# Sat2Graph: Road Graph Extraction through Graph-Tensor Encoding

Songtao He[1], Favyen Bastani[1], Satvat Jagwani[1], Mohammad Alizadeh[1], Hari Balakrishnan[1], Sanjay Chawla[2], Mohamed M. Elshrif[2], Samuel Madden[1], and Mohammad Amin Sadeghi[3]

[1] Massachusetts Institute of Technology
{songtao, favyen, satvat, alizadeh, hari, madden}@csail.mit.edu
[2] Qatar Computing Research Institute
schawla@hbku.edu.qa, melshrif77@gmail.com
[3] University of Tehran
m.a.sadeghi@gmail.com

**Abstract.** Inferring road graphs from satellite imagery is a challenging computer vision task. Prior solutions fall into two categories: (1) pixel-wise segmentation-based approaches, which predict whether each pixel is on a road, and (2) graph-based approaches, which predict the road graph iteratively. We find that these two approaches have complementary strengths while suffering from their own inherent limitations.

In this paper, we propose a new method, Sat2Graph, which combines the advantages of the two prior categories into a unified framework. The key idea in Sat2Graph is a novel encoding scheme, *graph-tensor encoding* (GTE), which encodes the road graph into a tensor representation. GTE makes it possible to train a simple, non-recurrent, supervised model to predict a rich set of features that capture the graph structure directly from an image. We evaluate Sat2Graph using two large datasets. We find that Sat2Graph surpasses prior methods on two widely used metrics, TOPO and APLS. Furthermore, whereas prior work only infers planar road graphs, our approach is capable of inferring stacked roads (e.g., overpasses), and does so robustly.

## 1 Introduction

Accurate and up-to-date road maps are critical in many applications, from navigation to self-driving vehicles. However, creating and maintaining digital maps is expensive and involves tedious manual labor. In response, automated solutions have been proposed to automatically infer road maps from different sources of data, including GPS tracks, aerial imagery, and satellite imagery. In this paper, we focus on extracting road network graphs from satellite imagery.

Although many techniques have been proposed [2, 3, 6, 9, 10, 20–22, 25, 32, 35, 36], extracting road networks from satellite imagery is still a challenging computer vision task due to the complexity and diversity of the road networks. Prior

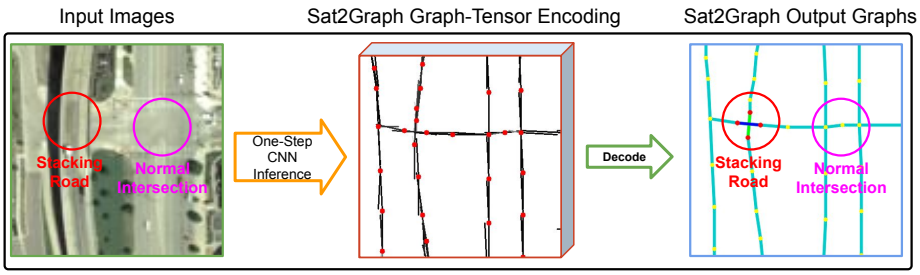


**Fig. 1.** Highlight of Sat2Graph.

solutions fall into two categories: pixel-wise segmentation-based approaches and graph-based approaches. Segmentation-based approaches assign a *roadness* score to each pixel in the satellite imagery. Then, they extract the road network graph using heuristic approaches. Here, the road segmentation acts as the intermediate representation of the road network graph. In contrast, graph-based approaches construct a road network graph directly from satellite imagery. Recently, Bastani *et al.* [2], as well as several follow-up works [10,21], utilize graph-based solutions that iteratively add vertices and edges to the partially constructed graph.

We observe that the approaches in these two categories often tradeoff with each other. Segmentation-based approaches typically have a wider receptive field but rely on an intermediate non-graph representation and a post-processing heuristic (e.g., morphological thinning and line following) to extract road network graphs from this intermediate representation. The usage of the intermediate non-graph representation limits the segmentation-based approaches, and they often produce noisy and lower precision road networks compared with the graph-based methods as a result. To encourage the neural network model to focus more on the graph structure of road networks, recent work [3] proposes to train the road segmentation model jointly with road directions, and the approach achieves better road connectivity through this joint training strategy. However, a postprocessing heuristic is still needed.

In contrast, graph-based approaches [2, 10, 21] learn the graph structure directly. As a result, graph-based approaches yield road network graphs with better road connectivity compared with the original segmentation-based approach [2]. However, the graph generation process is often iterative, resulting in a neural network model that focuses more on local information rather than global information. To take more global information into account, recent work [10, 21] proposes to improve the graph-based approaches with a sequential generative model, resulting in better performance compared with other state-of-art approaches.

Recent advancements [3,10,21] in segmentation-based approaches and graph-based approaches respectively are primarily focused on overcoming the inherent limitations of their baseline approaches, which are exactly from the same aspects that the methods in the competing baseline approach (i.e., from the other cate-

gory) claim as advantages. Based on this observation, a natural question to ask is if it is possible to combine the segmentation-based approach and the graph-based approach into one unified approach that can benefit from the advantages of both?

Our answer to this question is a new road network extraction approach, Sat2Graph, which combines the inherent advantages of segmentation-based approaches and graph-based approaches into one simple, unified framework. To do this, we design a novel encoding scheme, *graph-tensor encoding* (GTE), to encode the road network graph into a tensor representation, making it possible to train a simple, non-recurrent, supervised model that predicts graph structures holistically from the input image.

In addition to the tensor-based network encoding, this paper makes two contributions:

1. Sat2Graph surpasses state-of-the-art approaches in a widely used topology-similarity metric at all precision-recall trade-off positions in an evaluation over a large city-scale dataset covering 720 $km^2$ area in 20 U.S. cities and the popular SpaceNet roads dataset [30].
2. Sat2Graph can naturally infer stacked roads, which prior approaches don't handle.

## 2 Related work

**Traditional Approaches.** Extracting road networks from satellite imagery has long history [14, 31]. Traditional approaches generally use heuristics and probabilistic models to infer road networks from imagery. For examples, Hinz *et al.* [20] propose an approach to create road networks through a complicated road model that is built using detailed knowledge about roads and the environmental context, such as the nearby buildings, vehicles and so on. Wegner *et al.* [32] propose to model the road network with higher-order conditional random fields (CRFs). They first segment the aerial images into super-pixels, then they connect these super-pixels based on the CRF model.

**Segmentation-Based Approaches.** With the increasing popularity of deep learning, researchers have used convolutional neural networks (CNN) to extract road network from satellite imagery [3, 6, 9, 22, 35, 36]. For example, Cheng *et al.* [9] use an end-to-end cascaded CNN to extract road segmentation from satellite imagery. They apply a binary threshold to the road segmentation and use morphological thinning to extract the road center-lines. Then, a road network graph is produced through tracing the single-pixel-width road center-lines. Many other segmentation-based approaches proposed different improvements upon this basic graph extraction pipeline, including improved CNN backbones [6, 36], improved post-processing strategy [22], improved loss functions [22, 25], incorporating GAN [11, 28, 34], and joint training [3].

In contrast with existing segmentation-based approaches, Sat2Graph does not rely on the road segmentation as intermediate representation and learns the

graph structure directly.

**Graph-Based Approaches.** Graph-based approaches construct a road network graph directly from satellite imagery. Recently, Bastani *et al.* [2] proposed RoadTracer, a graph-based approach to generate road network in an iterative way. The algorithm starts from a known location on the road map. Then, at each iteration, the algorithm uses a deep neural network to predict the next location to visit along the road through looking at the surrounding satellite imagery of the current location. Recent works [10, 21] advanced the graph-based approach through applying sequential generative models (RNN) to generate road network iteratively. The usage of sequential models allows the graph generation model to take more context information into account compared with RoadTracer [2].

In contrast with existing graph-based approaches, Sat2Graph generates the road graphs in one shot (holistic). This allows Sat2Graph to easily capture the global information and make better coordination of vertex placement. The non-recurrent property of Sat2Graph also makes it easy to train and easy to extend (e.g., combine Sat2Graph with GAN). We think this simplicity of Sat2Graph is another advantage over other solutions.

**Using Other Data Sources and Other Digital Map Inference Tasks.** Extracting road networks from other data sources has also been extensively studied, e.g., using GPS trajectories collected from moving vehicles [1,5,8,12,13, 18, 29]. Besides road topology inference, satellite imagery also enables inference of different map attributes, including high-definition road details [19,23,24], road safety [26] and road quality [7].

## 3 Sat2Graph

In this section, we present the details of our proposed approach - Sat2Graph. Sat2Graph relies on a novel encoding scheme that can encode the road network graph into a three-dimensional tensor. We call this encoding scheme Graph-Tensor Encoding (GTE). This graph-tensor encoding scheme allows us to train a simple, non-recurrent, neural network model to directly map the input satellite imagery into the road network graph (i.e., edges and vertices). As noted in the introduction, this graph construction strategy combines the advantages of segmentation-based and graph-based approaches.

### 3.1 Graph-Tensor Encoding (GTE)

We show our graph-tensor encoding (GTE) scheme in Figure 2(a). For a road network graph $G = \{V, E\}$ that covers a $W$ meters by $H$ meters region, GTE uses a $\frac{W}{\lambda} \times \frac{H}{\lambda} \times (1 + 3 \cdot D_{max})$ 3D-tensor (denoted as $T$) to store the encoding of the graph. Here, the $\lambda$ is the spatial resolution, i.e., one meter, which restricts the encoded graph in a way that no two vertices can be co-located within a $\lambda \times \lambda$ grid, and $D_{max}$ is the maximum edges that can be encoded at each $\lambda \times \lambda$ grid.

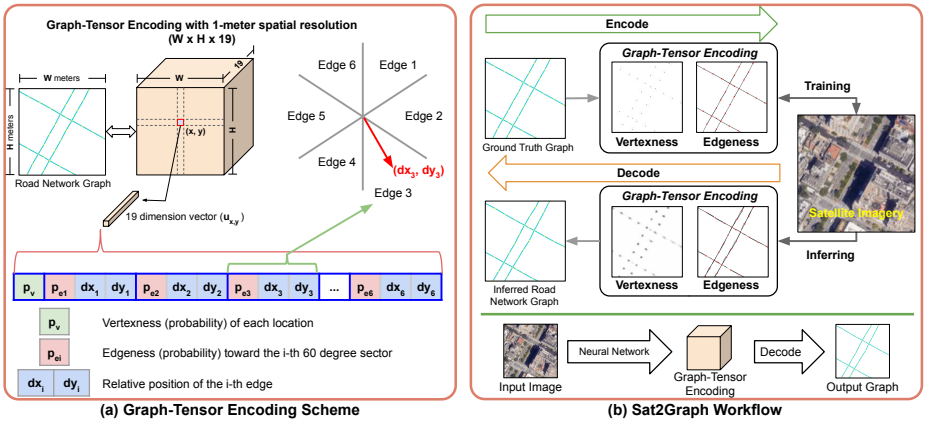


**Fig. 2.** Graph-Tensor Encoding and Sat2Graph workflow.

The first two dimensions of $T$ correspond to the two spatial axes in the 2D plane. We use the vector at each spatial location $u_{x,y} = [T_{x,y,1}, T_{x,y,2}, ..., T_{x,y,(1+3\cdot D_{max})}]^T$ to encode the graph information. As shown in Figure 2(a), the vector $u_{x,y}$ has $(1 + 3 \cdot D_{max})$ elements. Its first element $p_v \in [0, 1]$ (vertexness) encodes the probability of having a vertex at position $(x, y)$. Following the first element are $D_{max}$ 3-element groups, each of which encodes the information of a potential outgoing edge from position $(x, y)$. For the $i$-th 3-element group, its first element $p_{e_i} \in [0, 1]$ (edgeness) encodes the probability of having an outgoing edge toward $(dx_i, dy_i)$, i.e., an edge pointing from $(x, y)$ to $(x + dx_i, y + dy_i)$. Here, we set $D_{max}$ to six as we find that vertices with degree greater than six are very rare in road network graphs.

To reduce the number of possible different isomorphic encodings of the same input graph, GTE only uses the $i$-th 3-element group to encode edges pointing toward a $\frac{360}{D_{max}}$-degree sector from $(i-1) \cdot \frac{360}{D_{max}}$ degrees to $i \cdot \frac{360}{D_{max}}$ degrees. We show this restriction and an example edge (in red color) in Figure 2(a). This strategy imposes a new restriction on the encoded graphs – for each vertex in the encoded graph, there can only be at most one outgoing edge toward each $\frac{360}{D_{max}}$-degree sector. However, we find this restriction does not impact the representation ability of GTE for most road graphs. This is because the graphs encoded by GTE are *undirected*. We defer the discussion on this in Section 5.

**Encode** Encoding a road network graph into GTE is straightforward. For road network extraction application, the encoding algorithm first interpolates the segment of straight road in the road network graph. It selects the minimum number of evenly spaced intermediate points so that the distance between consecutive points is under $d$ meters. This interpolation strategy regulates the length of the edge vector in GTE, making the training process stable. Here, a small $d$ value,

e.g., $d < 5$, converts GTE back to the road segmentation, making GTE unable to represent stacking roads. A very large $d$ value, e.g. $d = 50$, makes the GTE hard to approximate curvy roads. For these reasons, we think a $d$ value between 15 to 25 can work the best. In our setup, we set $d$ to 20.

For stacked roads, the interpolation may produce vertices belonging to two overlapped road segments at the same position. When this happens, we use an iterative conflict-resolution algorithm to shift the positions of the endpoint vertices of the two edges. The goal is to make sure the distance between any two vertices (from the two overlapping edges) is greater than 5 meters. During training, this conflict-resolution pre-processing also yields more consistent supervision signal for stacked roads - overlapped edges tend to always cross near the middle of each edge. After this step, the encoding algorithm maps each of the vertices to the 3D-tensor $T$ following the scheme shown in Figure 2(a). For example, the algorithm sets the vertexness ($p_v$) of $u_{x,y}$ to 1 when there is a vertex at position $(x, y)$, otherwise the vertexness is set to 0.

**Decode** GTE's Decoding algorithm converts the predicted GTE (often noisy) of a graph back to the regular graph format (G = {V,E}). The decoding algorithm consists of two steps, (1) vertex extraction and, (2) edge connection. As both the vertexness predictions and edgeness predictions are real numbers between 0 and 1, we only consider vertices and edges with probability greater than a threshold (denoted as $p_{thr}$).

In the vertex extraction step, the decoding algorithm extracts the potential vertices through localizing the local maximas of the vertexness map (we show an example of this in Figure 2(b)). The algorithm only considers the local maximas with vertexness greater than $p_{thr}$.

In the edge connection step, for each candidate vertex $v \in V$, the decoding algorithm connects its outgoing edges to other vertices. For the $i$-th edge of vertex $v \in V$, the algorithm computes its distance to all nearby vertices $u$ through the following distance function,

$$\begin{aligned} d(v, i, u) =& ||(v_x + dx_i, v_y + dy_i) - (u_x, u_y)|| \\ &+ w \cdot cos_{dist}((dx_i, dy_i), (u_x - v_x, u_y - v_y)) \end{aligned} \tag{1}$$

, where $cos_{dist}(v_1, v_2)$ is the cosine distance of the two vectors, and $w$ is the weight of the cosine distance in the distance function. Here, we set $w$ to a large number, i.e., 100, to avoid incorrect connections. After computing this distance, the decoding algorithm picks up a vertex $u'$ that minimizes the distance function $d(v, i, u)$, and adds an edge between $v$ and $u'$. We set a maximum distance threshold, i.e., 15 meters, to avoid incorrect edges being added to the graph when there are no good candidate vertices nearby.

### 3.2 Training Sat2Graph

We use cross-entropy loss (denoted as $\mathcal{L}_{CE}$) and $L_2$-loss to train Sat2Graph. The cross-entropy loss is applied to vertexness channel ($p_v$) and edgeness channels

($p_{e_i}$ $i \in \{1, 2, ..., D_{max}\}$), and the $L_2$-loss is applied to the edge vector channels ($(dx_i, dy_i)$ $i \in \{1, 2, ..., D_{max}\}$). GTE is inconsistent along long road segments. In this case, the same road structure can be mapped to different ground truth labels in GTE representation. Because of this inconsistency, we only compute the losses for edgeness and edge vectors at position $(x, y)$ when there is a vertex at position $(x, y)$ in the ground truth. We show the overall loss function below ($\hat{T}, \hat{p_v}, \hat{p_{e_i}}, \hat{d_{x_i}}, \hat{d_{y_i}}$ are from ground truth),

$$\mathcal{L}(T, \hat{T}) = \sum_{(x,y)\in[1..W]\times[1..H]} \Bigg( \mathcal{L}_{CE}(p_v, \hat{p_v}) + \hat{T}_{x,y,1} \cdot \Big( \sum_{i=1}^{D_{max}} \big( \mathcal{L}_{CE}(p_{e_i}, \hat{p_{e_i}}) + \mathcal{L}_2((dx_i, dy_i), (\hat{dx_i}, \hat{dy_i})) \big) \Big) \Bigg) \tag{2}$$

In Figure 2(b), we show the training and inferring workflows of Sat2Graph. Sat2Graph is agnostic to the CNN backbones. In this paper, we choose to use the Deep Layer Aggregation (DLA) [33] segmentation architecture as our CNN backbone. We use residual blocks [17] for the aggregation function in DLA. The feasibility of training Sat2Graph with supervised learning is counter-intuitive because of the GTE's inconsistency. We defer the discussion of this to Section 5.

## 4 Evaluation

We now present experimental results comparing Sat2Graph to several state-of-the-art road-network generation systems.

### 4.1 Datasets

We conduct our evaluation on two datasets, one is a large city-scale dataset and the other is the popular SpaceNet roads dataset [30].

**City-Scale Dataset.** Our city-scale dataset covers 720 $km^2$ area in 20 U.S. cities. We collect road network data from OpenStreetMap [16] as ground truth and the corresponding satellite imagery through Google static map API [15]. The spatial resolution of the satellite imagery is set to one meter per pixel. This dataset enables us to evaluate the performance of different approaches at city scale, e.g., evaluating the quality of the shortest path crossing the entire downtown of a city on the inferred road graphs.

The dataset is organized as 180 tiles; each tile is a 2 km by 2 km square region. We randomly choose 15% (27 tiles) of them as a testing dataset and 5% (9 tiles) of them as a validation dataset. The remaining 80% (144 tiles) are used as training dataset.

**SpaceNet Roads Dataset.** Another dataset we used is the SpaceNet roads Dataset [30]. Because the ground truth of the testing data in the SpaceNet dataset is not public, we randomly split the 2549 tiles (non-empty) of the original training dataset into training(80%), testing(15%) and validating(5%) datasets.

Each tile is a 0.4 km by 0.4 km square. Similar to the city-scale dataset, we resize the spatial resolution of the satellite imagery to one meter per pixel.

### 4.2 Baselines

We compare Sat2Graph with four different segmentation-based approaches and one graph-based approach.

**Segmentation-Based Approaches.** We use four different segmentation-based approaches as baselines.

1. *Seg-UNet:* Seg-UNet uses a simple U-Net [27] backbone to produce road segmentation from satellite imagery. The model is trained with cross-entropy loss. This scheme acts as the naive baseline as it is the most straightforward solution for road extraction.
2. *Seg-DRM [22](ICCV-17):* Seg-DRM uses a stronger CNN backbone which contains 55 ResNet [17] layers to improve the road extraction performance. Meanwhile, Seg-DRM proposes to train the road segmentation model with soft-IoU loss to achieve better performance. However, we find training the Seg-DRM model with cross-entropy loss yields much better performance in terms of topology correctness. Thus, in our evaluation, we train the Seg-DRM model with cross-entropy loss.
3. *Seg-Orientation [3](ICCV-19):* Seg-Orientation is a recent state-or-the-art approach which proposes to improve the road connectivity by joint learning of road orientation and road segmentation. Similar to Seg-DRM, we show the results of Seg-Orientation trained with cross-entropy loss as we find it performs better compared with soft-IoU loss.
4. *Seg-DLA:* Seg-DLA is our enhanced segmentation-based approach which uses the same CNN backbone as our Sat2Graph model. Seg-DLA, together with Seg-UNet, act as the baselines of an ablation study of Sat2Graph.

**Graph-Based Approaches.** For graph-based approaches, we compare our Sat2Graph solution with RoadTracer [2](CVPR-18) by applying their code on our dataset. During inference, we use peaks in the segmentation output as starting locations for RoadTracer's iterative search.

### 4.3 Implementation Details

**Data Augmentation:** For all models in our evaluation, we augment the training dataset with random image brightness, hue and color temperature, random rotation of the tiles, and random masks on the satellite imagery.

**Training:** We implemented both Sat2Graph and baseline segmentation approaches using Tensorflow. We train the model on a V100 GPU for 300k iterations (about 120 epochs) with a learning rate starting from 0.001 and decreasing by 2x every 50k iterations. We train all models with the same receptive field,

i.e., 352 by 352. We evaluate the performance on the validation dataset for each model every 5k iterations during training, and pick up the best model on the validation dataset as the converged model for each approach to avoid overfitting.

### 4.4 Evaluation Metrics

In the evaluation, we focus on the topology correctness of the inferred road graph rather than edge-wise correctness. This is because the topology correctness is often crucial in many real-world applications. For example, in navigation applications, a small missing road edge in the road graph could make two regions disconnected. This small missing road segment is a small error in terms of edge-wise correctness but a huge error in terms of topology correctness.

We evaluate the topology correctness of the inferred road graphs through two metrics, TOPO [4] and APLS [30]. Here, we describe the high level idea of these two metrics. Please refer to [4,30] for more details about these two metrics.

**TOPO metric:** TOPO metric measures the similarity of sub-graphs sampled on the ground truth graph and the inferred graph from a seed location. The seed location is matched to the closest seed node on each graph. Here, given a seed node on a graph, the sub-graph contains all the nodes such that their distances (on the graph) to the seed node are less than a threshold, e.g., 300 meters. For each seed location, the similarity between two sampled sub-graphs is quantified as precision, recall and $F_1$-score. The metric reports the average precision, recall and $F_1$-score over randomly sampled seed locations over the entire region.

The TOPO metric has different implementations. We implement the TOPO metric in a very strict way following the description in [18]. This strict implementation allows the metric to penalize detailed topology errors.

**APLS metric:** APLS measures the quality of the shortest paths between two locations on the graph. For example, suppose the shortest path between two locations on the ground truth map is 200 meters, but the shortest path between the same two locations on the inferred map is 20 meters (a wrong shortcut), or 500 meters, or doesn't exist. In these cases, the APLS metric yields a very low score, even though there might be only one incorrect edge on the inferred graph.

### 4.5 Quantitative Evaluation

**Overall Quality.** Each of the approaches we evaluated has one major hyper-parameter, which is often a probability threshold, that allows us to make different precision-recall trade-offs. We change this parameter for each approach to plot an precision-recall curve. We show the precision-recall curves for different approaches in Figure 3. This precision-recall curve allows us to see the full picture of the capability of each approach. We also show the best achievable TOPO $F_1$-score and APLS score of each approach in Table 1 for reference.

From Figure 3, we find an approach may not always better than another approach at different precision-recall position (TOPO metric). For examples,

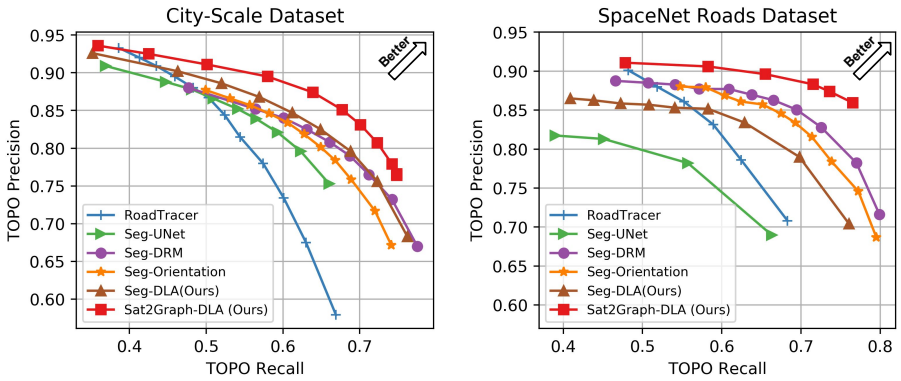


**Fig. 3.** TOPO metric precision-recall trade-off curves

the graph-based approach RoadTracer performs better than others when the precision is high, whereas the segmentation-based approach DeepRoadMapper performs better when the recall is high.

Meanwhile, we find an approach may not always better than another approach on both TOPO and APLS. For example, in Table 1, RoadTracer has the best APLS score but the worst TOPO $F_1$-score in the five baselines on the city-scale dataset. This is because RoadTracer is good at coarse-grained road connectivity and the precision of inferred road graphs rather than recall. For example, in Figure 4(a), RoadTracer is better compared with Seg-DRM and Seg-Orientation in terms of road connectivity when the satellite imagery is full of shadow.

In contrast, Sat2Graph surpasses all other approaches on APLS metric and at all TOPO precision-recall positions – for a given precision, Sat2Graph always has the best recall; and for a given recall, Sat2Graph always has the best precision. We think this is because Sat2Graph's graph-tensor encoding takes advantages from both the segmentation-based approaches and graph-based approaches, and allows Sat2Graph to infer stacking roads that none of the other approaches can handle. As an ablation study, we compare Sat2Graph-DLA with Seg-DLA (Seg-DLA uses the same CNN backbone as Sat2Graph-DLA). We find the superiority of Sat2Graph comes from the graph-tensor encoding rather than the stronger CNN backbone.

**Benefit from GTE.** In addition to the results shown in Table 1, we show the results of using GTE with other backbones. On our city-wide dataset, we find GTE can improve the TOPO $F_1$-score from 70.36% to 76.40% with the U-Net backbone and from 73.80% to 74.66% with the Seg-DRM backbone. Here, the improvement on Seg-DRM backbone is minor because Seg-DRM backbone has a very shallow decoder.

**Sensitivity on $w$.** In our decoding algorithm, we have a hyper-parameter $w$ which is the weight of the cosine distance term in equation 1. In Table 2, we

| Method | City-Scale Dataset | | | | SpaceNet Roads Dataset | | | |
|---|---|---|---|---|---|---|---|---|
| | Prec. | Rec. | $F_1$ | APLS | Prec. | Rec. | $F_1$ | APLS |
| RoadTracer [2](CVPR-18) | 78.00 | 57.44 | 66.16 | 57.29 | 78.61 | 62.45 | 69.60 | 56.03 |
| Seg-UNet | 75.34 | 65.99 | 70.36 | 52.50 | 68.96 | 66.32 | 67.61 | 53.77 |
| Seg-DRM [22](ICCV-17) | 76.54 | 71.25 | 73.80 | 54.32 | 82.79 | 72.56 | 77.34 | 62.26 |
| Seg-Orientation [3](ICCV-19) | 75.83 | 68.90 | 72.20 | 55.34 | 81.56 | 71.38 | 76.13 | 58.82 |
| Seg-DLA(ours) | 75.59 | 72.26 | 73.89 | 57.22 | 78.99 | 69.80 | 74.11 | 56.36 |
| Sat2Graph-DLA(ours) | 80.70 | 72.28 | **76.26** | **63.14** | 85.93 | 76.55 | **80.97** | **64.43** |

**Table 1.** Comparison of the *best achievable* TOPO $F_1$-score and APLS score. We show the best TOPO $F_1$-score's corresponding precision and recall just for reference not for comparison. (All the values in this table are percentages)

show how this parameter impacts the TOPO $F_1$-score on our city-wide dataset. We find the performance is robust to w - the $F_1$-scores are all greater than 76.2% with w in the range from 5 to 100.

| Value of $w$ | 1 | 5 | 10 | 25 | 75 | 100 | 150 |
|---|---|---|---|---|---|---|---|
| $F_1$-score | 75.87% | 76.28% | 76.62% | 76.72% | 76.55% | 76.26% | 75.68% |

**Table 2.** TOPO $F_1$ scores on our city-wide dataset with different $w$ values.

**Vertex threshold and edge threshold.** In our basic setup, we set the vertex threshold and the edge threshold of Sat2Graph to the same value. However, we can also use independent probability thresholds for vertices and edges. We evaluate this by choosing a fixed point and vary one probability threshold at a time. We find the vertex threshold dominates the performance and using a higher edge probability threshold (compared with the vertex probability) is helpful to achieve better performance.

**Stacking Road.** We evaluate the quality of the stacking road by matching the overpass/underpass crossing points between the ground truth graphs and the proposed graphs. In this evaluation, we find our approach has a precision of 83.11% (number of correct crossing points over the number of all proposed crossing points) and a recall of 49.81% (number of correct crossing points over the number of all ground-truth crossing points) on stacked roads. In fact only 0.37% of intersections are incorrectly predicted as overpasses/underpasses (false-positive rate). We find some small roads under wide highway roads are missing entirely. We think this is the reason for the low recall.

### 4.6 Qualitative Evaluation

**Regular Urban Areas.** In the regular urban areas (Figure 4), we find the existing segmentation-based approach with a strong CNN backbone (Seg-DLA) and better data augmentation techniques has already been able to achieve decent results in terms of both precision and recall, even if the satellite imagery is

full of shadows and occlusions. Compared with Sat2Graph, the most apparent remaining issue of the segmentation-based approach appears at parallel roads. We think the root cause of this issue is from the fundamental limitation of segmentation-based approaches — the road-segmentation intermediate representation. Sat2Graph eliminates this limitation through graph-tensor encoding, thereby, Sat2Graph is able to produce detailed road structures precisely even along closeby parallel roads.

**Stacked Roads.** We show the orthogonal superiority of Sat2Graph on stacked roads in Figure 5. None of the existing approaches can handle stacked roads, whereas Sat2Graph can naturally infer stacked roads thanks to the graph-tensor encoding. We find Sat2Graph may still fail to infer stacking roads in some complicated scenarios such as in Figure 5(d-e). We think this can be further improved in a future work, such as adding discriminative loss to regulate the inferred road structure.

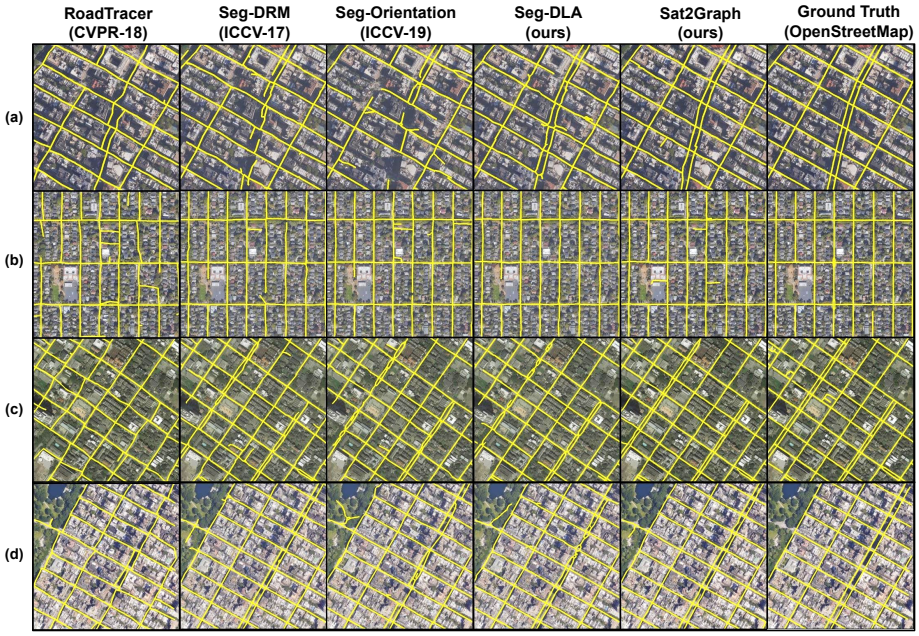


**Fig. 4.** Qualitative Comparison in Regular Urban Areas. We use the models that yield the best TOPO $F_1$ scores to create this visualization.

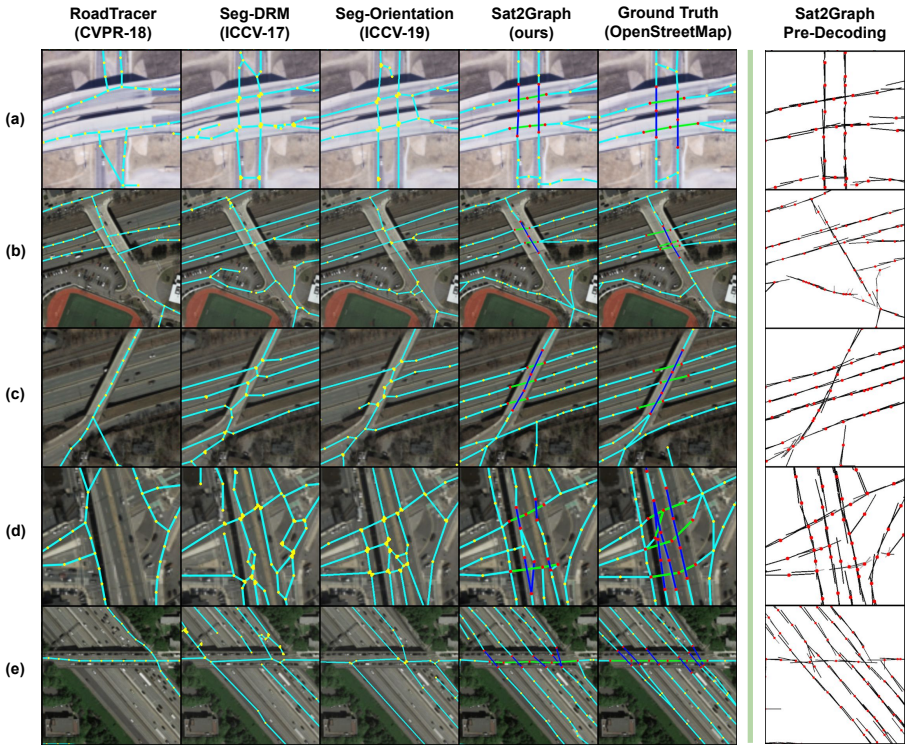


**Fig. 5.** Qualitative Comparison for Stacked Roads. Sat2Graph robustly infers stacked roads in examples (a-c), but makes some errors in (d) and (e). Prior work infers only planar graphs and incorrectly captures road topology around stacked roads in all cases. We highlight edges that cross without connecting in green and blue.

## 5 Discussion

There are two concerns regarding Sat2Graph: (1) it seems that the heavily restricted and non-lossless graph-tensor encoding may not be able to correctly represent all different road network graphs, and (2) training a model to output GTE representation with supervised learning seems impossible because the GTE representation is not consistent.

**Concern about the encoding capability.** We think there are two reasons that make GTE able to encode almost all road network graphs.

First, the road network graph is *undirected.* Although roads have directions, the road directions can be added later as road attributes, after the road network extraction. In this case, for each edge $e = (v_a, v_b)$, we only need to encode one link from $v_a$ to $v_b$ or from $v_b$ to $v_a$, rather than encode both of the two links. Even though GTE has the $\frac{360}{D_{max}}$-degree sector restriction on outgoing edges from one

vertex, this undirected-graph property makes it possible to encode very sharp branches such as the branch vertices between a highway and an exit ramp.

Second, the road network graph is *interpolatable.* There could be a case where none of the two links of an edge $e = (v_a, v_b)$ can be encoded into GTE because both $v_a$ and $v_b$ need to encode their other outgoing links. However, because the road network graph is interpolatable, we can always interpolate the edge $e$ into two edges $e_1 = (v_a, v')$ and $e_2 = (v', v_b)$. After the interpolation, the original geometry and topology remain the same but we can use the additional vertex $v'$ to encode the connectivity between $v_a$ and $v_b$.

In Table 3, we show the ratios of edges that need to be fixed using the *undirected* and *interpolatable* properties in our dataset with different $D_{\max}$ values.

| $D_{\max}$ | 3 | 4 | 5 | 6 | 8 |
|---|---|---|---|---|---|
| Fixed with the *undirected* property | 8.62% | 2.81% | 1.18% | 0.92% | 0.59% |
| Fixed with the *interpolatable* property | 0.013% | 0.0025% | 0.0015% | 0.0013% | 0.0013% |

**Table 3.**

**Concern about supervised learning.** Another concern with GTE is that for one input graph, there exist many different isomorphic encodings for it (e.g., there are many possible vertex interpolations on a long road segment.). These isomorphic encodings produce inconsistent ground truth labels. During training, this inconsistency of the ground truth can make it very hard to learn the right mapping through supervised learning.

However, counter-intuitively, we find Sat2Graph is able to learn through supervised learning and learn well. We find the key reason of this is because of the inconsistency of GTE representation doesn't equally impact the vertices and edges in a graph. For example, the locations of intersection vertices are always consistent in different isomorphic GTEs.

We find GTE has high label consistency for supervised learning at important places such as intersections and overpass/underpass roads. Often, these places are the locations where the challenges really come from. Although GTE has low consistency for long road segments, the topology of the long road segment is very simple and can still be corrected through GTE's decoding algorithm.

## 6 Conclusion

In this work, we have proposed a simple, unified road network extraction solution that combines the advantages from both segmentation-based approaches and graph-based approaches. Our key insight is a novel graph-tensor encoding scheme. Powered by this graph-tensor approach, Sat2Graph is able to surpass existing solutions in terms of topology-similarity metric at all precision-recall points in an evaluation over two large datasets. Additionally, Sat2Graph naturally infers stacked roads like highway overpasses that none of the existing approaches can handle.

## References


1. Ahmed, M., Karagiorgou, S., Pfoser, D., Wenk, C.: A comparison and evaluation of map construction algorithms using vehicle tracking data. GeoInformatica **19**(3), 601–632 (2015)
2. Bastani, F., He, S., Abbar, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Madden, S., DeWitt, D.: RoadTracer: Automatic extraction of road networks from aerial images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4720–4728 (2018)
3. Batra, A., Singh, S., Pang, G., Basu, S., Jawahar, C., Paluri, M.: Improved road connectivity by joint learning of orientation and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 10385–10393 (2019)
4. Biagioni, J., Eriksson, J.: Inferring road maps from global positioning system traces: Survey and comparative evaluation. Transportation research record **2291**(1), 61–71 (2012)
5. Biagioni, J., Eriksson, J.: Map inference in the face of noise and disparity. In: ACM SIGSPATIAL 2012 (2012)
6. Buslaev, A., Seferbekov, S.S., Iglovikov, V., Shvets, A.: Fully convolutional network for automatic road extraction from satellite imagery. In: CVPR Workshops. pp. 207–210 (2018)
7. Cadamuro, G., Muhebwa, A., Taneja, J.: Assigning a grade: Accurate measurement of road quality using satellite imagery. arXiv preprint arXiv:1812.01699 (2018)
8. Cao, L., Krumm, J.: From GPS traces to a routable road map. In: ACM SIGSPATIAL. pp. 3–12 (2009)
9. Cheng, G., Wang, Y., Xu, S., Wang, H., Xiang, S., Pan, C.: Automatic road detection and centerline extraction via cascaded end-to-end convolutional neural network. IEEE Transactions on Geoscience and Remote Sensing **55**(6), 3322–3337 (2017)
10. Chu, H., Li, D., Acuna, D., Kar, A., Shugrina, M., Wei, X., Liu, M.Y., Torralba, A., Fidler, S.: Neural turtle graphics for modeling city road layouts. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4522–4530 (2019)
11. Costea, D., Marcu, A., Slusanschi, E., Leordeanu, M.: Creating roadmaps in aerial images with generative adversarial networks and smoothing-based optimization. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 2100–2109 (2017)
12. Davies, J.J., Beresford, A.R., Hopper, A.: Scalable, distributed, real-time map generation. IEEE Pervasive Computing **5**(4) (2006)
13. Edelkamp, S., Schrödl, S.: Route planning and map inference with global positioning traces. In: Computer Science in Perspective. Springer (2003)
14. Fortier, A., Ziou, D., Armenakis, C., Wang, S.: Survey of work on road extraction in aerial and satellite images. Center for Topographic Information Geomatics, Ontario, Canada. Technical Report **241**(3) (1999)
15. Google: Google Static Maps API. https://developers.google.com/maps/documentation/maps-static/intro, accessed: 2019-03-21
16. Haklay, M., Weber, P.: OpenStreetMap: User-generated street maps. IEEE Pervasive Computing **7**(4), 12–18 (2008)
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

18. He, S., Bastani, F., Abbar, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Madden, S.: RoadRunner: Improving the precision of road network inference from gps trajectories. In: ACM SIGSPATIAL (2018)
19. He, S., Bastani, F., Jagwani, S., Park, E., Abbar, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Madden, S., Sadeghi, M.A.: Roadtagger: Robust road attribute inference with graph neural networks. arXiv preprint arXiv:1912.12408 (2019)
20. Hinz, S., Baumgartner, A.: Automatic extraction of urban road networks from multi-view aerial imagery. ISPRS Journal of Photogrammetry and Remote Sensing **58**(1-2), 83–98 (2003)
21. Li, Z., Wegner, J.D., Lucchi, A.: PolyMapper: Extracting city maps using polygons. arXiv preprint arXiv:1812.01497 (2018)
22. Máttyus, G., Luo, W., Urtasun, R.: DeepRoadMapper: Extracting road topology from aerial images. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3438–3446 (2017)
23. Mattyus, G., Wang, S., Fidler, S., Urtasun, R.: Enhancing road maps by parsing aerial images around the world. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1689–1697 (2015)
24. Máttyus, G., Wang, S., Fidler, S., Urtasun, R.: HD maps: Fine-grained road segmentation by parsing ground and aerial images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3611–3619 (2016)
25. Mosinska, A., Márquez-Neila, P., Koziński, M., Fua, P.: Beyond the pixel-wise loss for topology-aware delineation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
26. Najjar, A., Kaneko, S., Miyanaga, Y.: Combining satellite imagery and open data to map road safety. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
27. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
28. Shi, Q., Liu, X., Li, X.: Road detection from remote sensing images by generative adversarial networks. IEEE access **6**, 25486–25494 (2017)
29. Stanojevic, R., Abbar, S., Thirumuruganathan, S., Chawla, S., Filali, F., Aleimat, A.: Robust road map inference through network alignment of trajectories. In: Proceedings of the 2018 SIAM International Conference on Data Mining. SIAM (2018)
30. Van Etten, A., Lindenbaum, D., Bacastow, T.M.: Spacenet: A remote sensing dataset and challenge series. arXiv preprint arXiv:1807.01232 (2018)
31. Wang, W., Yang, N., Zhang, Y., Wang, F., Cao, T., Eklund, P.: A review of road extraction from remote sensing images. Journal of traffic and transportation engineering (english edition) **3**(3), 271–282 (2016)
32. Wegner, J.D., Montoya-Zegarra, J.A., Schindler, K.: Road networks as collections of minimum cost paths. ISPRS Journal of Photogrammetry and Remote Sensing **108**, 128–137 (2015)
33. Yu, F., Wang, D., Shelhamer, E., Darrell, T.: Deep layer aggregation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2403–2412 (2018)
34. Zhang, X., Han, X., Li, C., Tang, X., Zhou, H., Jiao, L.: Aerial image road extraction based on an improved generative adversarial network. Remote Sensing **11**(8), 930 (2019)
35. Zhang, Z., Liu, Q., Wang, Y.: Road extraction by deep residual U-Net. IEEE Geoscience and Remote Sensing Letters **15**(5), 749–753 (2018)

36. Zhou, L., Zhang, C., Wu, M.: D-LinkNet: LinkNet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction. In: CVPR Workshops. pp. 182–186 (2018)