Guiguang Ding. Towards real-time object detection in gigapixel-level video. *Neurocomputing*, 2022.

[Everingham *et al.*, 2010] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[Fan *et al.*, 2022] Jiahao Fan, Huabin Liu, Wenjie Yang, John See, Aixin Zhang, and Weiyao Lin. Speed up object detection on gigapixel-level images with patch arrangement. In *CVPR*, 2022.

[Ge *et al.*, 2021] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021.

[Girshick *et al.*, 2014] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[Girshick, 2015] Ross Girshick. Fast r-cnn. In *ICCV*, 2015.

[Han *et al.*, 2022] Qi Han, Zejia Fan, Qi Dai, Lei Sun, Ming-Ming Cheng, Jiaying Liu, and Jingdong Wang. On the connection between local attention and dynamic depth-wise convolution. In *ICLR*, 2022.

[Hassanieh *et al.*, 2012] Haitham Hassanieh, Piotr Indyk, Dina Katabi, and Eric Price. Simple and practical algorithm for sparse fourier transform. In *SODA*, 2012.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[He *et al.*, 2017] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

[Hoefler *et al.*, 2021] Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *JMLR*, 22(241):1–124, 2021.

[Lei *et al.*, 2025] Mengqi Lei, Siqi Li, Yihong Wu, Han Hu, You Zhou, Xinhu Zheng, Guiguang Ding, Shaoyi Du, Zongze Wu, and Yue Gao. Yolov13: Real-time object detection with hypergraph-enhanced adaptive visual perception. *arXiv preprint arXiv:2506.17733*, 2025.

[Li *et al.*, 2020] Changlin Li, Taojiannan Yang, Sijie Zhu, Chen Chen, and Shanyue Guan. Density map guided object detection in aerial images. In *CVPRW*, 2020.

[Li *et al.*, 2021] Changlin Li, Guangrun Wang, Bing Wang, Xiaodan Liang, Zhihui Li, and Xiaojun Chang. Dynamic slimmable network. In *CVPR*, 2021.

[Li *et al.*, 2022] Changlin Li, Guangrun Wang, Bing Wang, Xiaodan Liang, Zhihui Li, and Xiaojun Chang. Ds-net++: Dynamic weight slicing for efficient inference in cnns and vision transformers. *IEEE TPAMI*, 45(4):4430–4446, 2022.

[Li *et al.*, 2024a] Wenxi Li, Yuchen Guo, Jilai Zheng, Haozhe Lin, Chao Ma, Lu Fang, and Xiaokang Yang. Sparseformer: Detecting objects in HRW shots via sparse vision transformer. In *ACM MM*, 2024.

[Li *et al.*, 2024b] Wenxi Li, Ruxin Zhang, Haozhe Lin, Yuchen Guo, Chao Ma, and Xiaokang Yang. Saccadedet: A novel dual-stage architecture for rapid and accurate detection in gigapixel images. In *ECML-PKDD*, 2024.

[Li *et al.*, 2024c] Wenxi Li, Ruxin Zhang, Haozhe Lin, Yuchen Guo, Chao Ma, and Xiaokang Yang. Saccademot: Enhancing object detection and tracking in gigapixel images via scale-aware density estimation. In *ECAI*, 2024.

[Lin *et al.*, 2014] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[Lin *et al.*, 2017] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

[Liu *et al.*, 2016] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and

Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

[Liu *et al.*, 2021] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[Liu *et al.*, 2024] Chenglong Liu, Haoran Wei, Jinze Yang, Jintao Liu, Wenxi Li, Yuchen Guo, and Lu Fang. Gigahumandet: Exploring full-body detection on gigapixel-level images. In *AAAI*, 2024.

[Maddison *et al.*, 2017] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *ICLR*, 2017.

[Meng *et al.*, 2022] Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. Adavit: Adaptive vision transformers for efficient image recognition. In *CVPR*, 2022.

[Najibi *et al.*, 2019] Mahyar Najibi, Bharat Singh, and Larry S Davis. Autofocus: Efficient multi-scale inference. In *ICCV*, 2019.

[Newell *et al.*, 2016] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499. Springer, 2016.

[Nyquist, 1928] H. Nyquist. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers*, 47(2):617–644, 1928.

[Rao *et al.*, 2021] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *NeurIPS*, 2021.

[Redmon and Farhadi, 2018] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[Redmon *et al.*, 2016] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[Ren *et al.*, 2015] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[Song *et al.*, 2021] Lin Song, Songyang Zhang, Songtao Liu, Zeming Li, Xuming He, Hongbin Sun, Jian Sun, and Nanning Zheng. Dynamic grained encoder for vision transformers. In *NeurIPS*, 2021.

[Tian *et al.*, 2022] Zhi Tian, Xiangxiang Chu, Xiaoming Wang, Xiaolin Wei, and Chunhua Shen. Fully convolutional one-stage 3d object detection on lidar range images. In *NeurIPS*, 2022.

[Wang *et al.*, 2020] Xueyang Wang, Xiya Zhang, Yinheng Zhu, Yuchen Guo, Xiaoyun Yuan, Liuyu Xiang, Zerun Wang, Guiguang Ding, David Brady, Qionghai Dai, et al. Panda: A gigapixel-level human-centric video dataset. In *CVPR*, 2020.

[Wang *et al.*, 2021] Tao Wang, Li Yuan, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. Pnp-detr: Towards efficient visual analysis with transformers. In *ICCV*, 2021.

[Xia *et al.*, 2018] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. In *CVPR*, 2018.

[Yang *et al.*, 2019] Fan Yang, Heng Fan, Peng Chu, Erik Blasch, and Haibin Ling. Clustered object detection in aerial images. In *ICCV*, 2019.

[Zhang *et al.*, 2023] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. In *ICLR*, 2023.

[Zhou *et al.*, 2019] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.