DiffVector: Boosting Diffusion Framework for Building Vector Extraction from Remote Sensing Images

Bingnan Yang†, Mi Zhang*, Yuanxin Zhao†, Zhili Zhang,
Xiangyun Hu, and Jianya Gong

***Abstract*—Building vector maps play an essential role in many remote sensing applications, thereby boosting the deep learning based automatic building vector extraction methods. These approaches have achieved pleasant overall accuracy, but their predict-style framework struggles with perceiving subtle details within a tiny area, such as corners and adjacent walls. In this study, we introduce a denoising diffusion framework called DiffVector to generate representations for direct building vector extraction from the remote sensing (RS) images. Firstly, we develop a hierarchical diffusion transformer (HiDiT) to conditionally generate robust representations for detecting nodes and extracting corresponding features. The conditions of HiDiT are multi-level boundary attentive maps encoded from input RS images through a topology-concentrated Swin Transformer (TC-Swin). Subsequently, an edge biased graph diffusion transformer (EGDiT) takes extracted node features as conditions to produce new visual descriptors for the adjacency matrix prediction. In EGDiT, we replace the standard self-attention operation with an edge biased attention (EBA) to inject edge information for training stabilization. Furthermore, given typical challenges of training difficulty and weak perceptive ability in convectional diffusion paradigms, we conduct an isomorphic training strategy (ITS), ensuring that the training procedures of both HiDiT and EGDiT precisely mirror the inference phase. Quantitative and qualitative experiments have evidently demonstrated that DiffVector can achieve competitive performance compared to existing modern approaches, especially in metrics assessing topology quality.**

***Index Terms*—Vector extraction, Building extraction, Diffusion model, Deep learning.**

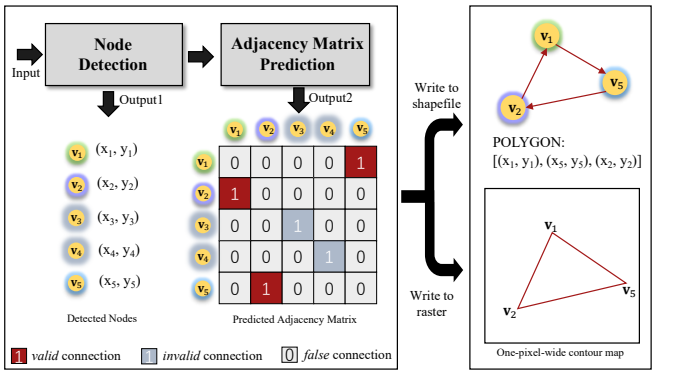
I. Introduction

BUILDING vector maps are structured as directional graphs recording footprints, connectivity and associated attributes, which supports distinct advantages of lossless scalability, convenient topological analysis, free attribute edition, and low storage cost. These characteristics establish the pivotal role of building polygon data in many remote sensing (RS) and geographic information systems (GIS) applications, such as population density estimation, disaster management and urban planning [1]–[3]. The brisk demand booms researches on automatic polygonal buildings extraction from remote sensing images in order to replace time-consuming and labor-intensive manual and semi-automatic production. Especially in recent years, deep learning (DL) methods offers promising solutions to automatic polygonal building map extraction and they can be grouped into three paradigms, namely segmentation-based, counter-based and node-based.

Segmentation-based approaches follow the two-step 'segmentation-vectorization' pipeline that firstly obtain segmentation results and then vecterize them to building vector polygons (i.e. building contours). Owing to the powerful DL-based segmentation models, segmentation-based building extraction can achieve stable raster maps. However, the inaccuracy at edges, such as omission and jaggies, is unavoidable due to the grid-based data form [4], [5]. Consequently, in order to achieve pleasant building polygons, this kind of methods are heavily reliant on optimization modules which are usually computationally intensive, expert-dependent and highly specialized. Unlike the former categories, contour-based methods directly extract building vector polygons from RS images without post-processing procedures via refining initialized building contours [6]–[8]. These approaches show promise in low-complex buildings while encounter failure in hollows, concave or complicated contours which limit their versatility. In contrast, the emerging node-based scheme is theoretically closest to the essence of vector data format. Approaches falling in this scope, such as PolyWorld [9] and TopDiG [10], directly extract building graphs from RS images without pre- or post-processing. As illustrated in Fig. 1, their typical pipeline is to decompose the vector extraction task into two sub-tasks, namely node extraction and adjacency matrix generation. The common practice of the former one is a dense prediction task aiming to generate heatmaps recording node probability, while the latter one is a sparse prediction task that leverages feature of detected nodes to predict adjacency matrix.

Although the node-based paradigm theoretically possesses the capability to handle building outlines of arbitrary complexity, in practical applications, existing methodologies frequently encounter difficulties in accurately capturing finer details. For example, they may overlook corner detection or fail to differentiate between closely situated buildings. This is exemplified in Fig. 2, where the imprecise focus on concave corners and adjacent building walls has led to conspicuous inaccuracies

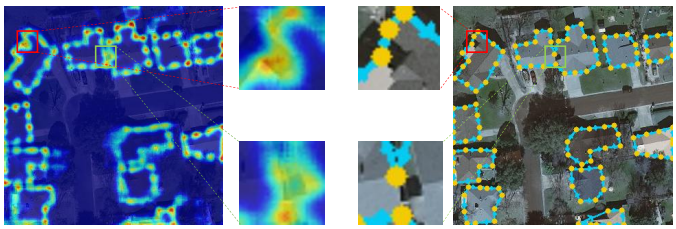
†: Equally contribute to this work, *Corresponding author: Mi Zhang
All authors are with School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China. (e-mail: bingnan.yang@whu.edu.cn; mizhang@whu.edu.cn; yuanxin.zhao@whu.edu.cn; zhangzhili@whu.edu.cn; huxy@whu.edu.cn; gongjy@whu.edu.cn)
Jianya Gong is also with State Key Laboratory of Information Enginnering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China.
Mi Zhang, Xiangyun Hu and Jianya Gong are also with Hubei Luojia Laboratory, Wuhan 430079, China.



(a) Raw Outputs (b) Supported Save Formats

Figure 1. The schematic diagram of the typical pipeline of node-based vector extraction approaches.

in the extraction results within those regions. Although these issues may not substantially affect the overall accuracy, they significantly hinder the practical applicability of the extraction outcomes. A potential root cause lies in the fact that current node-based building vector extraction techniques can only provide a coarse representation of these challenging areas within very limited windows, resulting in blurred features and inaccurate vector outputs.



(a) Attentive Maps (b) Vector Reuslts Visualization

Figure 2. The attentive map, building vector results, and detailed views of two typical problems in TopDiG extraction. The red and green colors indicate the missing corners and tangled adjacent walls, respectively.

As to this issue, the emerging diffusion model (DM) [11] is a theoretically potential solution to generate, instead of predict, better representations for the building vector extraction task. On the one hand, in terms of the node detection sub-task, the DM-based framework has reported its superiority over mainstream predictive scheme in similar dense prediction tasks, such as crisp edge prediction [12] and the single-node heatmap prediction for pose estimation task [13]. Therefore, it is reasonable to expect that the DM paradigm can also improve the performance of the node detection sub-task in node-based vector extraction methods. On the other hand, as to adjacency matrix prediction sub-task, diffusion frameworks have also been proved to be qualified to sparse predictions tasks like graph learning [14], [15]. However, these pioneering works all exhibit significant domain differences from the task of building vector extraction, rendering direct application impractical. In addition, there is currently limited exploration of combining diffusion model architectures to simultaneously accomplish dense and sparse prediction tasks. Therefore, the objective of this paper is to explore feasible solutions for boosting the diffusion like framework to the task of building vector extraction, in order to fully leverage the advantages of this architecture to achieve finer details in extracted results.

To achieve that, several challenges need to be solved:

To begin with, the widely adopted U-Net architecture in current diffusion models may be suboptimal to tackle both node detection and adjacency matrix prediction. The former task, as dense prediction, is conventionally solved by a conditional diffusion model (CDM) scheme which usually involves the information fusion operation among different modalities, such as image, timestep, and ground truth. In the multi-modal realm, transformer-based architectures have been proven superior over convolutional neural network (CNN) based structures, owing to their implicit advantages in cross-attention mechanisms. In terms of the adjacency matrix prediction task, the downsampling and upsampling mechanisms in U-Net unavoidably lead to information loss, which is unacceptable for the sparse graph like adjacency matrix. As shown in Fig. 3, corrupting only a tiny fraction of an adjacency matrix will severely and irreversibly damages the building topology graphs. Furthermore, priors like PolyWorld and TopDiG

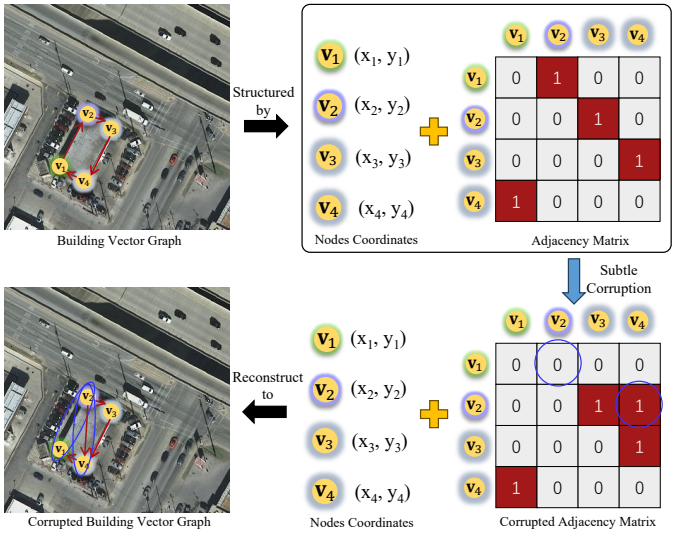


Figure 3. The organization structure of a building vector graph and the impact of the subtle corruption on the adjacency matrix. Red indicates directed connection between two nodes; Blue circles mark the corrupted connection (i.e. edge).

have demonstrated that the long-term message propagation of transformer architectures is crucial to stabilize adjacency matrix generation. Therefore, we adjust the emerging diffusion transformer (DiT) [16] scheme to conduct diffusion process in the latent feature space and generate representations for a sequential task-specific decoder.

Concretely, for the node detection, we introduce a hierarchical diffusion transformer (HiDiT) to generate robust representations for the sequential light weighting node decoder, consequently producing node heatmap (see Section III-B1). HiDiT utilizes multi-level boundary attentive maps, which are yielded by a topology-concentrated Swin Transformer (TC-Swin), as condition to generate representations for lightweight

node heatmap decoder. As to the graph generation task, we establish an edge biased graph diffusion transformer (EGDiT) to implement message passing among all extracted nodes (see Section III-B2). In EGDiT, node features are adopted to yield the edge embedding to incorporate edge-level information in the self-attention mechanism, thus enabling better representations generation for the following adjacency graph prediction.

Another challenge lies on the training difficulty and weak ability of regular diffusion paradigms, i.e. DDPM and DDIM, in the discriminative representation generation. The DDPMs suffer from low-efficiency and inevitable uncertainty caused by its stochastic and massive denoising steps. Targeting these issues, denoising diffusion implicit models (DDIM) [17] remove the Markov chain restriction to allow skip-step sampling and faster definite generation (see Section III-A). However, the common practice of DDIM executes the 'diffuse-denoise' procedure once with a randomly sampled step and is supervised by minimizing discrepancies between predicted noise and the original one during training. During inference, the DDIM scheme gradually denoises Gaussian noise to wanted outputs for multiple steps. Although theoretically equivalent, some prior works suggest that this scheme is easy to encounter overfitting problem in practice due to the difference between data distributions of training and inference [18], [19]. Moreover, supervising the model training directly by the specified perceive results may obtain better discriminative representations than vanilla noise supervision [12], [18]. Upon aforementioned insights, we introduce an isomorphic training strategy (ITS) that makes the training phase mirror the reasoning phase (see Section III-C). That is to say, the training process starts from the random Gaussian noise and is supervised by comparing the prediction and GT of the desired final outcomes, akin to the procedures in the inference phase.

In conclusion, in this study, we propose a transformer-based latent diffusion model, named DiffVector, to directly extract buildings vector graphs from remote sensing images. DiffVector sequentially predicts nodes and their adjacency matrix to extract vectorized buildings from RS images. Specifically, for node detection task, we design hierarchical diffusion transformer (HiDiT) that generates representations for a lightweight node detection decoder, conditioning multi-level boundary attentive maps encoded from input RS images. In terms of adjacency matrix prediction, we introduce edge biased graph diffusion transformer (EGDiT) to produce latent node descriptors, which are biased by edge information and conditioned by representations from HiDiT. By using the 'DiT+decoder' scheme, we enable the diffusion based model for both dense and sparse tasks and ultimately allow the node-based building vector extraction. In addition, to alleviate the uncertainty and overfitting issues, we implement the isomorphic training strategy (ITS) for both node detection and graph generation stages. Based on ITS, the training stages strictly mirror inference phases that denoise random Gaussian noise to target results of each subtask. Our contributions are concluded as follows:

(1) A denoising diffusion framework DiffVector is proposed to directly extract building vector graphs from remote sensing images. DiffVector deploys a 'DiT+decoder' scheme to detect node positions and predict their adjacency matrix. To our best knowledge, this is the first work applying diffusion models to vector extraction tasks.
(2) A hierarchical diffusion transformer (HiDiT) is designed to produce robust representations for detecting nodes and mining features, conditioning multi-level boundary attentive maps. An edge biased graph diffusion transformer (EGDiT) incorporates edge features to generate reliable visual descriptors for stable adjacency matrix prediction.
(3) An isomorphic training strategy (ITS) is introduced to directly denoise random Gaussian noise to desired results and is supervised during the training phase. ITS makes training phases of HiDiT and EGDiT mirror their inference process to facilitate the training and enhancing the capability of generating perceptive representations.
(4) DiffVector can achieve competitive performance compared to segmentation-based, contour-based, and previous node-based methods, especially with regard to topology quality.

II. Related Work

A. Deep learning based building vector extraction

In recent years, research on automatic building vector extraction methods based on deep learning has flourished. The current main technical approaches can be categorized into the three types, namely segmentation-based, contour-based and node-based.

Segmentation-based approaches are mainstream for the building vector extraction task and typically follow a two-step workflow of "segmentation-vectorization". Specifically, the raster prediction probability map produced by the image segmentation network is vectorized to obtain the final vector topological structure of buildings. Most studies adopt CNN frameworks like fully convolutional network (FCN) [20], U-Net [21] and ResNet [22]. For example, [23] modified U-Net by fusing the global representation of the first encoder layer with each output of other encoder stages to obtain multi-scale representations. [24] added residual attention at the end of each encoder layer and incorporated attention gate modules to refine the skip connections between multi-level encoder and decoder stages, improving the aggregation of multi-scale features. MSLANet [25] designs location channel attention to process representations of stage 1,2,4 in ResNet and then concatenates them with proposed multi-scale fusion module. Since the emergence of Transformer architecture [26], some researchers have investigated its talents in the building vector extraction task. [27]–[30] employed Swin Transformer [31] as the encoder to obtain multi-scale features for better building segmentation results. [32] adopted CNN-based encoder and Transformer-based decoder to achieve efficient building segmentation. [33] designed dual-path architecture to fully integrate the advantages of both CNN and Vision Transformer (ViT) [34].

Though aforementioned methods can obtain merit performance in the building segmentation, the raster results still struggle with edge jaggedness, over-smooth corners, fragmentation, necessitating the optimization modules for refined

results. [6] employed the Douglas-Peucker algorithm [35] to simply and regularize the vectorized building contours. [4] used directional field constraints to optimize building contours; [5], [36]–[39] leveraged boundary information to improve the accuracy and regularity of building segmentation results, thus obtaining better vectorized buildings. Generally speaking, segmentation-based methods are highly dependent on the accuracy of segmentation results, and optimization procedures are unavoidable for pleasant vector results at the cost of significantly increase in the computational demand and model complexity.

Contour-based methods can directly extract the building vector from RS images without post-processing modules. The idea is to first obtain an initial contour via image segmentation or object detection, and then optimize the coordinates of sampled contour nodes using methods such as circular convolution or graph convolutional networks (GCNs) to obtain the final building vector. These methods are highly dependent on the accuracy of the initial contour. [40] used rectangular object detection bounding boxes of building instances as initial contours and then refined them via a GCN to obtain building polygons from aerial images. In [41], building contours were initialized by preset closed circles and then refined through a novel cognitive graph convolution model. Unlike the above methods that use initial contours with fixed templates, BuildMapper [8] constructs adaptive initial contours for each building instance, further improving extraction accuracy. These methods can achieve end-to-end processing, do not require elaborate post-processing steps and consume less computational resources. The disadvantages are that they are only applicable to the extraction of low-complex objects and are difficult to tackle buildings with hollows, concave edges and other complicated outlines. Furthermore, the accuracy of contour initialization stage greatly and irreversibly impacts the final performance.

Node-based paradigms predict building contour nodes and their topological connectivity to directly extract building vector from RS images. [42] designed two parallel heads to simultaneously predict building corners and direction maps recording orientation angles of each edge, eventually realizing building vector extraction. [43] predicted both corners and connecting nodes, as well as both forward and backward direction of each node, achieving better building vector results. Some other works [44]–[46] utilized recurrent neural network (RNN) to literately track detected nodes from a start vertex to construct the vector graph of each building instance. Recently, PolyWorld [9] and TopDiG [10] propose to predict contour nodes and their adjacency matrix to construct final building vector results. Regardless of the varying topology construction approaches adopted in current node-based building vector extraction methods, the robustness of features is directly related to the overall performance of the entire model. The predict-style networks used in existing methods are all limited by a minimum receptive field, making it difficult to capture key features in extremely small regions. In this case, generative style based on diffusion models, which do not rely on the receptive field, may help alleviate this issue.

B. Diffusion models

Diffusion models have emerged as a powerful class of generative models in the field of deep learning, particularly for image synthesis tasks [47]–[50]. The foundational work in diffusion models can be traced back to the early 2010s with the introduction of the Denoising Score Matching (DSM) framework [51]. Subsequent advancements were marked by the introduction of the Denoising Diffusion Probabilistic Model (DDPM) [11], which leverages a Markov chain to simulate the data generation process. Each iteration within this chain encompasses a denoising network tasked with the removal of noise, thereby incrementally refining the data towards a coherent image representation. Despite the high fidelity of the generated images, DDPMs are often encumbered by their protracted sampling procedures. To address aforementioned issue of slow sampling issue in DDPM, the Denoising Diffusion Implicit Model (DDIM) [17] introduced a way by modifying the noise schedule to allow for faster convergence, thus speeding up the sampling process. Building upon these advancements, the Latent Diffusion Model (LDM) [52] was introduced, which operates the diffusion process within a lower-dimensional latent space. This approach not only enables more rapid sampling but also enhances the quality of the samples when compared to DDPMs. Furthermore, the LDM architecture facilitates greater control over the generation process. The Diffusion Transformer (DiT) [16] represents a significant departure from traditional diffusion models by employing transformer-based architectures. This innovation replaces the conventional neural networks with transformers, which are adept at capturing long-range dependencies within images, thus enhancing the model's generative prowess. Beyond the domain of image generation, diffusion models have been successfully applied to a myriad of generative tasks, including but not limited to image-to-image translation [53]–[56], super-resolution [57]–[59], cloud removal [60], [61] and so on.

C. Diffusion models for discriminative tasks

Discriminative tasks, unlike generative tasks, typically involve semantic perception, understanding and interpretation. Their outputs are a form of image patterns or discrete values that humans can comprehend. In the past several years, there have been a few of works introducing diffusion-like frameworks to discriminative tasks, which can be roughly divided into dense and sparse prediction tasks.

DM-based dense prediction tasks aim to generate a value for each pixel of input images. A few of priors have demonstrated the capability of the diffusion models in the zero-shot transfer segmentation [62], panoptic segmentation [63], [64], open-vocabulary segmentation [65]. medical image segmentation [66], [67] and RS image segmentation [19], [68]. Other applications of DM frameworks in dense prediction tasks include monocular depth estimation [18], [69], [70] and RS change detection [71], [72]. Though these works achieve competitive overall performance compared to conventional dense prediction models, they tend to overlook the importance of edge details. DiffusionEdge [12] leveraged latent denoise U-Net to generate robust representations for edge detection task,

obtaining impressive improvement of edge crispness. DiffPose [13] proposed a node diffusion architecture to conduct conditional single-node heatmap generation and achieved the state-of-the-art(SOTA) performance in the video human pose estimation. These two works illustrate the ability of DMs in delineating object boundaries and detecting nodes, respectively, which imply the feasibility of using DM framework to detect contour nodes for building vector extraction.

DM-based sparse prediction tasks target discrete data and are mainly explored in the graph learning field. Early DM-based graph learning model EDP-GNN [73] infused continuous Gaussian noise to adjacency matrices. [74] further modelled node and edge attributes by a novel stochastic differential equations. Afterwards, [75] found the discrete noise was more beneficial for the diffusion forward process of graph structured data than previous continuous Gaussian perturbations. Based on this observation, DiGress [14] advanced discrete diffusion process by progressively adding/removing edges or altering node attributes, noticeably improving the model performance. More recently, LGD [15] conducts diffusion processes by adding continuous noise to encoded graphs features in the latent space, enabling the capability in graph tasks of various types and levels.

According to aforementioned works, the DM framework, especially Latent Diffusion Model (LDM) paradigm has showcased impressive advantages and potentials in the discriminative tasks, including both dense and sparse prediction tasks. However, it is rarely explored in RS domain and few works investigate the collaborative utilization of different LDMs for both dense and sparse tasks. Therefore, this paper introduces DiffVector which adjusts the advanced Transformer-based LDM, namely Diffusion Transformer (DiT) to implement the building vector extraction task.

III. Methodology

A. Preliminaries

Denosing diffusion probabilistic model (DDPM). DDPMs define a Markovian chain process by gradually adding noise to sample data:

$$q\left(\boldsymbol{z}_t \mid \boldsymbol{z}_0\right)=\mathcal{N}\left(\boldsymbol{z}_t \mid \sqrt{\bar{\alpha}_t} \boldsymbol{z}_0,\left(1-\bar{\alpha}_t\right) \boldsymbol{I}\right), \tag{1}$$

which transforms data sample z_0 to a latent noisy sample z_t for $t \in \{0, 1, \ldots, T\}$ by adding noise to z_0. The constants $\bar{\alpha}_t := \sum_{t=0}^{t} \alpha_t = \sum_{t=0}^{T}(1-\beta_t)$ are hyper-parameters and β_t represents the noise variance schedule. By applying the reparameterization trick, we can sample $\boldsymbol{z}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{z}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$, where $\epsilon_t \sim \mathcal{N}(0, \boldsymbol{I})$. During the training process, learning a neural network $f_\theta(\boldsymbol{z}_t, t)$ to predict $\boldsymbol{z}_0$ from $\boldsymbol{z}_t$ under the guidance of condition x. At the inference stage, predicted data sample $\boldsymbol{z}_0$ is reconstructed from a random noise $\boldsymbol{z}_t$ through f_θ. Sampling in DDPM is stochastic, so even with the same initial noise, the prediction is uncertain.

Denoising diffusion implicit models (DDIM). In order to accelerate the efficiency of the DDPMs, the Non-Markovization method DDIM, is proposed [17]. Unlike DDPMs, there is no noise added in the reverse process, making it deterministic. Therefore, given an initial random noise, sampling through DDIM will always yield the same result, regardless of the number of sampling steps.

In this paper, we design two latent diffusion transformer-based networks, namely HiDiT and EGDiT, to generate representations for node detection and adjancency matrix prediction tasks, respectively. In our setting, instead of inconsistent training and inference procedures in traditional DDIMs, we conduct the ITS to establish a unified process for both phases.

B. Architecture

DiffVector adopts modified diffusion transformer architectures to directly extract building vector graphs from remote sensing imagery. As illustrated in Fig. 4. DiffVector firstly introduces a hierarchical diffusion transformer (HiDiT) to generate representations for extraction of potential contour nodes and corresponding visual feature descriptors (see Section III-B1). It is conditioned by multi-level boundary maps yielded by a topology-concentrated Swin Transformer (TCSwin) (see Section III-B1). Afterwards, an edge-biased graph diffusion transformer (EGDiT) is designed to produce reliable representations for the prediction of the adjacency matrix (see Section III-B2). To improve training stability, both HiDiT and EGDiT conduct the proposed isomorphic training strategy (ITS) that executes the procedure strictly same with the inference phase (see Section III-C).

1) Hierarchical diffusion transformer: Unlike U-Net architecture adopted in previous works [19], [71], the introduced hierarchical diffusion transformer (HiDiT) leverages the cross-attention ability of the transformer architecture to conduct conditional the denoising process. It is mainly composed of a topology-concentrated Swin Transformer (TCSwin), stacked HiDiT blocks and a lightweight node decoder. The proposed TCSwin encodes input RS image to capture compact and multi-level image features concentrated on building boundaries. These image features are received by DiT blocks as conditions to yield more robust representation for the ultimate node decoder. Details of each part are as follows:

TCSwin. We introduce a topology-concentrated Swin Transformer as the image encoder to extract multi-level boundary attentive maps as the condition for following HiDiT blocks (see Fig. 5). The standard Swin Transformer consists of four stages and receives an input image $I \in \mathcal{R}^{3\times H\times W}$. The shapes of representation maps produced by these four stages are $128 \times \frac{H}{4} \times \frac{W}{4}$, $256 \times \frac{H}{8} \times \frac{W}{8}$,$512 \times \frac{H}{16} \times \frac{W}{16}$,$1024 \times \frac{H}{32} \times \frac{W}{32}$, respectively. To capture compact perception of building boundaries, we establish four boundary blocks to predict building boundary masks from representation maps of each stage is modified. By using stacked transposed convolution operations, each of the boundary blocks yields a $1 \times H \times W$ boundary prediction map. Subsequently, all resultant maps are concatenated into the boundary attentive maps $F_s^{4\times H\times W}$ which provides multi-level perception on the building boundary area.

HiDiT blocks. As shown in Fig. 6 (a), HiDiT blocks input random Gaussian noise and additional conditions, i.e. timesteps, concatenated boundary attentive maps $F_s^{4\times H\times W}$, and output generated representations$F_g^{(\frac{H}{P}\times\frac{W}{P})\times D_1}$ where D_1 is the dimension and P is the patch size. F_s and t are

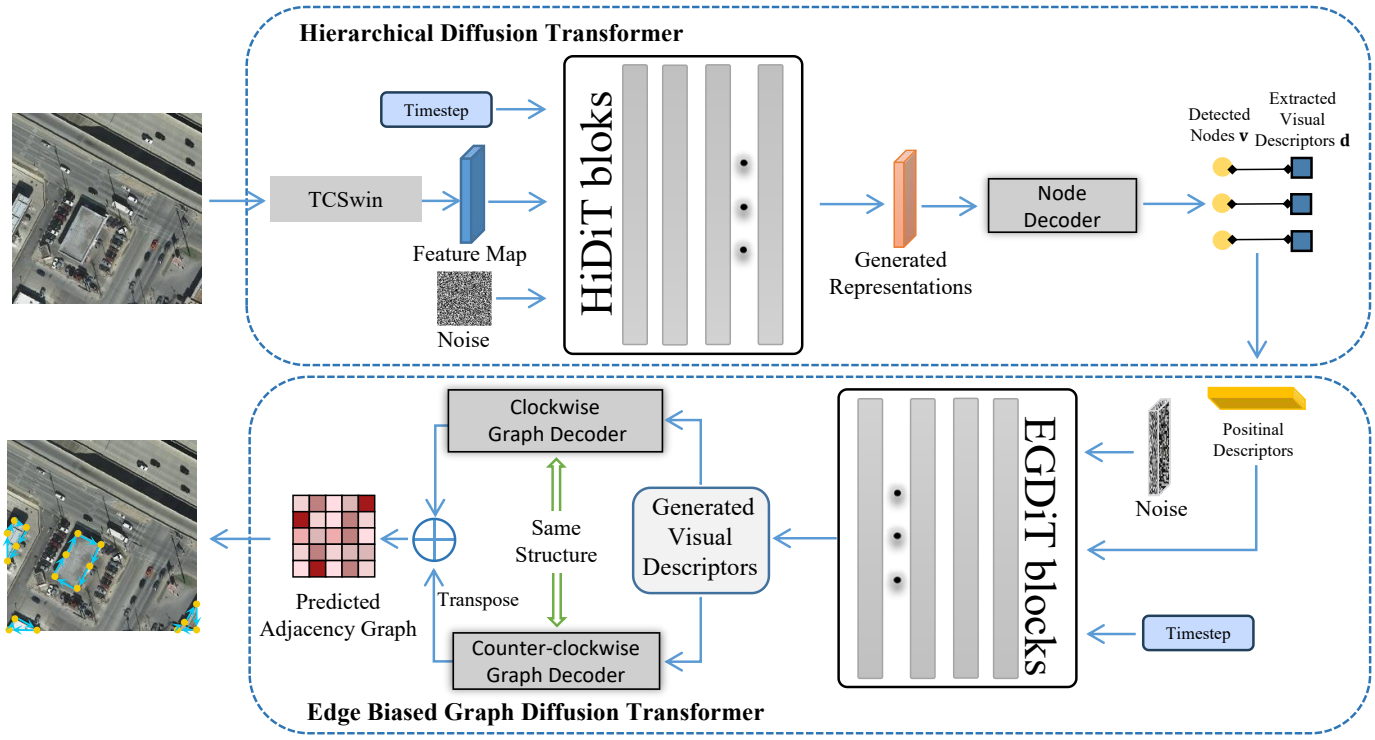


Figure 4. The overview structure of DiffVector. Diffvector consists of two sub-tasks: node extraction and adjacency matrix generation. We design corresponding diffusion models for each stream, named hierarchical diffusion transformer (HiDiT) and edge biased graph diffusion transformer (EGDiT). HiDiT conditions multi-level boundary attentive maps to generate representations for extracting key nodes and their corresponding visual descriptors, while EGDiT conditions node features to yield visual descriptors for the predictions of the adjacency matrix. In the results visualization, yellow dots refer to detected nodes while light blue arrows denote directed edges.

embedded by feature embedder and time embedder (TE), separately. Feature embedder is a patch embedding module to transform F_s into sequence-token embeddings. Time embedder first creates sinusoidal positional embeddings according to timesteps t and adopts multilayer perceptron (MLP) to generate timestep embeddings. Both embeddings are added together to get conditional embeddings. As shown in Fig. 6(a), in each HiDiT block, an additional multi-head cross-attention layer is utilized to integrate conditional information with encoded noise.

Node decoder. As shown in Fig. 8a, the node decoder firstly receives generated representations $F_g^{(\frac{H}{P}\times\frac{W}{P})\times D_1}$ and obtains feature map $F^{D_1\times H\times W}$ through an MLP layer and an unpachify operation. Subsequently, two 1×1 kernel size convolution blocks reduce the channel of F for the predicted heatmap $H_f^{H\times W}$. After obtaining F and H_f, N object nodes $\mathbf{v}_i \in \mathcal{V}^{N\times 2}$ and their corresponding feature descriptors $\mathbf{d}_i \in \mathcal{R}^{N\times D_1}$ are extracted by non-maximum suppression (NMS) [10] and grid sampling [8] approaches, respectively. These node coordinates and descriptors are sequentially utilized to produce conditions in the following EGDiT.

2) Edge biased graph diffusion transformer: Similarly, we introduce an edge biased graph diffusion transformer (EGDiT) to predict the connectivity among N extracted nodes in the form of the adjacency matrix $\mathcal{A}^{N\times N}$. EGDiT consists of stacked diffusion blocks and a light graph decoder. The transformer based architecture of EGDiT ensures that each predicted node $\boldsymbol{v}_i$ owns a global perception of all other nodes in the entire $\mathcal{V}$ set, facilitating reliable learning of adjacency graphs. Moreover, in order for the better understanding of graph structure, an edge biased attention (EBA) is incorporated in the self-attention mechanism within EGDiT blocks. The details of aforementioned three designs are as follows.

EGDiT blocks. Fig. 6(b) shows the structure of EGDiT blocks, which is a conditional diffusion scheme. In EGDiT, one of the input conditions is embedded descriptors $\mathbf{d}_{emd} \in \mathcal{R}^{N\times(D_1+2)}$ that concatenates detected nodes $\mathcal{V} \in \mathcal{R}^{N\times 2}$ and visual description $\mathbf{d} \in \mathcal{R}^{N\times D_1}$. Receiving $\mathbf{d}_{emd} \in \mathcal{R}^{N\times(D_1+2)}$, a two-layer MLP merges node positions and corresponding visual descriptors to produce positional descriptors $\mathbf{d}_{pos} \in \mathcal{R}^{N\times D_2}$. Next, EGDiT blocks process the $\mathbf{d}_{pos}$ and output generated visual descriptors $d_g^{N\times D_2}$. Inspired by [16], it employs adaptive layer normalization (adaLN) instead of the standard layer normalization method. Specifically, EGDiT conducts regression on the dimensional scale and shift parameters, denoted as λ and μ respectively, leveraging the sum of the embedding vectors of additional conditions. Additionally, it incorporates dimension-wise scaling parameters denoted as γ, which are zero-initialized and applied just prior to any residual connections within each EGDiT block. This zero-initialization ensures that each block functions as an identity operation, thereby expediting training.

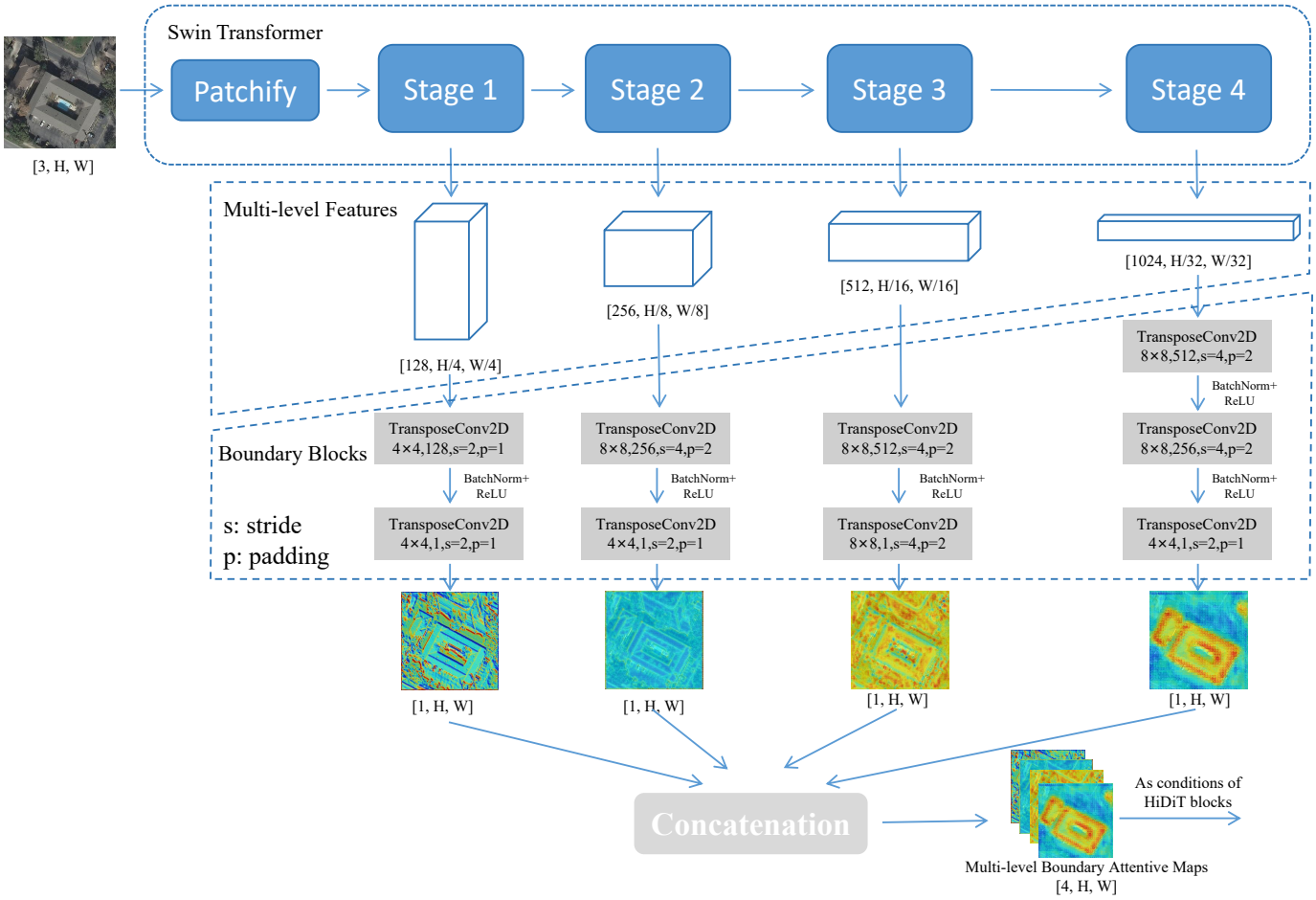


Figure 5. The structure of the topology-concentrated Swin Transformer (TCSwin).

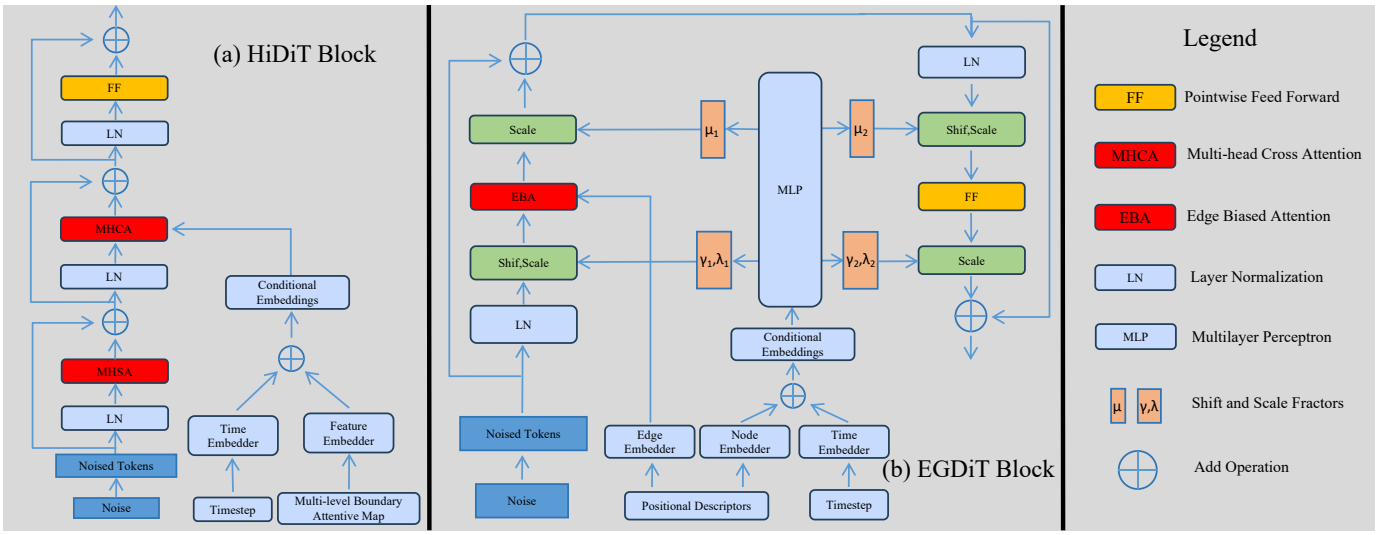


Figure 6. The details of the HiDiT block and EGDiT block.

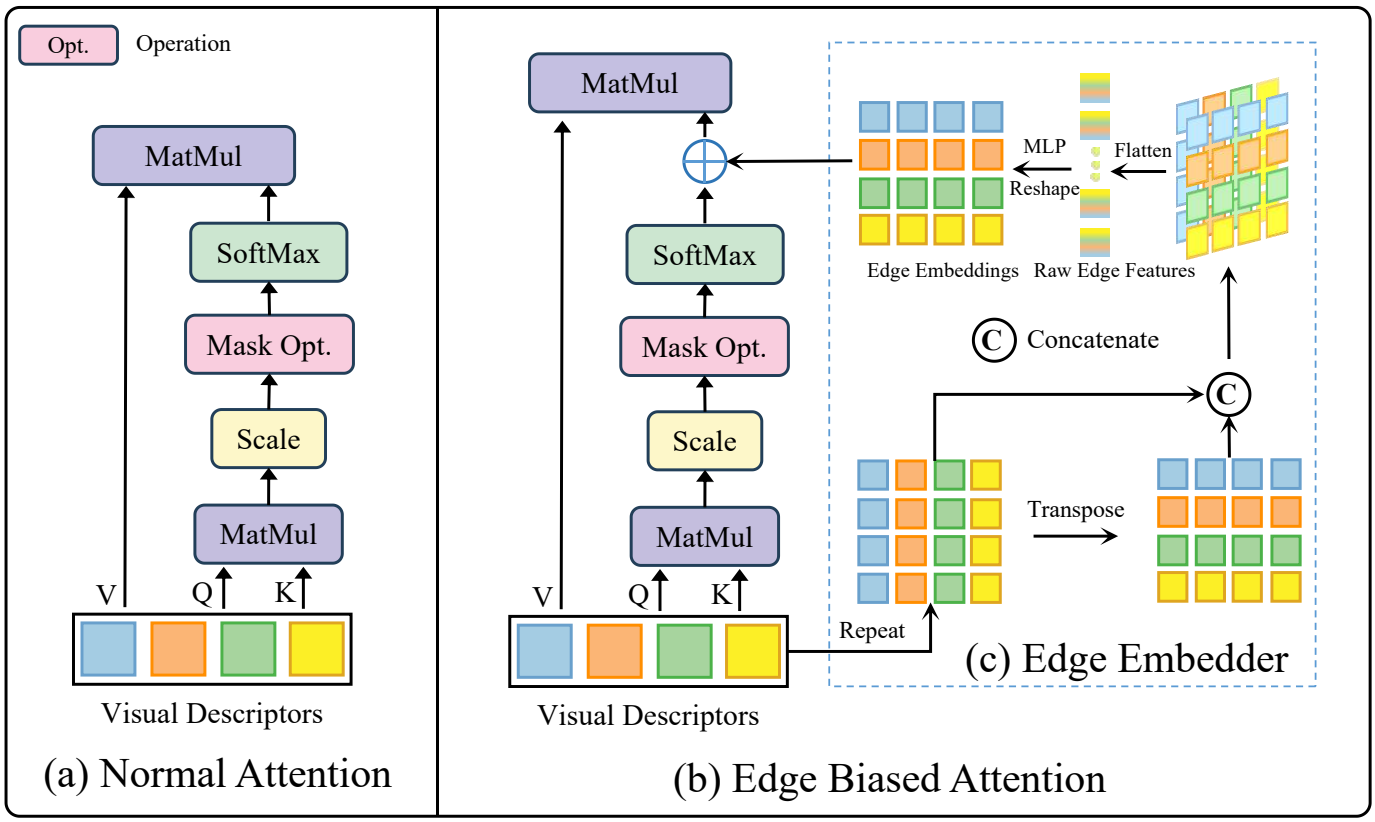


Figure 7. The visual comparison between normal self-attention mechanism and the proposed edge biased attention (EBA).

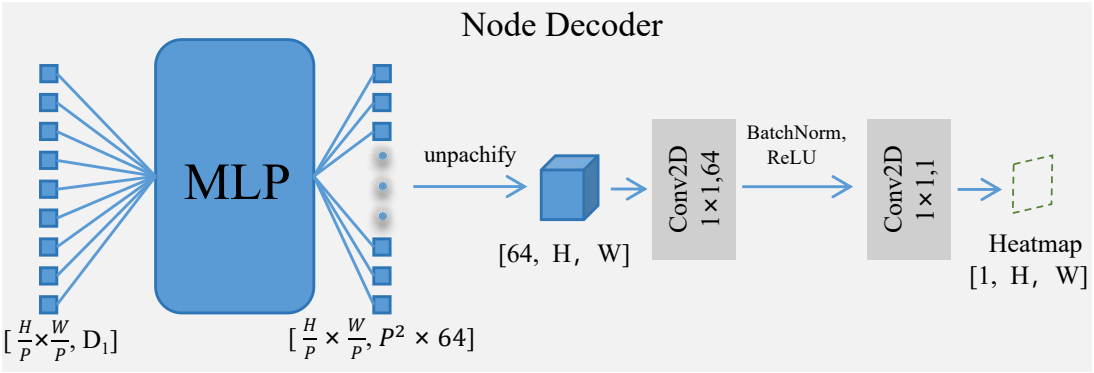
Edge biased attention. EGDiT manages to incorporate edge information for the better graph understanding by conducting edge biased attention (EBA) in self-attention (SA) mechanism of EGDiT blocks. As shown in Fig.7, the difference between regular self-attention in Transformer architectures and the proposed EBA is that the EBA produces extra edge embeddings to bias each attention matrix in the self-attention mechanism. In the EGDiT blocks, query, key and values for SA modules are all derived from visual descriptors $d_g^{N\times D_2}$. Consequently, the attention matrix can be interpreted as matching scores between node pairs, which implicitly encode edge features. In this context, edge embeddings, directly computed from $d_g^{N\times D_2}$ via the edge embedder (Fig.7(c)), serve as the shortcut to each attention matrix, stabilizing the SA processes.

Concretely, to obtain edge embeddings, we firstly reshape generated visual descriptors $\mathbf{d}_g^{N\times D_2}$ to $D_2 \times N$ and repeat it in a new channel to $\mathbf{d}_r^{D_2\times N\times N}$. After that, the $\mathbf{d}_r^{D_2\times N\times N}$ and its transposed version are concatenated and reshaped as the raw edge features $e_r^{(N\times N)\times(D_2\times 2)}$. Then the $e_r^{(N\times N)\times(D_2\times 2)}$ is fed into an MLP layer and yield the final edge embeddings $e^{h\times N\times N}$, where h denotes the number of transformer heads. As shown in Fig. 6(b), an edge biased attention layer adds edge embeddings to the encoded noise after multi-head self-attention.

Graph decoder. Generated visual descriptors $\mathbf{d}_g$ are fed into the graph decoder for the adjacency graph predictions. As shown in Fig. 8b, the graph decoder consists of a two-layer MLP and two 1×1 kernel convolutional layer, a batch normalization layer, and a rectified linear unit (ReLU). In EGDiT, two graph decoders, termed by clockwise and counter-clockwise graph decoders, with the same structures are adopted to predict clockwise and counter-clockwise connectivity between node pairs. The graph decoders receive the descriptors $\mathbf{d}_g$ and generates two directional adjacency matrices $\mathcal{A}_1 \in \mathcal{R}^{N\times N}$ and $\mathcal{A}_2 \in \mathcal{R}^{N\times N}$, which record clockwise and counter-clockwise connections among N detected nodes, respectively. These two graphs are added up to export the final directional adjacency graph $\mathcal{A}^{N\times N}$. Following common practice [9], [10], the $\mathcal{A}^{N\times N}$ is optimized through the Sinkhorn algorithm [76].

C. Isomorphic training strategy

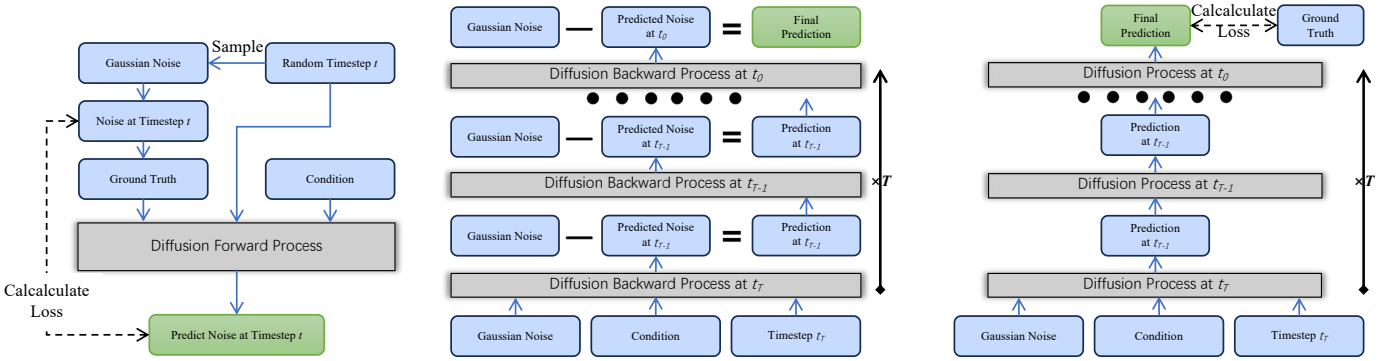
During training, both HiDiT and EGDiT follow an isomorphic training strategy (ITS) to establish training as the mirror of inference process (Algorithm 1). As shown in Fig. 9, two main characteristics distinguish ITS from previous DDIM based works [19], [68], [71]. On the one hand, preliminary experiments (see Table I) and existing work [18] have reported that the regular DDIM paradigm which inputs noisy ground truth (GT) targets for training (see Fig. 9(a)) is prone to encounter overfitting problem when solving dense prediction tasks. Therefore, ITS starts from a random Gaussian noise instead of noisy GT and gradually denoises the initialized noise to desired outcomes. On the other hand, recent advanced DM-based dense and sparse prediction models have demonstrated that directly supervising the training based on final targets



(a) Node Decoder

(b) Graph Decoder

Figure 8. An illustration of the node decoder and graph decoder structures.



(a) Training and inference in previous DM-based works

(b) Training and inference with ITS

Figure 9. The visual comparison of schematic diagrams between training and inference diffusion processes in previous DM-based works and the proposed isomorphic training strategy (ITS).

can achieve better performance than the conventional noise supervision [12]–[14], [18]. Therefore, the training objective of ITS is minimizing the discrepancy between predicted and ground truth outcomes (i.e. heatmap in HiDiT and adjacency matrix in EGDiT), rather than noise adopted in generative DMs.

Concretely, in both training and inference, we firstly initialize a Gaussian noise with the same shape as wanted outcomes and evenly sample multiple timesteps from the total sampling steps $t \in \{0, 1, \ldots, T\}$. Then the conditional denoising process is executed to yield desired outputs in a step-by-step manner. The denoising loop is applied to the entire 'DiT+decoder' structure. It means that each step can produce an intermediate output which is corrupted by next-level noise and acts as the input of the next step. Furthermore, the learning of this denoising process is based on the output results, rather than predicted noise at each step. Details of adopted noise schedule, time embedder and training objective are as follows:

Noise schedule. Noise schedule is a very important hyper-parameter, which controls the difficulty of denoising procedure [18]. During the training, as the input of the model, the degree of noise in $\boldsymbol{z}_t$ is completely determined by noise schedule β_t. Different data distributions often show different degrees

Algorithm 1 The pseudo code for training and inference procedures with the isomorphic training strategy (ITS). This process is executed for both HiDiT and EGDiT. The required input 'cond' denotes multi-level boundary attentive maps $F_s^{4\times H\times W}$ for HiDiT and positional node descriptors $\mathbf{d}_{pos} \in \mathcal{R}^{N\times D}$ for EGDiT.

```
def train_and_infer_with_ITS(cond):
  # condition embedding
  cond_emb = cond_embedder(cond)
  # initilize a random Gaussian noise
  noise = normal(0, 1)
  # in this work sample_number=10 (see Section 4.1.2)
  for step in range(sample_number):
    # time intervals
    t_now = 1- step / steps
    t_next = max(1 - (step + 1 + td) / steps, 0)
    map_denoised = DiTBlocks(noise, cond_emb)
    map_pred = decoder(map_denoised)
    # update noise for the next time step
    pred_noise = (noise - sqrt(sigmoid(t_now)) * sigmoid(map_pred)) / sqrt(sigmoid(-t_now))
    noise = sigmoid(map_pred) * sqrt(sigmoid(t_next)) + pred_noise * sqrt(sigmoid(-t_next))
  if train:
    loss = objective_func(map_pred, gt)
    return map_pred, loss
  else:
    return map_pred
```

of information redundancy, so the same level of independent noise destroys information differently. The optimal schedule for denoising may not yield the same optimal results within a different data distribution. Furthermore, an inadequate noise schedule could result in insufficient training regarding certain noise levels. Following [18], we utilize the improved cosine schedule:

$$\beta_t = -log\cos\left(\frac{t/T+s}{1+s}\cdot\frac{\pi}{2}\right)^{-2} - 1, \tag{2}$$

where small offset s is introduced to prevent β_0 from becoming excessively small for precise prediction and the cosine function is employed to ensure that β_t changes gradually, particularly when t approaches 0 or T, leading to a more stable generation process.

Time embedder. The time embedder (TE) adopts sinusoidal position embedding to inject timestep information, which is also vitally important in diffusion model:

$$\mathrm{TE}_{t,d} = \begin{cases} \sin\left(\frac{t}{mp^{2d/dim}}\right), & \text{if } d \text{ is even} \\ \cos\left(\frac{t}{mp^{2(d-1)/\dim}}\right), & \text{if } d \text{ is odd,} \end{cases} \tag{3}$$

where t denotes timesteps, d is the dimension index and dim represents output dimension. mp controls the minimum frequency of the embeddings.

Objective function. During HiDiT training, the final heatmap H_f is supervised by means square error (MSE) loss $\mathcal{L}_{dt}$ as follows:

$$\mathcal{L}_{dt} = \mathcal{M}\left(\mathbf{H}_f - \bar{H}_{gt}\right)^2, \tag{4}$$

where $\mathcal{M}$ denotes the arithmetic mean operation and $\bar{H}_{gt}$ is the ground truth heatmap of shape $H \times W$.

In the EGDiT training procedure, the predicted adjacency graph is supervised by a binary cross-entropy loss:

$$\mathcal{L}_{\text{gh}} = -(\overline{\mathcal{A}}\log(\mathcal{A}) + (\mathbf{1} - \overline{\mathcal{A}})\log(\mathbf{1} - \mathcal{A})), \tag{5}$$

where $\mathcal{A}$ represents the predicted adjacency graph and $\overline{\mathcal{A}}$ is the adjacency graph label. To solve the conflicts and balance problems between the losses of various tasks during the training process and improve the model training speed and training quality, we adopted the multi-task loss (MTL) [77] as follows:

$$\mathcal{L} = \sum_{\tau\in\mathcal{T}} \frac{1}{2\sigma_\tau^2}\cdot\mathcal{L}_\tau + ln(1+\sigma_\tau^2) \tag{6}$$

where τ indicates task loss index, $\mathcal{T}$ counts losses including the heatmap loss $\mathcal{L}_{dt}$ and the adjacency matrix loss $\mathcal{L}_{\text{gh}}$. σ is learnable parameters.

IV. Experimental Settings

A. Implement details

Network architectures. For TCSwin, the image encoder adopts Swin Transformer-B. HiDiT/EGDiT blocks consist of 6/12 transformer blocks with 384/768 feature hidden dimensions and 12/12 heads. Patch size P adopted in HiDiT is set as 8. The sample number is set to 10 in both the training and inference procedure (see Section V-A2) and time schedule ranges from 0 to 0.999. For each image, we extract $N = 320$ nodes, which can delineate contours of all building instances in most cases. All parameters are trainable.

Training. To begin with, DiffVector pretrains the TCSwin by adding a light decoder to conduct Eq. 4 supervised heatmap prediction. Afterward, the entire HiDiT is trained to detect precise nodes and obtain reliable features which are supervised by Eq. 10. Finally, the DiffVector is trained as a whole to output the building vector graphs, which is supervised by the Eq. 6. The training stage is executed utilizing Adam optimizer,

10^{-4} learning rate and early stopping strategy. The supporting platform is equipped with NVIDIA Tesla V100 32GB GPU and Intel Xeon Gold 5218 CPU @ 2.3GHz.

B. Datasets

DiffVector is evaluated on Inria Aerial Image Labeling dataset (Inria) [78] and CrowdAI Map Challenge dataset (CrowdAI) [79], covering polygonal buildings extraction from both aerial and satellite images.

Inria dataset provides a training set of 180 aerial RGB images and their binary annotation maps for the building segmentation task. The raw images are evenly acquired from 5 cities and are 5000×5000 pixels with a spatial resolution of $0.3m$. We split the first and last images of each city into validation set, resulting in 170 and 10 samples for training and validation, respectively. Then all raw images are cropped to tiles of 320×320 pixels with a stride size of 320 pixels and tiles without buildings are discarded.

CrowdAI dataset is comprised of 280741 and 60317 WorldView3 satellite images for training and validation. These images contain RGB spectral bands and each image measures 300×300 pixels with a spatial resolution of $0.3m$. Contours of building instances in all images are initially recorded in the form of MS COCO format. The small version of the validation set, which holds 1820 samples is adopted in this work. Notably, the raw dataset has been found to contain severe duplication and data leakage problems [80] trough the hashing method. Therefore, we followed [80] to filter out all duplicated samples from both training and validation sets. In addition, training samples duplicated or augmented from validation samples were also removed. The resultant training and validation sets contain 67440 and 1480 samples. The resultant training and validation sets contain 67440 and 1480 samples.

C. Evaluation metrics

We comprehensively evaluate the performance of DiffVector from three perspectives, namely mask-wise, boundary-wise and graph-wise metrics.

Mask-wise metrics are calculated from GT and predicted binary segmentation masks, which are rasterized from extracted building graphs (Fig. 10). During the evaluation, metrics of the thematic vector extraction are calculated through a two-class (foreground and background) confusion matrix (*CM*). In a *CM*, the true positive (*TP*) sum up the diagonal values, the false positive (*FP*) and false negative (*FN*) are the summation of column and row non-diagonal values, respectively. The mask-wise accuracy of DiffVector is measured by averaging intersection over union (mIoU^{mask}) of foreground and background, following Eq. 7.

$$\begin{aligned} TN &= SUM(CM) - (TP + FN + FP) \\ \text{OA} &= \frac{TP + TN}{TP + FN + FP + TN} \\ \text{Precision} &= \frac{TP}{TP + FP} \\ \text{Recall} &= \frac{TP}{TP + FN} \\ \text{F1} &= \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \\ \text{IoU} &= \frac{TP}{TP + FP + FN} \end{aligned} \tag{7}$$

Boundary-wise metrics measures differences between predicted and GT boundary region maps. The involved metrics include overall accuracy (OA^{bdy}), precision (Precision^{bdy}), recall (Recall^{bdy}), F1-score (F1-score^{bdy}) and mean IoU (mIoU^{bdy}). They are calculated following Eq. 7 and all boundary-wise metrics average the scores of both foreground and background for more comprehensive evaluation. By focusing on buffered building boundaries, these metrics are more sensitive to errors of predicted boundaries. As shown in Fig. 10, to obtain the boundary region masks, we first transform extraction polygons to one-pixel boundary masks and then apply dilation operation to them based on the 5×5 window size kernel. The GT boundary area maps are calculated from GT segmentation masks through edge detection and dilation operation.

Graph-wise metrics directly evaluate the topology completeness of predicted building vector graphs (Fig. 10). We adopt the average path length similarity (APLS) which studies bidirectional differences between lengths of all unique Dijkstra's shortest paths [81] in the ground truth graph vector graph $\bar{\mathbf{G}}$ and predicted graph $\mathbf{G}$. The APLS metric scores fall in the range of 0 (poor) to 1 (perfect). Given an image sample, the APLS for the thematic vector extraction is calculated as in Eq. 8:

$$\mathbf{APLS} = \frac{\mathbf{2}}{\frac{\mathbf{1}}{\mathbf{S}_{\mathbf{G}\to\bar{\mathbf{G}}}} + \frac{\mathbf{1}}{\mathbf{S}_{\mathbf{G}\leftarrow\bar{\mathbf{G}}}}}, \tag{8}$$

where

$$S_{\mathbf{G}\to\bar{\mathbf{G}}} = 1 - \frac{1}{NU}\sum min\left\{1, \frac{|L(a,b) - L(a',b')|}{L(a,b)}\right\} \tag{9}$$

denotes the difference score of path lengths mapping from $\mathbf{G}$ to $\bar{\mathbf{G}}$; $L(a,b)$ indicates the Dijkstra's shortest path from node a to node b in $\bar{\mathbf{G}}$ while $L(a',b')$ is the predicted one in $\mathbf{G}$; NU means the number of unique paths for this class.

V. Results and Discussion

A. Ablation study

To verify the effectiveness of designed components and parameters in DiffVector, we conduct comprehensive ablation experiments on the Inria dataset. Specifically, the importance of different network architectures in HiDiT and EGDiT is evaluated, including with/without ITS strategy, edge bias attention (EBA), multi-task loss (MTL) and whether adopt diffusion scheme or not (Section V-A1). Then we investigate the influences of different sampling steps in accuracy and

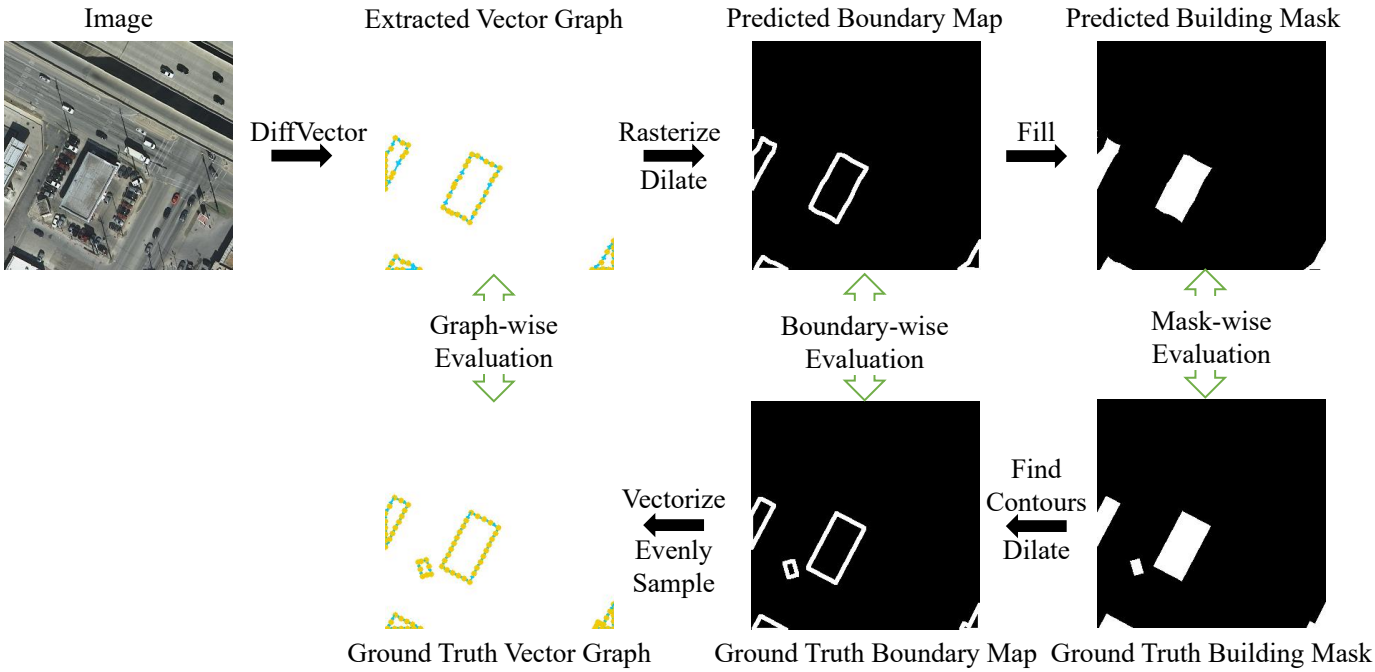


Figure 10. The schematic diagram of performing mask-wise, boundary-wise and graph-wise evaluations.

model complexity (Section V-A2). Furthermore, the runtime comparison between ITS and previous diffusion paradigms is conducted in the Section V-A3.

1) Importance of different components: Experiments in Table I measure the effectiveness and contributions of different components in DiffVector. The baseline architecture adopts all proposed designs (i.e. HiDiT, EGDiT, EBA, ITS and MTL) and the effectiveness of each component is measured by the accuracy changes after disusing it. All experiments are trained with a batch size of 12, a fixed learning rate of 10^{-4}, the Adam optimizer and the early stop strategy.

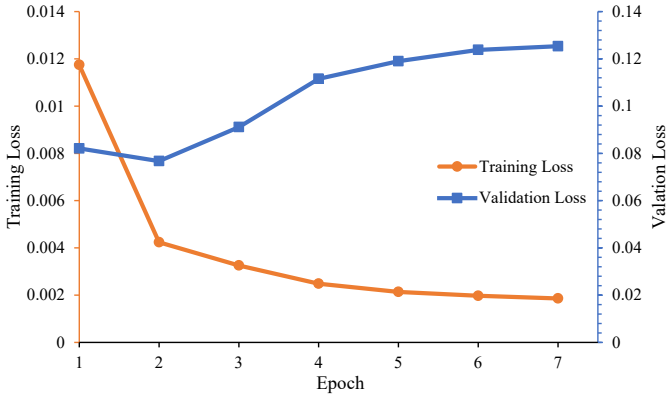


Figure 11. The training and validation losses of each epoch.

From the perspective of sub-tasks, the diffusion paradigm is crucial for both node detection and adjacency graph generation tasks. Aiming at node detection, HiDiT blocks generate representations conditioning the image features extracted by the TCSwin (see Section III-B1). The resultant representations are utilized not only for node detection but also serve as conditions for the generation of visual descriptors in the EGDiT. Therefore, the changes in HiDiT lead to more influences on the final performance of DiffVector than EGDiT. To be specific, replacing the diffusion-based HiDiT with sole TCSwin results in remarkable decreases in all metric scores with 4.81%, 5.14% and 7.05% lower mIoU^{mask}, mIoU^{bdy} and APLS. As to the adjacency graph prediction task, using diffusion-based EGDiT produces better results than deploying standard Vision Transformer with 0.48%, 0.71% and 2.83% increases in mIoU^{mask}, mIoU^{bdy} and APLS.

From the perspective of key components and training strategies, the importance of the ITS has been demonstrated for both HiDiT and EGDiT, but to different degrees of criticality. As illustrated by Fig. 11, without ITS, HiDiT encounters severe overfitting and eventually makes the training of following EGDiT infeasible. Somewhat differently, instead of the complete failure, lacking ITS in EGDiT leads to an accuracy drop on all three levels of metrics with 0.66%, 1.00% and 2.45% lower mIoU^{mask}, mIoU^{bdy} and APLS, receptively. Furthermore, as to EGDiT, the edge biased attention (EBA) benefits the performance with 0.18%, 0.27% and 0.92% increases in mIoU^{mask}, mIoU^{bdy} and APLS. Finally, training DiffVector the multi-task loss improves the performance with 0.12%, 0.21% and 0.83% increases in mIoU^{mask}, mIoU^{bdy} and APLS.

2) Influences of sampling number: The sampling number determines the granularity (i.e. steps) of sampling from Gaussian Noise during the denoising process. It not only affects the prediction quality of the final results, but also directly leads to changes in model size and computing expense of DiffVector. Therefore, we analyze the influences of sampling numbers by measuring the accuracy, total parameter size (#Params.) and multiply-accumulate operations (MACs). Studies in this section adopt DiffVector with all proposed components and

Table I. Impacts of key components.

HiDiT blocks	HiDiT ITS	EGDiT blocks	EGDiT ITS	EBA	MTL	mIoU^{mask}	mIoU^{bdy}	APLS
✓	✓	✓	✓	✓	✓	85.87	71.16	44.63
Using Sole TCSwin		✓	✓	✓	✓	81.06(-4.81)	66.02(-5.14)	37.58(-7.05)
✓	✓	*Using Standard Vision Transformer*			✓	85.39(-0.48)	70.45(-0.71)	41.80(-2.83)
✓	×	✓	✓	✓	✓	Encounter Failure due to Overfitting		
✓	✓	✓	×	✓	✓	85.21(-0.66)	70.16(-1.00)	42.18(-2.45)
✓	✓	✓	✓	×	✓	85.69(-0.18)	70.89(-0.27)	43.71(-0.92)
✓	✓	✓	✓	✓	×	85.75(-0.12)	70.85(-0.21)	43.80(-0.83)

Table II. Influences of sampling steps on network complexity and accuracy. 'A+B+C' in #Params. and MACs columns donate sequentially adding up statistics of TCSwin, HiDiT excluding TCSwin and EGDiT.

HiDiT Steps	EGDiT Steps	#Params.(Mb)	MACs (T)	mIoU^{mask}	mIoU^{bdy}	APLS
1	1	121.94+16.28+135.77=273.95	0.52+0.03+0.38=0.93	Encounter Failure due to Overfitting		
5	5	121.94+16.28+135.77=273.95	0.52+0.13+0.54=1.19	85.83	70.96	43.21
10	10	121.94+16.28+135.77=273.95	0.52+0.26+0.75=1.53	85.87	71.96	44.64
20	20	121.94+16.28+135.77=273.95	0.52+0.52+1.17=2.21	86.12	71.22	44.05

Table III. The efficiency comparison between ITS and two conventional diffusion paradigms. '[A, B]' in Steps donate the sample step during training and inference, respective. FPS means frame per second.

Paradigm	Steps	Training FPS	Inference FPS
DDPM	[1, 1000]	4.35	0.28
DDIM	[1, 10]	4.45	13.31
ITS	[10, 10]	3.99	13.15

training strategies before (see Section V-A1). The timesteps are evenly sampled from the complete timestep set 0-1000 with fixed interval strides. Limited by accessible computing resources, we only test 4 different settings of the sample number, namely 1, 5, 10 and 20, as presented in Table II. Note that in DiffVector, the ITS scheme determines that the sampling number is identical between the training and inference phases, as described in Section III-C.

Experiments reveal that more sampling steps keep improving overall performance with the increases in mIoU^{mask} from 85.83% to 86.12%. While the topology quality of results reaches peak when the sampling number is 10 and decreases after that. In terms of complexity analysis, enlarging the sampling number does not increase the total parameter size of DiffVector, but it exponentially inflates the number of computation operations. Given the aforementioned statics, we set the sampling number as 10 to achieve the best trade-off between the accuracy and efficiency.

3) The efficiency analysis of different diffusion paradigms: We further evaluate the running speed of DiffVector with different diffusion schemes during training and inference phases via the metric of frame per second (FPS). The DDPM and DDIM paradigms share the same training procedure where one 'diffuse-denoise' process occurs, thereby they report similar runtime for the training stage. In contrast, the proposed ITS iteratively denoises from Gaussian noise to final predictions, which conducts multiple 'diffuse-denoise' procedures and leads to slower training FPS than DDPM and DDIM. During the inference stage, ITS and DDIM implement similar 'skip-step' denoise process, thus they obtain a similar running efficiency which is much faster than step-by-step denoise scheme in DDPM.

It is worth noting that, despite the theoretical expectation that the training and inference phases of ITS should consume similar amounts of time due to their strictly identical procedures, there is actually a significant difference in the calculated FPS. This discrepancy arises because, in our FPS calculations, we intentionally retained processes that are exclusive to the training phase but not needed for inference, aiming for a more realistic representation of practical applications. Such processes encompass online adjacency matrix label generation, loss backpropagation, and model parameter updates.

B. Comparison with state-of-the-art methods

We compare DiffVector with other relevant state-of-the-art (SOTA) vector extraction approaches on the Inria and CrowdAI datasets from the quantitative and qualitative perspectives.

1) Quantitative comparison: Table IV and Table V present the quantitative results of different approaches on the Inria and CrowdAI datasets, representative. For the Inria dataset, DiffVector achieves state-of-the-art performance with respect to all mask-wise, boundary-wise and graph-wise metrics, reporting 85.87% mIoU^{mask}, 71.16% mIoU^{bdy} and 44.63% APLS, respectively. It outperforms other modern approaches with at least 1.2%, 4.47% and 5.97% higher mIoU^{mask}, mIoU^{bdy} and APLS. Among segmentation-based methods, HD-Net establishes an extra branch to perceive boundary features and extracts more accuracy and regular building

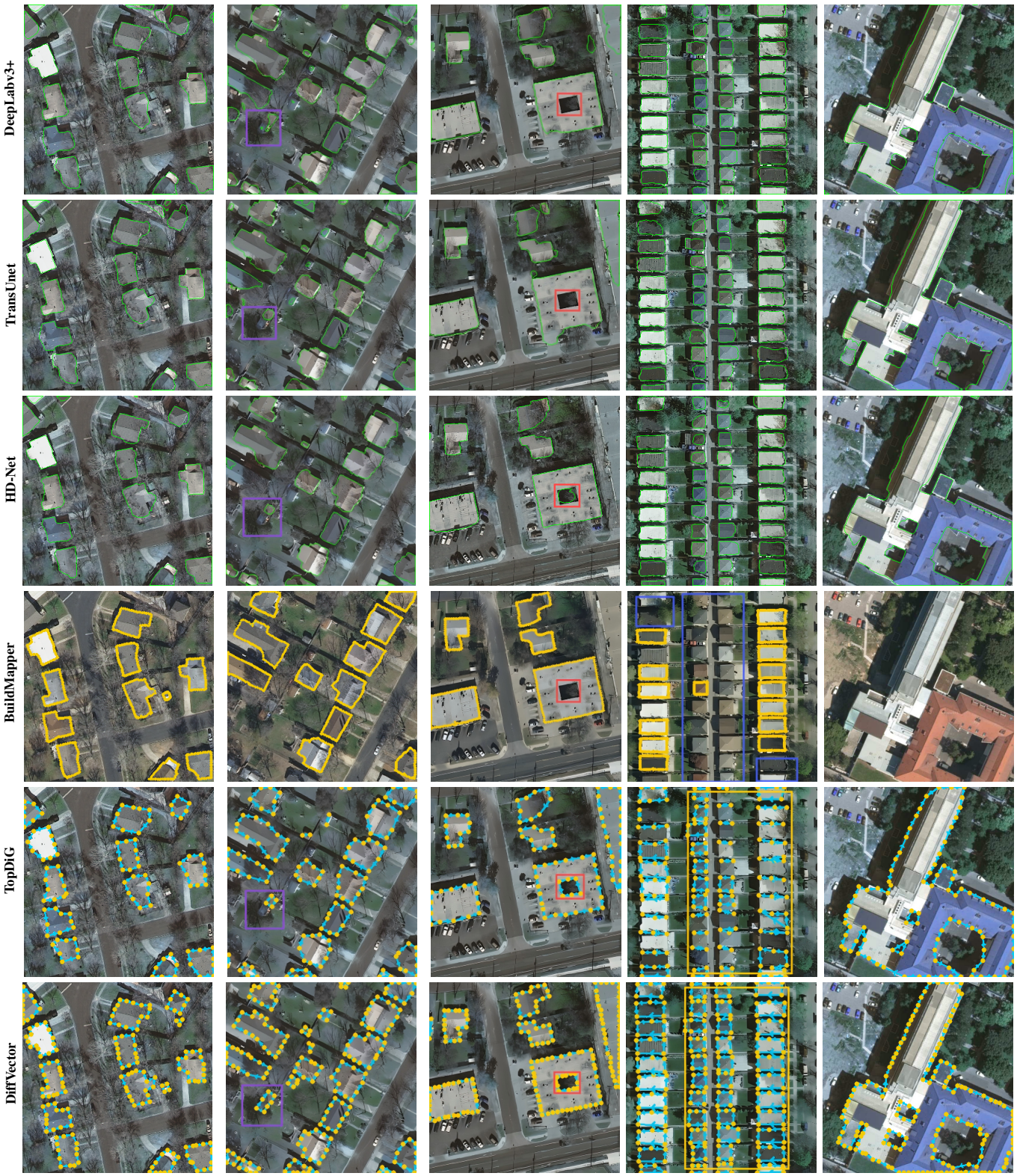


Figure 12. Visual comparisons on the Inria dataset. Green lines draw the vectorized segmentation results. Red, blue, yellow and purple rectangles indicate the superiority of DiffVector over other methods in non-trivial scenarios, namely, hollows, dense buildings, adjacent walls and occluded buildings.

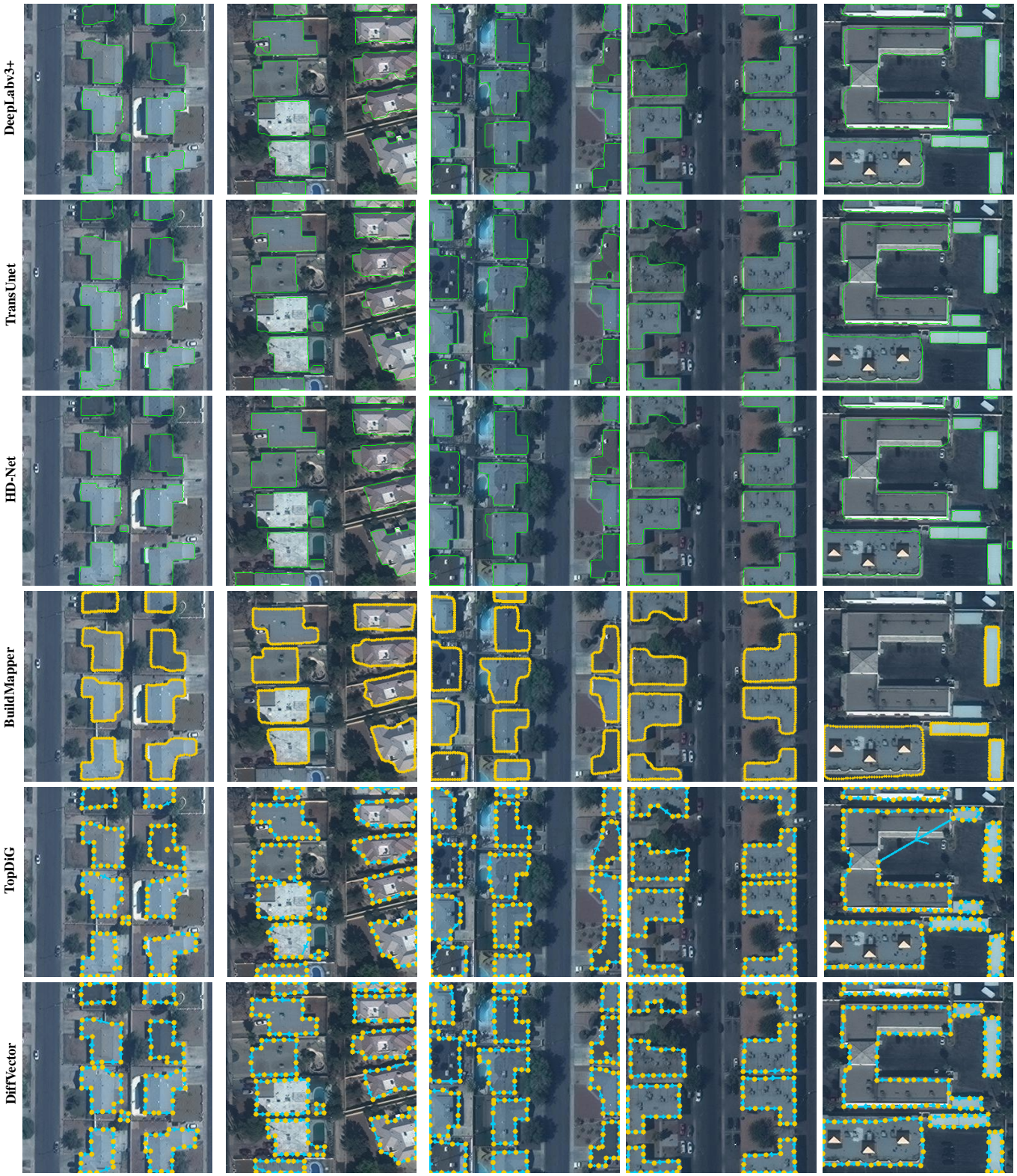


Figure 13. Visual comparisons on the CrowdAI dataset.

Table IV. Quantitative comparisons on the Inria dataset.

Method	mIoU^{mask}	OA^{bdy}	Precision^{bdy}	Recall^{bdy}	F1-score^{bdy}	mIoU^{bdy}	APLS
Segmentation-based:							
HRNet	82.42	93.00	77.29	72.88	74.83	64.59	35.42
DeepLabv3+	84.67	93.39	78.46	75.59	76.93	66.69	38.50
TransUnet	83.26	93.01	76.98	75.14	76.02	65.71	37.66
UNetFormer	80.00	92.30	74.48	71.38	72.79	62.55	31.89
HD-Net	83.88	93.60	79.38	75.74	77.39	67.22	37.70
Contour-based:							
DeepSnake	57.58	91.91	71.04	58.20	58.93	52.79	26.56
BuildMapper	64.42	93.27	78.34	64.23	65.97	58.88	39.46
Node-based:							
TopDiG	82.92	93.61	79.82	74.08	76.54	66.41	38.66
DiffVector	**85.87**	**94.57**	**82.85**	**79.28**	**80.93**	**71.16**	**44.63**

Table V. Quantitative comparisons on the CrowdAI dataset.

Method	mIoU^{mask}	OA^{bdy}	Precision^{bdy}	Recall^{bdy}	F1-score^{bdy}	mIoU^{bdy}	APLS
Segmentation-based:							
HRNet	83.60	91.06	72.71	78.40	75.10	64.33	42.73
DeepLabv3+	**90.96**	94.51	82.90	**82.20**	82.54	72.89	41.55
TransUnet	90.21	94.28	82.17	81.40	81.78	71.98	40.24
UNetFormer	87.63	93.41	79.49	78.06	78.75	68.52	38.13
HD-Net	90.45	**94.68**	**83.66**	82.14	**82.89**	**73.34**	39.53
Contour-based:							
DeepSnake	58.09	92.17	72.77	57.60	58.65	52.45	33.62
BuildMapper	76.66	93.06	78.09	69.28	71.24	62.46	22.16
Node-based:							
TopDiG	87.89	93.76	80.86	78.17	79.43	69.33	42.42
DiffVector	88.90	94.17	81.90	80.69	81.27	71.42	**45.48**

polygons than other segmentation-based approaches. Contour-based approaches exhibit the worst performance in all three-fold evaluations. Within the scope of node-based methods, DiffVector exhibits superiority over TopDiG in all evaluated metrics, especially gaining much better topology completeness with 5.97% higher APLS.

For the CrowdAI dataset, DiffVector still produces reliable and stable predictions with competitive accuracy scores *w.r.t* other modern approaches. Particularly, it reports the highest APLS score of 45.48% among all methods, which implies the distinctive topology completeness of its predictions. DeepLabv3+ obtains the best overall performance with the highest mIoU^{mask} while HD-Net achieves the best scores on the OA^{bdy}, Precision^{bdy}, F1-score^{bdy} and mIoU^{bdy}. It is worth noting that though generally achieving SOTA performance, HD-Net ranks second to last on the APLS metric, perhaps due to its disadvantages in preserving topological details.

2) Qualitative comparison: Fig. 12 and Fig. 13 exhibit the visual comparison among DeepLabv3+, TransUnet, HD-Net, BuildMapper, TopDiG and DiffVector on the Inira and CrowdAI datasets, respectively. These comparisons evidently demonstrate that DiffVector owns the ability to detect hollows, delineate regular building polygons, and partially tackle occlusion and high density issues.

(a) Prediction (b) Detected nodes (c) Heatmap (d) Ground Truth

Figure 14. An typical example that DiffVector fails to deal with the extreme occlusion issue and adjacent walls.

As shown in the Fig. 12 third and fifth columns, DiffVector is always capable of recognizing hollows within building instances. By contrast, prevailing approaches DeepLabv3+, TransUnet, HD-Net, BuildMapper and TopDiG encounter more or less failure cases, implying their instability in this regard. In the Fig. 12 second column, purple rectangles imply that DiffVector and previous node-based approach TopDiG showcases capability of tacking the occlusion from trees. This advantage may be attributed to their topology-concentrated image encoders. Samples in Fig. 12 and Fig. 13 illustrate that building polygons extracted by DiffVector always appear more regular than those produced by other works. Within the realm of node-based paradigm, compared to previous TopDiG, DiffVector delineates more sharp building contours and reveals distinguished advantages in some non-trivial scenes, such as densely distributed buildings with adjacent walls (Fig. 12 yellow rectangles) and building occluded by trees (Fig. 12 purple rectangles). In addition, samples in Fig. 13 release the common data distribution in CrowdAI that the vast majority of scenes only contain low-complexity building instances. Consequently, the involved approaches perform well in most cases. The rareness of non-trivial scenarios may be an essential reason why DiffVector fails to show superiority over other methods in both quantitative and qualitative comparisons.

According to the quantitative and qualitative comparisons, unique advantages which differing the proposed DiffVector from various previous works can be summarized. Firstly, compared to segmentation-based approaches, DiffVector always delineates more regular building outlines and directly outputs building vector without any post-processing procedures, such as vectorization, regularization and simplification. Secondly, compared to contour-based methods, the DiffVector avoids misdetection errors from contour initialization step and is capable of tackling buildings with hollows, concave outlines and various scales. Finally, compared to previous node-based models, the DiffVector showcases superiority in non-trivial scenarios like adjacent or occluded buildings.

VI. Limitations and Future Work

Though DiffVector have exhibited pleasant performance, it still struggles with two main issues. On the one hand, DiffVector showcases some ability in tackling building shaded by trees and building walls closed to each other, such as Fig. 12 second column and Fig. 13 second column. However, it still fails to deal with some more extreme cases. As shown in the Fig. 14 red rectangle, the lush canopy obscures a part of the target building and consequently DiffVector loses the perception of that part, resulting in incomplete the predicted building vector graph. Besides, green rectangles indicate that DiffVector have troubles in distinguishing walls that are very close to each other. Fig. 14c clearly illustrates the subtle recolonization of the gap between these adjacent walls. On the other hand, as shown in Table VI, DiffVector exhibits less optimal training and inference runtimes when juxtaposed with other cutting-edge methodologies. The underlying principle of the diffusion framework dictates that multiple denoising steps are indispensable for attaining satisfactory generation outcomes, consequently impeding the runtime efficiency of DiffVector.

To address the occlusion issue, we plan to adopt the popular

Table VI. The runtime comparison between DiffVector and other methods in the Inria. FPS indicates frame per second.

Method	Training FPS	Inference FPS
Segmentation-based:		
HRNet	13.34	19.39
DeepLabv3+	5.66	19.09
TransUnet	3.21	16.86
UNetFormer	6.39	21.62
HD-Net	13.00	15.73
Contour-based:		
DeepSnake	22.51	25.04
BuildMapper	4.12	60.09
Node-based:		
TopDiG	8.30	13.45
DiffVector	3.99	13.15

multi-modal paradigm [82] to integrate multi-source data, such as synthetic aperture radar (SAR) [83] and multi-spectral images [84], to obtain perception under trees. As to adjacent walls, a potential solution is to establish an extra branch to refine the extracted node coordinates [8], which supervised by multi-task losses [85], [86]. In order to speed up the runtime of diffusion procedures, approaches such as model distillation [87], feature cacheing [88] and rectified flow [89] can be explored to accelerate the network computing or reduce the number of denoise steps.

VII. Conclusion

In this work, we have introduced the DiffVector, which is a novel denoising diffusion framework to directly extracts building vector graphs from remote sensing images. In the DiffVector, a hierarchical diffusion transformer (HiDiT) is established to generate robust representations for detecting candidates contour nodes and mining corresponding nodes features. It is conditioned by multi-level boundary attentive maps encoded from input images through a topology-concentrated Swin Transformer (TCSwin). Subsequently, we propose an edge biased graph diffusion transformer (EGDiT) that takes node features as conditions to generate visual descriptors for the prediction of the adjacency matrix. In EGDiT, we replace the standard self-attention mechanism with an edge biased attention (EBA) to incorporate edge features for better prediction of adjacency graphs. Furthermore, an isomorphic training strategy (ITS) is executed to formulate training procedures of both HiDiT and EGDiT as the exact mirror of the denoising process during the inference stage. Ablation analysis has evidently demonstrated the importance and effectiveness of introduced components. Quantitative and qualitative comparisons with other modern approaches reveal that DiffVector is capable of achieving competitive performance in the building vector graph extraction task. Meanwhile, the adeptness of DiffVector releases the potential of the diffusion paradigm in the vector extraction field. We hope this study provides valuable insights for further works in the realm of vector extraction.

Acknowledgment

This work was supported by the Key Research and Development Program of Hubei Province (No. 2023BAB173), the Chinese National Natural Science Foundation Projects (No. 41901265), a Major Program of the National Natural Science Foundation of China (No. 92038301), and was supported in part by the Special Fund of Hubei Luojia Laboratory (No. 220100028)

References

[1] A. G. Yeh, "Urban planning and gis," *Geographical information systems*, vol. 2, no. 877-888, p. 1, 1999.

[2] K. Lwin and Y. Murayama, "A gis approach to estimation of building population for micro-spatial analysis," *Transactions in GIS*, vol. 13, no. 4, pp. 401–414, 2009.

[3] G. Boo, E. Darin, D. R. Leasure, C. A. Dooley, H. R. Chamberlain, A. N. Lázár, K. Tschirhart, C. Sinai, N. A. Hoff, T. Fuller *et al.*, "High-resolution population estimation using household survey data and building footprints," *Nature communications*, vol. 13, no. 1, p. 1330, 2022.

[4] N. Girard, D. Smirnov, J. Solomon, and Y. Tarabalka, "Polygonal building extraction by frame field learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5891–5900.

[5] Y. Li, D. Hong, C. Li, J. Yao, and J. Chanussot, "Hd-net: High-resolution decoupled network for building footprint extraction via deeply supervised body and boundary decomposition," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 209, pp. 51–65, 2024.

[6] S. Wei, S. Ji, and M. Lu, "Toward automatic building footprint delineation from aerial images using cnn and regularization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 3, pp. 2178–2189, 2019.

[7] S. Peng, W. Jiang, H. Pi, X. Li, H. Bao, and X. Zhou, "Deep snake for real-time instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8533–8542.

[8] S. Wei, T. Zhang, S. Ji, M. Luo, and J. Gong, "Buildmapper: A fully learnable framework for vectorized building contour extraction," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 197, pp. 87–104, 2023.

[9] S. Zorzi, S. Bazrafkan, S. Habenschuss, and F. Fraundorfer, "Polyworld: Polygonal building extraction with graph neural networks in satellite images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1848–1857.

[10] B. Yang, M. Zhang, Z. Zhang, Z. Zhang, and X. Hu, "Topdig: Class-agnostic topological directional graph extraction from remote sensing images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1265–1274.

[11] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[12] Y. Ye, K. Xu, Y. Huang, R. Yi, and Z. Cai, "Diffusionedge: Diffusion probabilistic model for crisp edge detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 7, 2024, pp. 6675–6683.

[13] R. Feng, Y. Gao, T. H. E. Tse, X. Ma, and H. J. Chang, "Diffpose: Spatiotemporal diffusion model for video-based human pose estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 14 861–14 872.

[14] C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard, "Digress: Discrete denoising diffusion for graph generation," in *Proceedings of the 11th International Conference on Learning Representations*, 2023.

[15] C. Zhou, X. Wang, and M. Zhang, "Unifying generation and prediction on graphs with latent graph diffusion," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[16] W. Peebles and S. Xie, "Scalable diffusion models with transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4195–4205.

[17] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.

[18] Y. Ji, Z. Chen, E. Xie, L. Hong, X. Liu, Z. Liu, T. Lu, Z. Li, and P. Luo, "Ddp: Diffusion model for dense visual prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 21 741–21 752.

[19] B. Kolbeinsson and K. Mikolajczyk, "Multi-class segmentation from aerial views using recursive noise diffusion," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 8439–8449.

[20] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[21] O. Ronneberger, P. Fischer, and T. 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*. Springer, 2015, pp. 234–241.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[23] S. D. Khan, L. Alarabi, and S. Basalamah, "An encoder–decoder deep learning framework for building footprints extraction from aerial imagery," *Arabian Journal for Science and Engineering*, vol. 48, no. 2, pp. 1273–1284, 2023.

[24] N. Ahmadian, A. Sedaghat, and N. Mohammadi, "Building footprint extraction from remote sensing images with residual attention multi-

scale aggregation fully convolutional network," *Journal of the Indian Society of Remote Sensing*, vol. 52, no. 11, pp. 2417–2429, 2024.
[25] X. Dai, M. Xia, L. Weng, K. Hu, H. Lin, and M. Qian, "Multiscale location attention network for building and water segmentation of remote sensing image," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–19, 2023.
[26] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.
[27] X. Xiao, W. Guo, R. Chen, Y. Hui, J. Wang, and H. Zhao, "A swin transformer-based encoding booster integrated in u-shaped network for building extraction," *Remote Sensing*, vol. 14, no. 11, p. 2611, 2022.
[28] X. Chen, C. Qiu, W. Guo, A. Yu, X. Tong, and M. Schmitt, "Multiscale feature learning by transformer for building extraction from satellite images," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2022.
[29] S. Chan, Y. Wang, Y. Lei, X. Cheng, Z. Chen, and W. Wu, "Asymmetric cascade fusion network for building extraction," *IEEE Transactions on Geoscience and Remote Sensing*, 2023.
[30] H. Zhang, H. Dou, Z. Miao, N. Zheng, M. Hao, and W. Shi, "Extracting building footprint from remote sensing images by an enhanced vision transformer network," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.
[31] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.
[32] L. Wang, R. Li, C. Zhang, S. Fang, C. Duan, X. Meng, and P. M. Atkinson, "Unetformer: A unet-like transformer for efficient semantic segmentation of remote sensing urban scene imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 190, pp. 196–214, 2022.
[33] L. Wang, S. Fang, X. Meng, and R. Li, "Building extraction with vision transformer," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–11, 2022.
[34] A. Dosovitskiy, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.
[35] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica: the international journal for geographic information and geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.
[36] H. Guo, B. Du, L. Zhang, and X. Su, "A coarse-to-fine boundary refinement network for building footprint extraction from remote sensing imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 183, pp. 240–252, 2022.
[37] A. Hu, L. Wu, S. Chen, Y. Xu, H. Wang, and Z. Xie, "Boundary shape-preserving model for building mapping from high-resolution remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–17, 2023.
[38] H. Guo, X. Su, C. Wu, B. Du, and L. Zhang, "Decoupling semantic and edge representations for building footprint extraction from remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–16, 2023.
[39] C. Wang, J. Chen, Y. Meng, Y. Deng, K. Li, and Y. Kong, "Samplolybuild: Adapting the segment anything model for polygonal building extraction," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 218, pp. 707–720, 2024.
[40] S. Wei and S. Ji, "Graph convolutional networks for the automated production of building vector maps from aerial images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–11, 2021.
[41] Z. Du, H. Sui, Q. Zhou, M. Zhou, W. Shi, J. Wang, and J. Liu, "Vectorized building extraction from high-resolution remote sensing images using spatial cognitive graph convolution model," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 213, pp. 53–71, 2024.
[42] Y. Zhu, B. Huang, J. Gao, E. Huang, and H. Chen, "Adaptive polygon generation algorithm for automatic building extraction," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–14, 2021.
[43] Z. Shu, X. Hu, H. Dai, L. Duan, Z. Zhang, L. Zhang, and L. Zhang, "Robust extraction of vectorized buildings via bidirectional tracing of keypoints from remotely sensed imagery," *IEEE Transactions on Geoscience and Remote Sensing*, 2023.
[44] W. Zhao, C. Persello, and A. Stein, "Building outline delineation: From aerial images to polygons with an improved end-to-end learning framework," *ISPRS journal of photogrammetry and remote sensing*, vol. 175, pp. 119–131, 2021.
[45] W. Huang, H. Tang, and P. Xu, "Oec-rnn: Object-oriented delineation of rooftops with edges and corners using the recurrent neural network from the aerial images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–12, 2021.
[46] Z. Liu, H. Tang, and W. Huang, "Building outline delineation from vhr remote sensing images using the convolutional recurrent neural network embedded with line segment information," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–13, 2022.
[47] S. Gu, D. Chen, J. Bao, F. Wen, B. Zhang, D. Chen, L. Yuan, and B. Guo, "Vector quantized diffusion model for text-to-image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 696–10 706.
[48] X. Liu, D. H. Park, S. Azadi, G. Zhang, A. Chopikyan, Y. Hu, H. Shi, A. Rohrbach, and T. Darrell, "More control for free! image synthesis with semantic diffusion guidance," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 289–299.
[49] A. K. Bhunia, S. Khan, H. Cholakkal, R. M. Anwer, J. Laaksonen, M. Shah, and F. S. Khan, "Person image synthesis via denoising diffusion model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 5968–5976.
[50] L. Yang, J. Liu, S. Hong, Z. Zhang, Z. Huang, Z. Cai, W. Zhang, and B. Cui, "Improving diffusion-based image synthesis with context prediction," *Advances in Neural Information Processing Systems*, vol. 36, 2024.
[51] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural computation*, vol. 23, no. 7, pp. 1661–1674, 2011.
[52] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.
[53] C. Saharia, W. Chan, H. Chang, C. Lee, J. Ho, T. Salimans, D. Fleet, and M. Norouzi, "Palette: Image-to-image diffusion models," in *ACM SIGGRAPH 2022 conference proceedings*, 2022, pp. 1–10.
[54] N. Tumanyan, M. Geyer, S. Bagon, and T. Dekel, "Plug-and-play diffusion features for text-driven image-to-image translation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1921–1930.
[55] B. Li, K. Xue, B. Liu, and Y.-K. Lai, "Bbdm: Image-to-image translation with brownian bridge diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern Recognition*, 2023, pp. 1952–1961.
[56] M. Xia, Y. Zhou, R. Yi, Y.-J. Liu, and W. Wang, "A diffusion model translator for efficient image-to-image translation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
[57] Y. Wang, W. Yang, X. Chen, Y. Wang, L. Guo, L.-P. Chau, Z. Liu, Y. Qiao, A. C. Kot, and B. Wen, "Sinsr: diffusion-based image super-resolution in a single step," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 25 796–25 805.
[58] Z. Yue, J. Wang, and C. C. Loy, "Resshift: Efficient diffusion model for image super-resolution by residual shifting," *Advances in Neural Information Processing Systems*, vol. 36, 2024.
[59] R. Dong, S. Yuan, B. Luo, M. Chen, J. Zhang, L. Zhang, W. Li, J. Zheng, and H. Fu, "Building bridges across spatial and temporal resolutions: Reference-based super-resolution via change priors and conditional diffusion model," *arXiv preprint arXiv:2403.17460*, 2024.
[60] X. Zou, K. Li, J. Xing, Y. Zhang, S. Wang, L. Jin, and P. Tao, "Diffcr: A fast conditional diffusion framework for cloud removal from optical satellite images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 62, pp. 1–14, 2024.
[61] J. Sui, Y. Ma, W. Yang, X. Zhang, M.-O. Pun, and J. Liu, "Diffusion enhancement for cloud removal in ultra-resolution remote sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.
[62] J. Tian, L. Aggarwal, A. Colaco, Z. Kira, and M. Gonzalez-Franco, "Diffuse attend and segment: Unsupervised zero-shot segmentation using stable diffusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 3554–3563.
[63] T. Chen, L. Li, S. Saxena, G. Hinton, and D. J. Fleet, "A generalist framework for panoptic segmentation of images and videos," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 909–919.
[64] J. Chen, J. Lu, X. Zhu, and L. Zhang, "Generative semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7111–7120.
[65] J. Xu, S. Liu, A. Vahdat, W. Byeon, X. Wang, and S. De Mello, "Open-vocabulary panoptic segmentation with text-to-image diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2955–2966.
[66] A. Rahman, J. M. J. Valanarasu, I. Hacihaliloglu, and V. M. Patel, "Ambiguous medical image segmentation using diffusion models," in

Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2023, pp. 11 536–11 546.

[67] J. Wu, R. Fu, H. Fang, Y. Zhang, Y. Yang, H. Xiong, H. Liu, and Y. Xu, "Medsegdiff: Medical image segmentation with diffusion probabilistic model," in *Medical Imaging with Deep Learning*. PMLR, 2024, pp. 1623–1639.

[68] T. Amit, T. Shaharbany, E. Nachmani, and L. Wolf, "Segdiff: Image segmentation with diffusion probabilistic models," *arXiv preprint arXiv:2112.00390*, 2021.

[69] B. Ke, A. Obukhov, S. Huang, N. Metzger, R. C. Daudt, and K. Schindler, "Repurposing diffusion-based image generators for monocular depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 9492–9502.

[70] Y. Duan, X. Guo, and Z. Zhu, "Diffusiondepth: Diffusion denoising approach for monocular depth estimation," in *European Conference on Computer Vision*. Springer, 2025, pp. 432–449.

[71] W. G. C. Bandara, N. G. Nair, and V. M. Patel, "Ddpm-cd: Remote sensing change detection using denoising diffusion probabilistic models," *arXiv preprint arXiv:2206.11892*, 2022.

[72] Y. Wen, X. Ma, X. Zhang, and M.-O. Pun, "Gcd-ddpm: A generative change detection model based on difference-feature guided ddpm," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[73] C. Niu, Y. Song, J. Song, S. Zhao, A. Grover, and S. Ermon, "Permutation invariant graph generation via score-based generative modeling," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 4474–4484.

[74] J. Jo, S. Lee, and S. J. Hwang, "Score-based generative modeling of graphs via the system of stochastic differential equations," in *International conference on machine learning*. PMLR, 2022, pp. 10 362–10 383.

[75] K. K. Haefeli, K. Martinkus, N. Perraudin, and R. Wattenhofer, "Diffusion models for graphs benefit from discrete state spaces," *arXiv preprint arXiv:2210.01549*, 2022.

[76] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," *Advances in neural information processing systems*, vol. 26, 2013.

[77] L. Liebel and M. Körner, "Auxiliary tasks in multi-task learning," *arXiv preprint arXiv:1805.06334*, 2018.

[78] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark," in *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2017, pp. 3226–3229.

[79] S. P. Mohanty, J. Czakon, K. A. Kaczmarek, A. Pyskir, P. Tarasiewicz, S. Kunwar, J. Rohrbach, D. Luo, M. Prasad, S. Fleer *et al.*, "Deep learning for understanding satellite imagery: An experimental survey. front," *Artif. Intell*, vol. 3, no. 534696, pp. 10–3389, 2020.

[80] Y. K. Adimoolam, B. Chatterjee, C. Poullis, and M. Averkiou, "Efficient deduplication and leakage detection in large scale image datasets with a focus on the crowdai mapping challenge dataset," *arXiv preprint arXiv:2304.02296*, 2023.

[81] E. W. Dijkstra, "A note on two problems in connexion with graphs," in *Edsger Wybe Dijkstra: His Life, Work, and Legacy*, 2022, pp. 287–290.

[82] Y. Wang, Y. Wan, Y. Zhang, B. Zhang, and Z. Gao, "Imbalance knowledge-driven multi-modal network for land-cover semantic segmentation using aerial images and lidar point clouds," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 202, pp. 385–404, 2023.

[83] X. Ma, X. Zhang, M.-O. Pun, and M. Liu, "A multilevel multimodal fusion transformer for remote sensing semantic segmentation," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[84] Y. Du, Q. Sheng, W. Zhang, C. Zhu, J. Li, and B. Wang, "From local context-aware to non-local: A road extraction network via guidance of multi-spectral image," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 203, pp. 230–245, 2023.

[85] X. Zhang, W. Yu, M.-O. Pun, and W. Shi, "Cross-domain landslide mapping from large-scale remote sensing images using prototype-guided domain-aware progressive representation learning," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 197, pp. 1–17, 2023.

[86] X. Zhang, W. Wu, M. Zhang, W. Yu, and P. Ghamisi, "Prototypical unknown-aware multiview consistency learning for open-set cross-domain remote sensing image classification," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[87] Y. Li, H. Wang, Q. Jin, J. Hu, P. Chemerys, Y. Fu, Y. Wang, S. Tulyakov, and J. Ren, "Snapfusion: Text-to-image diffusion model on mobile devices within two seconds," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[88] X. Ma, G. Fang, and X. Wang, "Deepcache: Accelerating diffusion models for free," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 762–15 772.

[89] X. Liu, X. Zhang, J. Ma, J. Peng *et al.*, "Instaflow: One step is enough for high-quality diffusion-based text-to-image generation," in *The Twelfth International Conference on Learning Representations*, 2023.