

Beyond Endpoints: Path-Centric Reasoning for Vectorized Off-Road Network Extraction

Supplementary Material

7. Training the Interactive Model

A core component of our annotation pipeline is the interactive model that generates graph proposals from sparse user clicks. To achieve this, we employ an end-to-end training paradigm that learns to interpret user intent without requiring real-time human interaction during the training phase. The primary challenge is bridging the gap between a static training setup and a dynamic interactive task. We overcome this by introducing a strategy to simulate user prompts, enabling the model to learn the mapping from sparse spatial cues to dense road network structures.

7.1. Simulated Prompt Generation

During training, we simulate user clicks by automatically sampling positive and negative points from the ground-truth annotations.

- **Positive prompts** are sampled from topologically significant locations on the ground-truth graph. Specifically, we identify all vertices that are either junctions (degree > 2) or endpoints (degree $= 1$). A random subset of these keypoints is selected to serve as positive guidance for the model.
- **Negative prompts** are sampled from background regions to prevent spurious graph generation. To ensure these points are not sampled too close to road boundaries, which could create ambiguity for the model, we first establish a buffer zone by dilating the ground-truth road mask with a radius of dist_{\min} . Negative points are then exclusively sampled from the area outside this buffer, representing unambiguous non-road regions.

To enhance the model’s robustness to imprecise clicks, we apply a minor random spatial jitter to the coordinates of all sampled points. This process is detailed in Algorithm 1.

7.2. Prompt Encoding and Feature Fusion

The simulated point prompts are transformed into a spatial representation that can be fused with the image features. Given a set of prompt coordinates $P = \{p_i\}$, we first compute a distance transform map $D \in \mathbb{R}^{H \times W}$, where each pixel (u, v) stores the minimum Euclidean distance to any point in P . This distance map is then processed by a shallow convolutional encoder (two 3×3 conv layers) to produce a multi-channel prompt feature map, F_{prompt} . This feature map is fused with the image features, F_{image} , from the main ViT encoder via element-wise addition. The resulting fused features, $F_{\text{fused}} = F_{\text{image}} + F_{\text{prompt}}$, are then passed to the geometry decoder, effectively guiding the final road

Algorithm 1 Positive and Negative Prompt Generation

Require: GT graph $G = (V, E)$; A dist_{\min} radius; Number of positive prompts N_{pos} and negative prompts N_{neg} .

Ensure: A set of simulated and jittered prompt points P .

```
1: function SIMULATEPROMPTPOINTS( $G, N_{\text{pos}}, N_{\text{neg}}$ )
2:    $V_{\text{key}} \leftarrow \{v \in V \mid \text{degree}(v) \neq 2\}$ 
3:    $P_{\text{pos}} \leftarrow \text{RandomSample}(V_{\text{key}}, N_{\text{pos}})$ 
4:
5:    $M_{\text{road}} \leftarrow \text{RasterizeToMask}(G)$ 
6:    $K \leftarrow \text{CreateStructuringElement}(\text{radius} = \text{dist}_{\min})$ 
7:    $M_{\text{buffer}} \leftarrow \text{MorphologicalDilation}(M_{\text{road}}, K)$ 
8:    $A_{\text{background}} \leftarrow \neg M_{\text{buffer}}$ 
9:    $P_{\text{neg}} \leftarrow \text{RandomSampleFromArea}(A_{\text{background}}, N_{\text{neg}})$ 
10:
11:    $P_{\text{combined}} \leftarrow P_{\text{pos}} \cup P_{\text{neg}}$ 
12:    $P \leftarrow \text{ApplySpatialJitter}(P_{\text{combined}})$ 
13:   return  $P$ 
14: end function
```

graph prediction as shown in the interactive branch of our main pipeline (see Fig. 2 in the main paper).

7.3. Implementation Details

For reproducibility, we specify the key hyperparameters for the prompt simulation. We sample up to $N_{\text{pos}} = 10$ positive prompts, with the number of negative prompts set by a 1:1 ratio. The buffer radius for negative sampling is $\text{dist}_{\min} = 50$ pixels, and a spatial jitter with a standard deviation of 3.0 pixels is applied to all prompt coordinates. The interactive training is conducted on 1024×1024 patches, and critical training parameters such as batch size, learning rate, and optimizer settings are kept identical to those of the baseline automated model. The only additions are the prompt simulation and feature fusion steps; the loss function and optimization process remain unchanged, highlighting the efficiency of our approach.

8. The Interactive Annotation System

To operationalize our interactive framework, we developed a full-featured, web-based annotation tool. This section details its user interface (UI) and the AI-assisted workflow that enables rapid and accurate road network labeling.

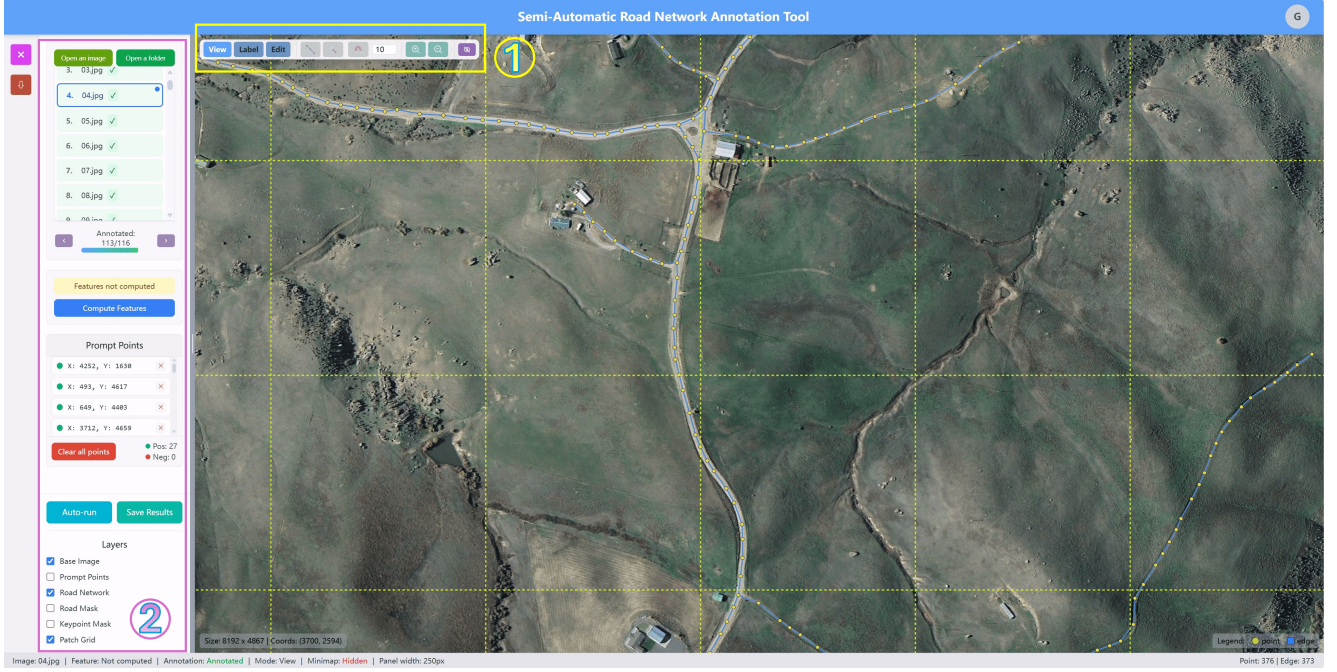


Figure 6. The user interface of our annotation tool, featuring the primary editing toolbar in Region ① and the main control and information panel in Region ②, both designed for an intuitive and efficient user experience.

8.1. User Interface Overview

The annotation tool, shown in Figure 6, is designed for clarity and efficiency. The UI is centered around a main canvas displaying the satellite imagery. The core functionalities are split between two main areas. Region ① contains the primary editing toolbar, which supports three distinct modes: a *View* mode for free panning and zooming; a *Label* mode, which is the core of the interactive process where users can place positive (left-click) and negative (right-click) prompts; and an *Edit* mode for fine-grained manual adjustments to the generated graph, such as moving vertices or adding and deleting edges. Region ② serves as the main control and information panel, handling data I/O, providing options for pre-computing image features to accelerate inference, listing prompt point coordinates, managing display layers, and housing the main “Auto-run” and “Save Results” buttons. Additionally, a persistent status bar provides helpful auxiliary information, such as full image dimensions and real-time cursor coordinates, to aid annotators with spatial awareness and precise editing.

8.2. Large-Scale Image Inference

The Challenge and Our Strategy. A fundamental challenge is applying our model, which processes 1024×1024 inputs, to high-resolution satellite imagery (e.g., $8K \times 4K$). A naive approach of processing every patch is computationally wasteful, especially in off-road scenes where roads

are sparse. To overcome this, we developed a prompt-driven, overlapping patch-based inference pipeline. This strategy ensures that computation is focused exclusively on the user’s regions of interest, enabling a smooth and highly efficient interactive experience. The process, detailed in Algorithm 2, consists of three key stages.

Stage 1: Prompt-Guided Mask Generation. The large input image is first partitioned into a grid of 1024×1024 patches with a significant overlap (e.g., 256 pixels) to prevent artifacts at the boundaries. The inference process is initiated by and centered around the user’s sparse prompts. When the user clicks “Auto-run”, the system first identifies the minimal set of 1024×1024 patches required to cover all prompt points. Inference is run only on this active subset of patches to produce local road and keypoint probability maps. These generated local maps are then seamlessly stitched together into a unified global map for the relevant region. In overlapping areas, pixel values are blended via a weighted average to ensure smooth transitions.

Stage 2: Vertex Extraction and Topological Inference. From the fused global keypoint map, a consistent set of candidate vertices is extracted via Non-Maximum Suppression (NMS). With these vertices established, the system then determines their connectivity. To do this efficiently, we revisit each of the previously activated patches. Within a given

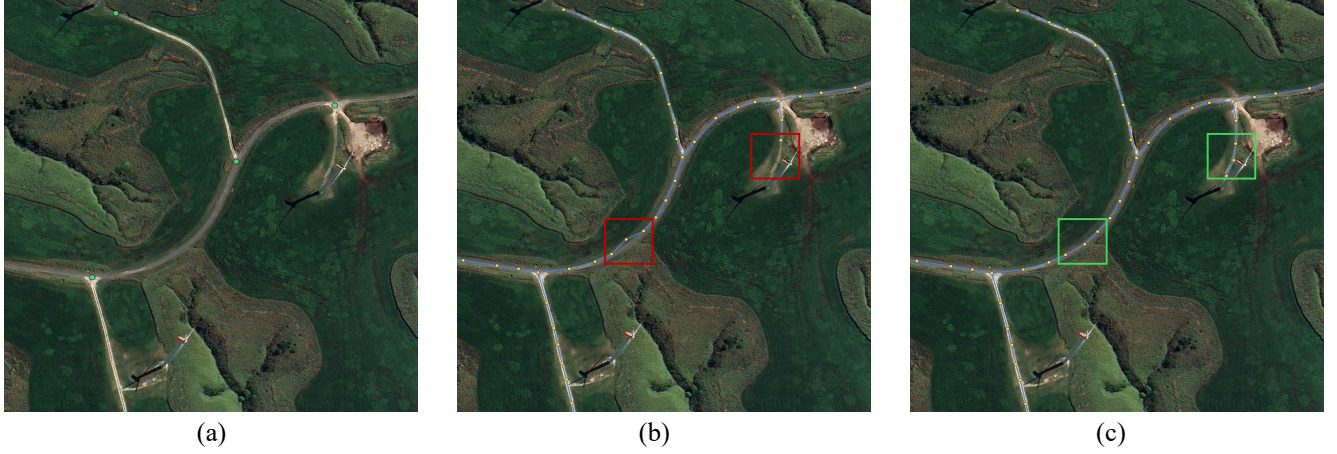


Figure 7. The AI-assisted “Prompt-Propose-Refine” workflow. (a) An annotator provides sparse prompts at key locations. (b) The model generates a high-quality proposal, which may contain minor errors highlighted by red boxes. (c) The user quickly refines these areas, with the corrected sections shown in green boxes.

patch, we consider only the subset of global vertices that fall within its boundaries and use our MaGTopoNet module to predict the probability of edges between them based on the local image context.

Stage 3: Global Edge Aggregation and Final Graph. Since patches overlap, a single candidate edge may be evaluated independently in multiple patches, providing a robust ensembling opportunity. In this final stage, we aggregate all connectivity predictions from across the active patches. The scores for each unique edge are averaged, and an edge is included in the final graph only if its average score exceeds a confidence threshold. This ensures the final road network is topologically coherent, resolving local ambiguities and producing a single, unified graph for the user to refine.

8.3. The “Prompt-Propose-Refine” Workflow

Our system transforms the laborious task of manual road tracing into a highly efficient, human-in-the-loop validation process, which we term the “Prompt-Propose-Refine” workflow. As illustrated in Figure 7, this process unfolds in three intuitive steps.

- **(a) Prompt:** The annotator begins by inspecting the image and placing a sparse set of positive and negative prompts at key topological locations, such as junctions, endpoints, or ambiguous areas.
- **(b) Propose:** After placing the prompts, the annotator clicks “Auto-run”. The system feeds these prompts to our interactive MaGRoad model, which generates a high-quality road graph proposal in real-time. This proposal serves as a strong baseline, often capturing the majority of the road network correctly.

Algorithm 2 Prompt-Driven Large-Scale Inference

Require: Large image I , User prompts P .

Ensure: Final graph $G = (V, E)$.

```

1: function INFERFROMPROMPTS( $I, P$ )
2:    $A_{\text{patches}} \leftarrow \text{IdentifyActivePatches}(P)$ 
3:    $M_{\text{local}} \leftarrow \{\}$ 
4:   for patch in  $A_{\text{patches}}$  do
5:      $M_{\text{local}}[\text{patch}] \leftarrow \text{Model.infer}(I, \text{patch})$ 
6:   end for
7:    $M_{\text{global}} \leftarrow \text{FuseMasks}(M_{\text{local}})$ 
8:    $V \leftarrow \text{ExtractVerticesFromMask}(M_{\text{global}})$ 
9:    $S_{\text{edge}} \leftarrow \{\}$ 
10:  for patch in  $A_{\text{patches}}$  do
11:     $V_{\text{patch}} \leftarrow \text{GetVerticesInPatch}(V, \text{patch})$ 
12:     $S_{\text{patch}} \leftarrow \text{MaGTopoNet}(V_{\text{patch}}, M_{\text{local}}[\text{patch}])$ 
13:     $\text{UpdateEdgeScores}(S_{\text{edge}}, S_{\text{patch}})$ 
14:  end for
15:   $E \leftarrow \text{AggregateAndThresholdEdges}(S_{\text{edge}})$ 
16:  return  $G = (V, E)$ 
17: end function

```

- **(c) Refine:** The annotator’s task is then reduced to curation. They examine the proposal for inaccuracies highlighted by red boxes. By switching to *Edit* mode, they can perform a range of quick corrections, such as repositioning vertices and adding missed connections, to achieve the final, accurate graph shown in the green boxes.

This synergistic “Prompt-Propose-Refine” paradigm combines the pattern recognition strength of the deep model with the nuanced judgment of a human annotator, achieving both high efficiency and accuracy.

9. Dataset Organization

9.1. Data Partitioning Strategy

Constructing a high-quality vectorized dataset from large-scale satellite imagery requires a well-designed processing pipeline. Raw gigapixel images (for example, $8k \times 4k$) are too large for direct training, so they are first divided into manageable 1024×1024 patches. To ensure that each patch contains sufficient information and that the overall dataset reflects diverse topological patterns, we adopt a “Generate–Filter–Select” strategy. This procedure converts raw imagery into a curated collection of samples while reducing repetitive content, as summarized in Algorithm 3.

Candidate Generation and Graph Cropping. We first employ a sliding window approach to generate a comprehensive pool of candidate patches using two distinct strategies. The primary set A consists of non-overlapping patches sampled on a strict grid with stride 1024 to ensure basic coverage. A supplementary set B is generated using a dense, overlapping sliding window with stride 256 to capture diverse road contexts and shift-variant topologies that might be split across boundaries in the primary grid. A critical challenge here is maintaining graph validity at patch boundaries. We utilize a robust graph cropping algorithm that computes precise geometric intersections between road edges and patch borders. This ensures that all road segments within a patch are properly terminated or connected, preventing invalid topology such as dangling edges or isolated nodes outside the field of view.

Density-Based Filtering. In off-road environments, vast regions may contain no road networks. To maintain training efficiency, we filter candidates based on road length density. We compute the total length of road segments within each patch and normalize it by the patch area. Candidates falling below a predefined density threshold τ_{density} are identified as empty background and discarded. This step ensures that the model focuses on regions with valid learning signals.

Topology-Aware Diversity Selection. A common issue in sliding-window datasets is the inclusion of highly repetitive samples (e.g., identical straight roads shifted by a few pixels). To address this, we introduce a topology-aware selection mechanism using the Weisfeiler-Lehman (WL) Graph Kernel. We first retain all valid patches from the primary Set A. Then, we iteratively evaluate candidates from Set B. For each candidate, we extract its graph topology and compute its WL similarity score against spatially neighboring patches that have already been selected. A candidate is added to the final dataset only if its maximum similarity score is below a threshold τ_{sim} . This strategy explicitly encourages the inclusion of topologically distinct samples

Algorithm 3 Topology-Aware Data Partitioning

Require: Large images \mathcal{I} , ground-truth graphs \mathcal{G} ; density threshold τ_{density} , similarity threshold τ_{sim} .

Ensure: Final dataset \mathcal{D} .

```

1: function GENERATEANDSELECT( $\mathcal{I}, \mathcal{G}$ )
2:    $\mathcal{D} \leftarrow \emptyset$ 
3:   for each ( $I, G$ ) in ( $\mathcal{I}, \mathcal{G}$ ) do
4:      $\triangleright$  Step 1: Candidate Generation
5:      $S_A \leftarrow \text{SLIDINGWINDOW}(I, G, \text{stride} = 1024)$ 
6:      $S_B \leftarrow \text{SLIDINGWINDOW}(I, G, \text{stride} = 256)$ 
7:      $\triangleright$  Step 2: Density Filtering
8:      $S_A \leftarrow \{p \in S_A \mid \text{DENSITY}(p) \geq \tau_{\text{density}}\}$ 
9:      $S_B \leftarrow \{p \in S_B \mid \text{DENSITY}(p) \geq \tau_{\text{density}}\}$ 
10:     $\triangleright$  Step 3: Diversity Selection
11:     $\mathcal{D}_{\text{local}} \leftarrow S_A$ 
12:    SORTBYDENSITY( $S_B$ )
13:    for each patch  $p$  in  $S_B$  do
14:       $N \leftarrow \text{GETSPATIALNEIGHBORS}(p, \mathcal{D}_{\text{local}})$ 
15:       $\text{sim}_{\text{max}} \leftarrow \max_{n \in N} \text{WLSIM}(p, G, n, G)$ 
16:      if  $\text{sim}_{\text{max}} < \tau_{\text{sim}}$  then
17:         $\mathcal{D}_{\text{local}} \leftarrow \mathcal{D}_{\text{local}} \cup \{p\}$ 
18:      end if
19:    end for
20:     $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_{\text{local}}$ 
21:  end for
22:  return  $\mathcal{D}$ 
23: end function

```

(such as complex junctions or winding paths) while suppressing redundant simple structures.

9.2. Dataset Statistics

After applying our partitioning and selection strategy, the final WildRoad dataset comprises a total of 9,274 curated patches. Table 6 details the distribution across the training, validation, and test sets. The training set contains 6,448 patches with over 4,000 km of road network and more than 11,000 intersections, providing a rich source of topological variety for model learning. The validation and test sets are similarly structured, ensuring a robust evaluation of generalization capability across diverse off-road scenarios.

Table 6. Dataset statistics for road network analysis.

Dataset	Files	Length (km)	Intersections	Endpoints
Train	6,448	4,104.99	11,172	35,530
Val	1,493	951.61	2,573	8,298
Test	1,333	810.95	2,180	6,941