

Patch-based Selection and Refinement for Early Object Detection

Tianyi Zhang
 University of Minnesota
 zhan9167@umn.edu

Kishore Kasichainula
 Arizona State University
 kkasicha@asu.edu

Yaoxin Zhuo
 Arizona State University
 yzhuo6@asu.edu

Baoxin Li
 Arizona State University
 baoxin.li@asu.edu

Jae-Sun Seo
 Cornell Tech
 jseo28@asu.edu

Yu Cao
 University of Minnesota
 yuca@umn.edu

Abstract

Early object detection (OD) is a crucial task for the safety of many dynamic systems. Current OD algorithms have limited success for small objects at a long distance. To improve the accuracy and efficiency of such a task, we propose a novel set of algorithms that divide the image into patches, select patches with objects at various scales, elaborate the details of a small object, and detect it as early as possible. Our approach is built upon a transformer-based network and integrates the diffusion model to improve the detection accuracy. As demonstrated on BDD100K, our algorithms enhance the mAP for small objects from 1.03 to 8.93, and reduce the data volume in computation by more than 77%.

1. Introduction

Object detection (OD) plays a vital role in numerous real-world applications, such as autonomous driving, and robotics. Despite the proliferation of diverse algorithms for this task, existing methods still face significant challenges in early object detection, a crucial aspect enabling prompt and proactive decision-making. In such scenarios, objects in captured images are often significantly reduced in size due to long distances. As illustrated in Fig. 1, when images contain only a limited number of objects, and the performance of object detection significantly deteriorates due to insufficient data volume.

To address this challenge, we can exploit super-resolution (SR) algorithms to reconstruct the higher-resolution images, thereby augmenting the data available for subsequent object detection models. SR is also a classic problem in computer vision, boasting a plethora of solutions tailored for this task. Recently, the diffusion models, such as DDPM [13], have showcased remarkable capabilities in image generation and demonstrated greater stability

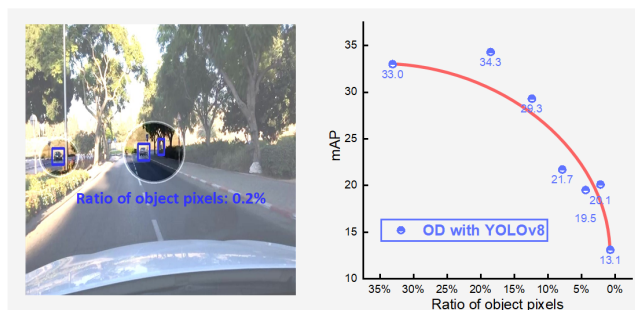


Figure 1. (Left): Objects occupy only a small proportion of the entire image in this example of BDD100K dataset. (Right): With object pixels decreasing, the OD performance rapidly drops.

compared to generative adversarial networks (GAN) [10]. Moreover, research focusing on the application of Conditional Diffusion Models (CDM) [14, 30], for SR has yielded notable advancements. Through the utilization of diffusion models for high-resolution image generation, we can achieve substantial enhancements in object detection performance, particularly for datasets with a low object-to-image ratio. However, diffusion models come with a significant computational cost, which poses a challenge for real-world applications like autonomous driving. From the image example in Fig. 1, the holistic refinement of the image results in a considerable computational burden on background pixels, leading to an excessive waste of resources that does not yield any meaningful contributions to OD.

In this paper, we introduce a novel algorithm, named Dichotomized Patch Refinement (DPR), to tackle the aforementioned problem. DPR leverages CDM to exclusively reconstruct patches that encompass objects, employing a Patch-Selector module for accurate patch classification. While the task of directly localizing small objects presents considerable challenges, discerning the presence or absence of objects within patches proves to be a more feasible ap-

proach. By leveraging the Patch-Selector module, we can efficiently filter out irrelevant patches that do not contribute to the subsequent OD task. This strategy significantly reduces the data volume, enabling the immediate generation of refined images using CDM to greatly enhance object detection accuracy. To facilitate the module’s implementation, we devise a hierarchical patch encoder inspired by the structure of the Swin Transformer [22] to extract embeddings for individual patches. Furthermore, we incorporate a patch classifier through the introduction of a classification token, following a similar approach to ViT [8]. Moreover, in line with our network’s hierarchical structure, we introduce a pyramid patch class label to ensure an ample inclusion of positive patches. Our experiments, conducted on the BDD100K dataset, provide compelling evidence of DPR’s efficacy and accuracy for early object detection.

To summarize, our key contributions are as follows:

- We design a Patch-Selector module, incorporating the attention mechanism, to effectively sift desired patches containing objects from images. Moreover, we introduce a hierarchical architecture and employ a pyramid loss function to further improve the selection process.
- By harnessing the capabilities of Conditional Diffusion Models (CDM), we effectively refine solely the selected patches, yielding enhanced performance in object detection.
- By enlarging negative patches with interpolation, we seamlessly combine all processed patches to form complete images. Through comprehensive experiments on both patches and entire images, we demonstrate that our DPR achieves competitive early object detection performance with 77.2% reduction of the computation.

2. Related work

2.1. Diffusion Models for Image SR

Initially, ConvNets gained prominence in image super-resolution [18, 23], particularly with the seminal work on the SRCNN model [7]. However, the introduction of generative adversarial networks (GAN) by Goodfellow *et al.* [10] revolutionized the field, offering unprecedented image generation capabilities. GAN-based SR methods [5, 16, 17, 19, 34], have since become prevalent. These techniques employ game theory-inspired competition between a generator and a discriminator to drive iterative improvements and generate high-quality images. Nonetheless, challenges related to training stability and model convergence persist in GAN-based SR methods.

Instead, diffusion models [31] have demonstrated superior performance in image generation and exhibit enhanced

stability. The introduction of DDPM by Jonathan *et al.* [13], has further popularized the use of diffusion models in the field of image generation, displacing the reliance on GANs. Additionally, recent research has focused on techniques for fast sampling [11, 15, 24, 25, 36, 37]. DDIM [32] accelerates the sampling process by $10\times$ to $50\times$ through the introduction of a more efficient class of implicit probabilistic models. Given the remarkable performance of diffusion models in image generation, several studies have explored their application in SR by leveraging CDM. For instance, Saharia *et al.* proposed SR3 [30], which demonstrates improved SR performance based on CDM. Similarly, Jonathan *et al.* introduced Cascaded Diffusion Models [14], which further advances the field of SR.

2.2. Object Detection (OD)

Traditional methods for OD, such as Faster RCNN [9], primarily rely on convolutional layers. The introduction of anchor boxes in Faster R-CNN, a two-stage OD algorithm, has significantly transformed conventional methodologies. Consequently, numerous convolution-based methods, such as YOLO [2, 26–28], Mask R-CNN [12], have emerged and continually improved performance in OD.

Furthermore, the attention mechanism [33], initially introduced in ViT [8] for image classification, has been widely adopted in various computer vision tasks, including OD. This is primarily due to the transformer’s ability to model long-range dependencies. Carion *et al.* proposed DETR [3], which formulates OD as a direct set prediction problem and employs a transformer encoder-decoder network. DINO [4], introduced by Caron *et al.*, leverages self-supervised learning to develop a new transformer network based on ViT. To reduce computation, Liu *et al.* proposed Swin Transformer [21, 22], which incorporates a novel window-based self-attention mechanism. Inspired by BERT [6] in natural language processing, Bao *et al.* presented BEiT [1] for computer vision applications.

3. Methodology

As illustrated in Fig. 2, DPR comprises three crucial modules: Patch-Selector, Patch-Refiner, and Patch-Organizer. The Patch-Selector module is responsible for extracting patch features and performing classification. Subsequently, the Patch-Refiner module elaborates on the positive patches, leveraging CDM to reconstruct them to a higher resolution, thereby enhancing object detection precision. Lastly, to completely show the efficiency and accuracy of our proposed method, we employ inexpensive interpolation techniques to enlarge the negative patches and organize all patches into entire images to facilitate a direct comparison with the original images. In this section, we provide a detailed discussion of all the modules, and outline the specific training procedures of DPR, which are presented in

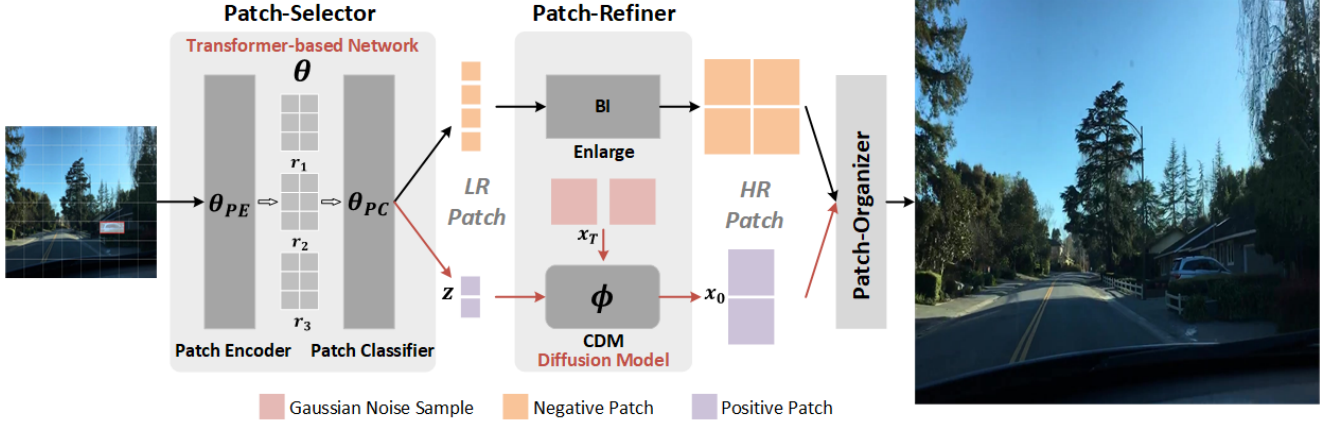


Figure 2. **Overall architecture of DPR (Dichotomized Patch Refinement)**. By dividing all patches of the original image into two groups based on whether it contains objects or not before the image reconstruction, we leverage CDM to process only positive patches to reduce computation and improve the performance for the subsequent OD task since negative patches don't contribute to OD. There are two major components for training: Patch-Selector module with learnable parameters θ , and CDM with parameters ϕ .

Algorithm 1. Additionally, Algorithm 2 elucidates the sampling and testing processes.

Algorithm 1 Training with DPR

- Input:** Image data $\mathbf{I}_{in} \in \mathbb{R}^{H_{in} \times W_{in} \times C_{in}}$, patch class labels $\mathbf{y}_1 \in \mathbb{R}^{\frac{H}{2} \times \frac{W}{2} \times 1}$, $\mathbf{y}_2 \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times 1}$, $\mathbf{y}_3 \in \mathbb{R}^{\frac{H}{8} \times \frac{W}{8} \times 1}$, and hyper-parameters $\alpha_{1:T}$
- 1: Randomly initialize Patch-Selector model parameters θ , and CDM model parameters ϕ
 - 2: **for** each epoch $t = 1, 2, \dots$ **do**
 - 3: $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3 = f_{\theta_{PE}}(\mathbf{I}_{in})$ \triangleright Patch encoding with TL
 - 4: $\mathbf{s}_i = \text{softmax}(f_{\theta_{PC}}(\mathbf{r}_i)) \quad \forall i \in 1, 2, 3$ \triangleright Patch classification
 - 5: $\mathcal{L}_P(\theta) = \sum_{i=1}^3 (-\mathbf{y}_i \log(\mathbf{s}_i) - \beta(1-\mathbf{y}_i) \log(1-\mathbf{s}_i))$
 \triangleright Pyramid loss
 - 6: $\theta \leftarrow \theta - \eta_{\theta} \nabla_{\theta} \mathcal{L}_P(\theta)$ \triangleright Update Patch-Selector model
 - 7: **end for**
 - 8: $\mathbf{s} = \max(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3)$ \triangleright Aggregation of predictions
 - 9: $\mathbf{z} = \mathbf{I}_{in} \times \mathbf{s}$ \triangleright Positive patch selection
 - 10: **repeat**
 - 11: $(\mathbf{z}, \mathbf{x}_0) \sim p(\mathbf{z}, \mathbf{x}_0)$ \triangleright Sample positive patch data
 - 12: $t \sim \text{Uniform}(\{1, \dots, T\})$
 - 13: $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
 - 14: $\phi \leftarrow \phi - \eta_{\phi} \nabla_{\phi} \|f_{\phi}(\mathbf{z}, \sqrt{\alpha_t} \mathbf{x}_0 + (1-\alpha_t)\epsilon, t) - \epsilon\|^2$
 \triangleright Update CDM model
 - 15: **until** converged

3.1. Patch-Selector

Network architecture. This module splits the image into 8×8 patches non-overlapping and classifies them to

determine if it contains objects or not. Specifically, as depicted in Fig. 3, the input image, $\mathbf{I}_{in} \in \mathbb{R}^{H_{in} \times W_{in} \times C_{in}}$ (H_{in} , W_{in} , and C_{in} are the input image height, width and the number of channels), undergoes a hierarchical patch encoder comprising multiple Transformer Layers (TL). This process generates patch representations at three different scales, namely $\mathbf{r}_1 \in \mathbb{R}^{\frac{H}{2} \times \frac{W}{2} \times 2C}$, $\mathbf{r}_2 \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times 4C}$, $\mathbf{r}_3 \in \mathbb{R}^{\frac{H}{8} \times \frac{W}{8} \times 8C}$, as the following equations,

$$\mathbf{r}_1 = TL_1(TL_0(EL(\mathbf{I}_{in}))) \quad (1)$$

$$\mathbf{r}_2 = TL_2(\mathbf{r}_1) \quad (2)$$

$$\mathbf{r}_3 = TL_3(\mathbf{r}_2) \quad (3)$$

where TL_i denotes the i th Transformer Layer, and EL is the embedding layer at the beginning of the network. H , W depend on the patch size.

Our TL is similar to the Swin Transformer structure, and it consists of three components: one feature merging layer for representation down-sampling, one window-based multi-head self-attention block (W-MSA), and another shifted window-based multi-head self-attention block (SW-MSA) to capture information across windows. Specifically, W-MSA splits the input feature into $n * n$ non-overlapping windows, where n depends on the window size and feature size, and captures global contextual information within each window. W-MSA solely considers connections within each window, potentially missing out on connections across windows. To address this limitation, SW-MSA shifts the feature by the half of window size before partitioning to enable the cross-window connections.

Specifically, the features \mathbf{r}_1 , \mathbf{r}_2 , and \mathbf{r}_3 correspond to patches of size $\frac{2H_{in}}{H} \times \frac{2W_{in}}{W}$, $\frac{4H_{in}}{H} \times \frac{4W_{in}}{W}$ and $\frac{8H_{in}}{H} \times \frac{8W_{in}}{W}$, respectively. To classify these patches, we compute the cross-attention with the learnable classification token,

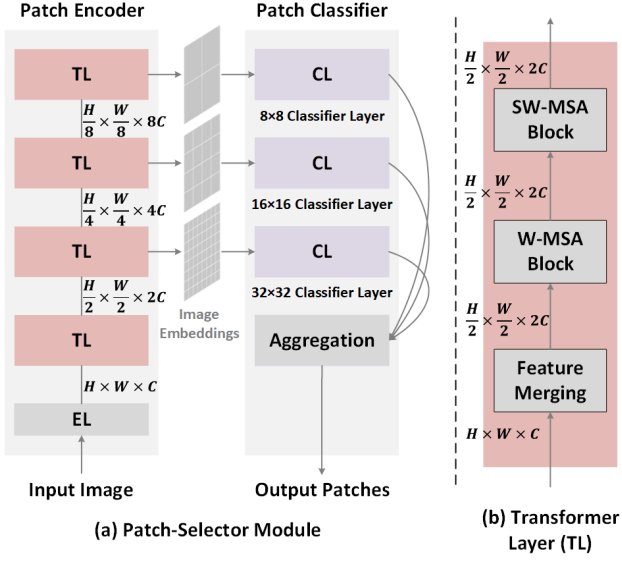


Figure 3. **The design of Patch-Selector Module.** (a) Utilizing a hierarchical architecture encoder, input images are embedded into features at three different scales. Subsequently, patches within these features undergo classification and aggregation to form the final output. (b) Each Transformer Layer (TL) includes a feature merging block and multiple window-based self-attention blocks.

denoted as c . The computation can be expressed as follows:

$$\mathbf{Q}_i = \mathbf{r}_i \mathbf{W}_i^q, \quad \mathbf{K}_i = \mathbf{c} \mathbf{W}_i^k, \quad \mathbf{V}_i = \mathbf{c} \mathbf{W}_i^v \quad \forall i \in 1, 2, 3 \quad (4)$$

$$\mathbf{A}_i = \text{softmax}\left(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{d}}\right) \mathbf{V}_i \quad \forall i \in 1, 2, 3 \quad (5)$$

where \mathbf{W}_i^q , \mathbf{W}_i^k , and \mathbf{W}_i^v are linear layer weights for query, key, and value matrices.

Next, the features are passed through a multi-layer perceptron (MLP) and a softmax layer to predict the class for each patch as follows,

$$\mathbf{s}_i = \text{softmax}(\text{MLP}_i(\mathbf{A}_i)) \quad \forall i \in 1, 2, 3 \quad (6)$$

where MLP_i denotes the output layer for the i th feature embeddings.

Accordingly to the network structure, we introduce a pyramid label that contains, $\mathbf{y}_1 \in \mathbb{R}^{\frac{H}{2} \times \frac{W}{2} \times 1}$, $\mathbf{y}_2 \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times 1}$, and $\mathbf{y}_3 \in \mathbb{R}^{\frac{H}{8} \times \frac{W}{8} \times 1}$, to supervise the training of Patch-Selector module by assigning positive labels to the patches that contain objects.

To minimize information loss, the final prediction is derived by selecting the maximum value from three scales after up-sampling to the same size with bilinear interpolation, ensuring the retention of a greater number of patches.

Algorithm 2 Sampling and Testing with DPR

Input: Image data $\mathbf{I}_{in} \in \mathbb{R}^{H_{in} \times W_{in} \times C_{in}}$, and hyper-parameters $\alpha_{1:T}$

- 1: $\mathbf{z} = f_{\theta}(\mathbf{I}_{in})$ \triangleright Partition to patches and classify
- 2: **for** each patch $\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^K$ **do**
- 3: **if** \mathbf{z} is positive **then**
- 4: Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
- 5: **for** $t = T, \dots, 1$ **do**
- 6: $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\epsilon_t = 0$
- 7: $\mathbf{x}_{t-1} \leftarrow \frac{1}{\sqrt{\alpha_t}}(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\alpha_t}} f_{\phi}(\mathbf{z}, \tilde{\mathbf{x}}_t, t)) + \beta_t \epsilon_t$ \triangleright Remove the noise iteratively
- 8: **end for**
- 9: **else**
- 10: $\mathbf{x}_0 = \text{Enlarge}(\mathbf{z})$ \triangleright Enlarge negative patches
- 11: **end if**
- 12: **end for**
- 13: Randomly initialize OD model parameters θ'
- 14: **for** each epoch $t = 1, 2, \dots$ **do**
- 15: Update model θ' with \mathbf{x}_0
- 16: **end for**
- 17: Output the prediction of object classes and bounding boxes with trained model, and evaluate mAP

Loss Function. The loss for each patch is computed using cross-entropy. To incorporate predictions from the hierarchical network at three scales and reduce false negative (FN) predictions, we introduce a combined loss formulation. This formulation involves the weighted sum of individual losses and can be expressed as follows

$$\mathcal{L}_P = \sum_{i=1}^3 (-\mathbf{y}_i \log(\mathbf{s}_i) - \beta(1 - \mathbf{y}_i) \log(1 - \mathbf{s}_i)) \quad (7)$$

where β is a hyper-parameter to adjust weight, and we set it to 0.01 in our experiments.

3.2. Patch-Refiner

Depending on the patch class, different refinement approaches are employed. For positive patches, the conditional diffusion models (CDM) reconstructs them with finer details. Conversely, negative patches are scaled up using simpler up-sampling methods, such as bilinear interpolation (BI), in the Enlarge module.

CDM. Diffusion Models consist of a forward process that progressively corrupts the input data over T timesteps by keeping adding Gaussian noise, and a reverse process to restore the original data from the final corrupted data. And for CDM, the reconstruction of the corrupted data is performed based on an additional signal that is related to the original data, such as a lower-resolution image in the context of super-resolution (SR).

Let $\mathbf{z} \in \mathbb{R}^{H_p \times W_p \times C_p}$ (H_p , W_p , C_p are the patch height, width, and the number of channels) denote the low-resolution patches we obtain from the Patch-Selector module while $\mathbf{x}_0 \in \mathbb{R}^{8H_p \times 8W_p \times C_p}$ is high-resolution data. Then, the forward process of our CDM is adding Gaussian noise to \mathbf{x}_0 over T steps as follows,

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I}) \quad (8)$$

$$q(\mathbf{x}_{1:T} | \mathbf{x}_0) = \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1}) \quad (9)$$

$$= \mathcal{N}(\mathbf{x}_T; \sqrt{\bar{\alpha}_T} \mathbf{x}_0, (1 - \bar{\alpha}_T) \mathbf{I}) \quad (10)$$

where $\alpha_{1:T}$, $\beta_{1:T}$ are hyper-parameters, subject to $0 < \alpha_t < 1$, $\alpha_t + \beta_t = 1$, and $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$. They determine the variance of the noise added at each iteration. And $\bar{\alpha}_t$ should be small enough, so that the final signal \mathbf{x}_T we acquire after the forward process is roughly also a standard Gaussian noise.

To gradually recover the original data from the final noise, the CDM model $f_\phi(\mathbf{z}, \tilde{\mathbf{x}}_t, t)$ is trained to predict the added noise in each step with the input of low-resolution image \mathbf{z} , noisy image $\tilde{\mathbf{x}}_t$, and t , where the noisy image at timestep t could be obtained from Eq. (10):

$$\tilde{\mathbf{x}}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + (1 - \bar{\alpha}_t) \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (11)$$

And for the reverse sampling process, the model recovers the high-resolution patch \mathbf{x}_0 from \mathbf{x}_T conditioned on \mathbf{z} with the following equations,

$$p_\phi(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{z}) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\phi(\mathbf{z}, \tilde{\mathbf{x}}_t, t), \sigma_t^2 \mathbf{I}) \quad (12)$$

We set the variance $\sigma_t^2 \mathbf{I}$ to β_t , and we could compute the mean with the estimated noise from CDM model as follows,

$$\mu_\phi(\mathbf{z}, \tilde{\mathbf{x}}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} f_\phi(\mathbf{z}, \tilde{\mathbf{x}}_t, t) \right) \quad (13)$$

Finally, the iterative elaboration process is done with the following equation:

$$\mathbf{x}_{t-1} \leftarrow \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} f_\phi(\mathbf{z}, \tilde{\mathbf{x}}_t, t) \right) + \beta_t \boldsymbol{\epsilon}_t \quad (14)$$

where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

Enlarge. In real-world applications, we discard all negative patches since they do not contribute to the subsequent object detection (OD) task. However, this approach can compromise the integrity of the dataset labels, which in turn affects the fairness of experimental comparisons. To ensure a fair evaluation and demonstrate the effectiveness of our approach, we perform scaling on these negative patches using bilinear interpolation (BI), nearest interpolation, or bicubic interpolation, thereby matching them to the same resolution as the positive patches.

3.3. Patch-Organizer

By leveraging this module, we combine all the refined positive and negative patches based on their original locations (i.e., the indices of the x-axis and y-axis in the output of Patch-Selector), resulting in the generation of entire images to provide further evidence of the advancements achieved by our DPR algorithm, accompanied by reduced computational requirements.

4. Experiments

4.1. Dataset and Training Details

As described in Sec. 1, we evenly partition the BDD100K dataset [35] based on the ratio of object pixels into several subsets to test OD, and we select a subset of small ratio to simulate the early detection scenario, where distant objects are typically smaller in size. Our algorithm primarily focuses on enhancing OD performance for the subset with the longest distance, which is named FBDD and consists of images with an object pixel ratio of less than 1.5%. And we select another subset named NBDD, which contains larger objects with a foreground pixel ratio ranging from 15% to 23%, for model fine-tuning. Both subsets contain around 4000 training images and about 1000 validation images with the original size of 1280×720 .

For Patch-Selector optimization, we resize all the images to be 1024×1024 before inputting them to the model. The first embedding layer utilizes a kernel and stride size of 16, with a channel number of 96. We set the learning rates to 0.001 for the convolution-based network and 0.00001 for the attention-based network. For each TL, the depth, window size, and attention head number are set to 2, 7, 3. To align with the hierarchical network structure, we introduce a pyramid label that encompasses three scales: 8×8 , 16×16 , and 32×32 . The patch selection results from the three different scales are then aggregated to a output resolution of 8×8 . Once this module gets optimized, input images are resized to be 128×128 or 64×64 for training.

We mainly train the CDM to upscale the 16×16 patches to 128×128 in 1000 timesteps for OD evaluation, although our results show that it also performs well for larger resolution reconstruction. The network architecture is based on U-Net [29], with parameters similar to SR3 [30]. We conduct OD testing using YOLOv8, a state-of-the-art OD algorithm. We experimented with two NVIDIA A6000 GPUs.

4.2. CDM for Patch Refinement

We perform an extensive evaluation of the CDM for patch refinement, comparing its performance against BI. In Tab. 1, we present the results for 4 different scales with BI or CDM. In all cases, the high-resolution output is 128×128 given various input low-resolution patches. We evaluate metrics such as Peak Signal-to-Noise Ratio (PSNR) and

Patch resolution	Scale	PSNR \uparrow		SSIM \uparrow		FID \downarrow		KID \downarrow		mAP \uparrow	
		BI	CDM	BI	CDM	BI	CDM	BI	CDM	BI	CDM
4×4	$\times 32$	18.06	21.64	0.7560	0.9045	388.70	16.32	0.4624	0.0111	1.50	10.30
8×8	$\times 16$	19.96	23.76	0.8390	0.9384	276.7	8.399	0.3155	0.0033	3.48	9.14
16×16	$\times 8$	22.33	24.85	0.9044	0.9518	161.7	8.120	0.1707	0.0032	5.89	12.00
32×32	$\times 4$	25.61	22.00	0.9557	0.9160	51.12	23.72	0.0440	0.0147	11.20	13.80

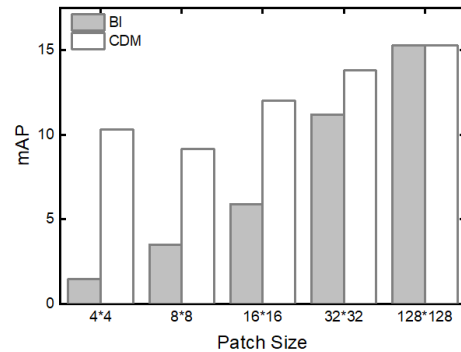
Table 1. **Results of patch refinement.** The patches generated by CDM could provide better features for image classification and OD.

Structural Similarity Index (SSIM), which measure the similarity between the generated patches and the ground-truth high-resolution patches. Additionally, we employ Fréchet Inception Distance (FID) and Kernel Inception Distance (KID) to compare the features extracted from these patches for image classification. Furthermore, we measure the mean Average Precision (mAP), which serves as the evaluation metric for our primary objective, OD. Generally, the patches generated by CDM outperform those from BI across all metrics. Despite that 32×32 patches from BI exhibit a higher similarity to the original patches, as indicated by the results of PSNR and SSIM, their features for image classification and OD are still inferior to those generated by CDM. The superiority of CDM is explicitly demonstrated with the mAP comparison in Fig. 4a.

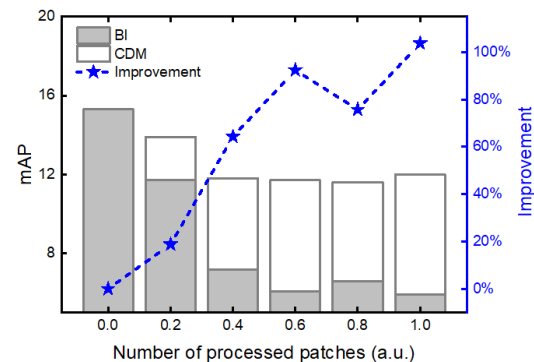
In addition, we create some new datasets for OD evaluation by gradually substituting the original high-resolution patches with the processed patches obtained from BI or CDM. As shown in Fig. 4b, the OD performance exhibits a more pronounced improvement with an increased proportion of processed patches from CDM. This observation underscores the notable advantages offered by CDM.

4.3. Patch Selection

Architecture of Patch-Classifier. While we have established the viability of CDM for SR, the challenge lies in accurately selecting patches containing objects. Achieving superior performance in the subsequent OD task requires careful consideration of the true positive rate (TPR) during the patch selection stage, as any irreversible information loss at this stage can severely degrade the detection performance. To address this, we utilize multiple transformer layers as the encoder to generate patch embeddings. And our primary focus is on the design of the Patch-Classifier module, which determines the presence of objects in each patch. The impact of the adopted techniques in the design of the Patch-Selector Module is presented in Table 2. Initially, we employed convolution layers (Conv-C), which yielded satisfactory results on NBDD dataset. However, by introducing a learnable token and utilizing cross-attention, we achieved even better performance (Attention-C). Moreover, by incorporating a hierarchical network structure and pyramid label (Attention-PC), we observed further improvements across



(a) OD results by refining patches of various scales.



(b) OD results by refining a different number of patches.

Figure 4. (a) CDM performs better for refining images. (b) More processed patches by CDM provide better performance.

all metrics, particularly in terms of TPR. Comparatively, convolution-based networks also benefited from the hierarchical structure and pyramid label (Conv-PC), but they were unable to match the performance of the attention-based.

Aggregation and pyramid loss. The results in the sixth row (Attention-AC) of Tab. 2 demonstrate that incorporating an aggregation block reduces information loss, as evidenced by the higher TPR. Furthermore, by modifying the loss function to place greater emphasis on positive patches, we observed further improvements in TPR, as shown in the seventh row (Attention-WC). With our final Patch-Selector architecture, we achieved a decent TPR for the FBDD dataset, as indicated in the last row of the table.

Method	TPR (Recall)	maxF	IoU
Conv-C	0.7539	0.7983	0.6350
Attention-C	0.8700	0.8521	0.7192
Conv-PC	0.8277	0.8600	0.7297
Attention-PC	0.9084	0.8855	0.7459
Attention-AC	0.9511	0.8809	0.6499
Attention-WC	0.9720	0.6946	0.6283
FBDD	0.9101	-	-

Table 2. After adopting all the techniques, our final architecture, Attention-WC, performs the best. The last row is the result of Attention-WC for FBDD dataset while others pertain to NBDD.

Model size. We explore different model sizes for the Patch-Selector module, specifically using 4, 5, or 6 transformer layers. In Tab. 3, utilizing a network with only 4 transformer layers can achieve equivalent performance in patch selection while reducing FLOPs to 5.01%.

Method	TPR	#Params	FLOPs(B)
Attention-PC/6	0.8962	1103.99M	121.47
Attention-PC/5	0.8917	336.31M	27.03
Attention-PC/4	0.8472	119.71M	6.09
Attention-AC/6	0.9537	1103.99M	121.47
Attention-AC/5	0.9459	336.31M	27.03
Attention-AC/4	0.9423	119.71M	6.09

Table 3. **Results for model size.** Four Transformer Layers achieve similar performance with much lower computation.

4.4. Comparison of OD Performance

To fully demonstrate the merit of our approach, we not only detect objects from patches with bounding box labels different from the original image due to patch partitioning for OD performance comparison, but we also integrate the entire images for detection. As we scale the 16×16 patches to 128×128 , we use the results obtained by directly feeding the 16×16 patches into the OD model as the baseline for patch-wise detection. We compare the performance of our DPR with this baseline as well as other methods. Additionally, since we have 8×8 patches, the entire image is scaled from 128×128 to 1024×1024 . Similarly, we use the results of the low-resolution 128×128 image as the baseline for image-wise detection.

Patch-wise detection. Besides our approach, we generate high-resolution patches with another two methods, BI and SR3 [30], for comparison. BI simply scales up all patches to 128×128 using bilinear interpolation while SR3 is a conditional diffusion model (CDM) based on DDPM that performs entire image super-resolution (SR). The PP

columns in Tab. 5 present the results when we feed only positive patches to OD, which is the approach we adopt in real-world applications. For BI and SR3, we assume that they can perfectly select positive patches (i.e., TPR is 1). The PP columns show our DPR performs the best.

To address the potential unfairness in the previous experiments, we also evaluate OD with both positive and negative patches, shown in the AP columns. As mentioned in Sec. 3, negative patches of DPR are enlarged with BI. Additionally, we conduct an experiment where all negative patches are replaced with black patches to simulate the removal of negative patches, denoted as DPR-B. DPR achieves comparable performance to SR3 with significantly fewer average refined patches of each image (14.59 on average versus 64). This highlights the computational efficiency of our approach. Interestingly, DPR-B outperforms DPR, suggesting that the selection results of our Patch-Selector module contribute to OD. By excluding the negative patches, which may introduce noise and confusion, our approach focuses solely on the positive patches, leading to improved detection results.

Image-wise detection. Figure 5 shows the visual comparison of BI and our DPR after integrating patches. While the overall generated images from DPR appear similar to BI, the crucial patches containing objects exhibit finer details, indicating that only a small amount of data need to be processed by CDM, leading to more efficient computation.

Quantitative results for OD are presented in Tab. 4. To compare with another SR method, SwinIR [20], and maintain consistency, we align our evaluation with SwinIR’s setting, upscaling images from 64×64 to 512×512 . We show the results of ground truth high-resolution images in the table as the upper bound. SR3 can perform much better than transformer-based SwinIR. DPR achieves the highest mAP among all the methods, resulting in an improved mAP from 0.194 to 4.33, while DPR can enhance mAP from 1.03 to 8.93 when upscaling images from 128×128 to 1024×1024 .

Efficiency of our approach. To trade off the computation and performance, we experiment with various thresholds for patch classification when upscaling images from 64×64 to 512×512 in Tab. 6. The second row, yielding

Image Size	Method	mAP	TPR	Precision
512×512	GT	7.48	0.106	0.309
	GT	0.194	0.017	0.009
512×512	BI	0.732	0.024	0.235
	SwinIR [20]	0.674	0.026	0.103
	SR3 [30]	2.38	0.061	0.423
	DPR(Ours)	4.33	0.078	0.457
1024×1024	DPR(Ours)	8.93	0.142	0.274

Table 4. The last row of 1024×1024 is upscaled from 128×128 . DPR performs the best for images-wise OD evaluation.

Patch Size	Method	# Patch	mAP		mAP ₅₀		TPR		Precision		FLOPs (B)
			PP	AP	PP	AP	PP	AP	PP	AP	
16 × 16	-	-	1.99	1.63	4.25	3.33	0.0493	0.0362	0.0818	0.0677	-
128 × 128	BI	100%	2.82	2.37	5.59	4.46	0.0618	0.0382	0.2430	0.2910	34.41
	SR3 [30]	100%	4.55	3.46	8.23	6.92	0.0843	0.0623	0.3170	0.3930	34.41
	DPR	22.8%	5.12	3.45	9.05	6.82	0.0886	0.0606	0.1930	0.3870	7.85
	DPR-B	-	-	3.54	-	7.82	-	0.0557	-	0.5200	-

Table 5. **Results of patch-wise OD.** PP denotes the experiments with only positive patches, and AP is tested for all patches. The third column (# Patch) shows the ratio of reconstructed patches for each image. mAP is the primary evaluation metric for OD. Our DPR obtains higher mAP with fewer patches refined.

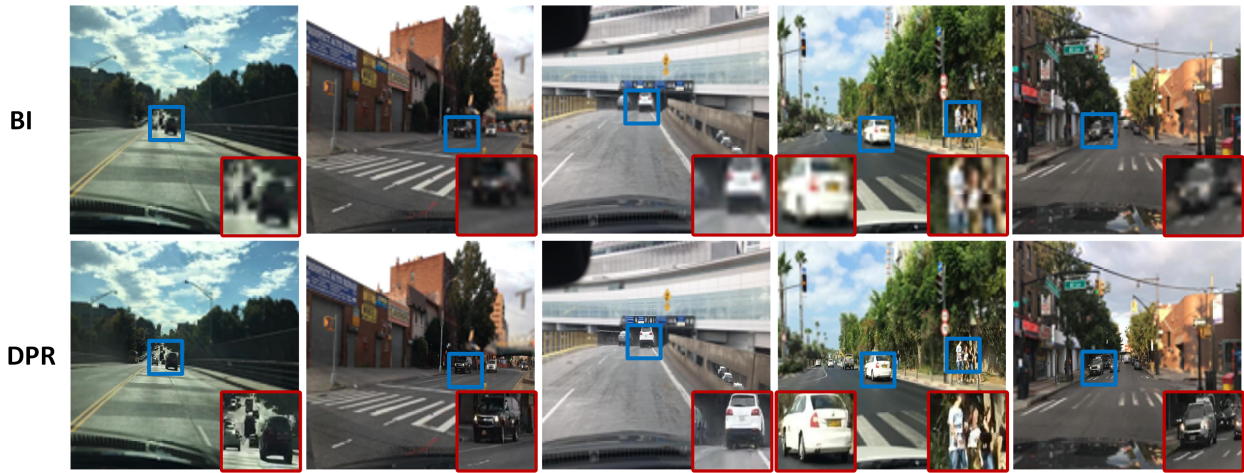


Figure 5. **The visual comparison of Bilinear interpolation and CDM.** The resolution of all images is 1024 × 1024 generated from 128 × 128 input when each patch of the images is scaled up from 16 × 16 to 128 × 128. Within the red boxes, the enlarged key patches from our DPR that contain objects exhibit finer details.

PS TPR	# Patch	mAP	mAP ₅₀	Precision
0.9813	71%	4.65	9.56	0.399
0.8972	37%	4.33	8.96	0.457
0.6290	11%	2.53	5.75	0.463
0.2120	3%	2.53	6.12	0.422

Table 6. By selecting different thresholds to assign patches, the second row achieves a comparable performance with 63% computation reduction. PS TPR of 0.8972 means about 90% positive patches are correctly selected by Patch-Selector (PS).

mAP of 4.33, stands out as the optimal choice, achieving 63% computation reduction.

For FBDD up-sampling from 128 × 128 to 1024 × 1024 with the same threshold, our PS module outputs only 22.8% patches for CDM generation and OD, and the FLOPs of PS are negligible compared to CDM, which means we save 77.2% computation compared to full-image generation, as demonstrated in Tab. 5.

5. Conclusion

In this paper, we propose a novel Dichotomized Patch Refinement algorithm (DPR) to efficiently enhance the OD performance by selectively reconstructing the high-resolution patches of images with conditional diffusion models. With a hierarchical transformer-based network and pyramid loss function, positive patches containing objects are accurately located. With patch-wise CDM, low-resolution positive patches are significantly refined, thereby improving the performance of the subsequent OD task. And the experimental results on the BDD100k dataset show that DPR effectively improves the mAP for early object detection from 1.03 to 8.93 with only 22.8% computation.

Acknowledgments

This work is supported by the Center for the Co-Design of Cognitive Systems (CoCoSys), one of seven centers in Joint University Microelectronics Program 2.0 (JUMP 2.0), a Semiconductor Research Corporation (SRC) program sponsored by the Defense Advanced Research Projects Agency (DARPA).

References

- [1] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 2
- [2] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 2
- [3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 213–229. Springer, 2020. 2
- [4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 2
- [5] Kelvin CK Chan, Xintao Wang, Xiangyu Xu, Jinwei Gu, and Chen Change Loy. Glean: Generative latent bank for large-factor image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14245–14254, 2021. 2
- [6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2
- [7] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part IV 13*, pages 184–199. Springer, 2014. 2
- [8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2
- [9] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 2
- [10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 1, 2
- [11] Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via min-snr weighting strategy. *arXiv preprint arXiv:2303.09556*, 2023. 2
- [12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2
- [13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 1, 2
- [14] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23(47):1–33, 2022. 1, 2
- [15] Rongjie Huang, Zhou Zhao, Huadai Liu, Jinglin Liu, Chenye Cui, and Yi Ren. Prodiff: Progressive fast diffusion model for high-quality text-to-speech. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2595–2605, 2022. 2
- [16] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 2
- [17] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 2
- [18] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016. 2
- [19] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017. 2
- [20] Jingyun Liang, Jiezhong Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1833–1844, 2021. 7
- [21] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022. 2
- [22] 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 *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 2
- [23] Xiaoqiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. *Advances in neural information processing systems*, 29, 2016. 2
- [24] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 2
- [25] Hao Phung, Quan Dao, and Anh Tran. Wavelet diffusion models are fast and scalable image generators. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10199–10208, 2023. 2
- [26] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2

- [27] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. [2](#)
- [28] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. [2](#)
- [29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015. [5](#)
- [30] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. [1](#), [2](#), [5](#), [7](#), [8](#)
- [31] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. [2](#)
- [32] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. [2](#)
- [33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. [2](#)
- [34] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV) workshops*, pages 0–0, 2018. [2](#)
- [35] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. [5](#)
- [36] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *arXiv preprint arXiv:2204.13902*, 2022. [2](#)
- [37] Hongkai Zheng, Weili Nie, Arash Vahdat, Kamyar Aziz-zadenesheli, and Anima Anandkumar. Fast sampling of diffusion models via operator learning. *arXiv preprint arXiv:2211.13449*, 2022. [2](#)