TopoAL: An Adversarial Learning Approach for Topology-Aware Road Segmentation

Subeesh Vasu, Mateusz Kozinski, Leonardo Citraro, and Pascal Fua

CVLab**, EPFL, Lausanne, Switzerland
firstname.lastname@epfl.ch

Abstract. Most state-of-the-art approaches to road extraction from aerial images rely on a CNN trained to label road pixels as foreground and remainder of the image as background. The CNN is usually trained by minimizing pixel-wise losses, which is less than ideal to produce binary masks that preserve the road network's global connectivity.
To address this issue, we introduce an Adversarial Learning (AL) strategy tailored for our purposes. A naive one would treat the segmentation network as a generator and would feed its output along with ground-truth segmentations to a discriminator. It would then train the generator and discriminator jointly. We will show that this is not enough because it does not capture the fact that most errors are local and need to be treated as such. Instead, we use a more sophisticated discriminator that returns a label pyramid describing what portions of the road network are correct at several different scales.
This discriminator and the structured labels it returns are what gives our approach its edge and we will show that it outperforms state-of-the-art ones on the challenging RoadTracer dataset.

Keywords: Road networks, Adversarial learning, Generative adversarial network, Topology learning

1 Introduction

Many state-of-the-art algorithms for reconstructing road networks from aerial images approach the problem in terms of foreground/background binary segmentation [20, 19, 18, 8, 3], where the road pixels are the foreground ones. They rely on deep networks and often deliver better performance than approaches that directly predict the road networks as graphs instead of binary masks [2, 17, 9], even though they fail to account for the connectivity patterns of road networks. There have been several recent efforts at enforcing connectivity constraints on the segmentation outputs by designing topology-aware loss functions [23, 22] or by relying on multi-task learning [3, 33]. These approaches to enforcing connectivity on the output of a binary segmentation algorithm are mostly implicit: The network or the loss functions are modified in such a way that the resulting segmentations yield a more road-like connectivity.

** This work was funded in part by the Swiss National Science Foundation

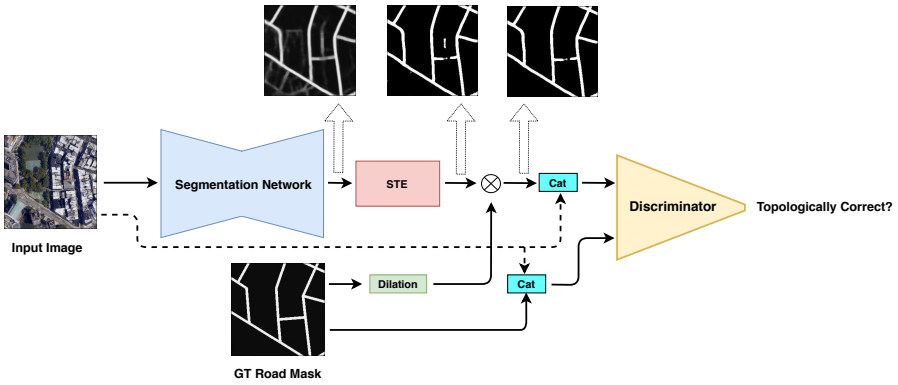

Fig. 1: **Network Architecture.** We use a segmentation network to predict the road probability maps which is then passed through a straight through estimator (STE) to generate binarised predictions. This is followed by multiplication with dilated ground truth masks to generate prediction based input to the discriminator. Another input to the discriminator is the ground truth mask which is used only during discriminator training. The road masks are concatenated with the input image before feeding them to the discriminator. Discriminator is then trained to predict spatial-aware dynamic decisions on the topological correctness of inputs.

In this paper, we propose a different and more explicit approach that relies on Adversarial Learning (AL). We use the training methodology of Generative Adversarial Networks (GAN) depicted by Fig. 1 to reduce topological discrepancies between the probability maps produced by our segmentation network and that of real road networks. A naive way to do this would be to treat the segmentation network as a generator and to feed its output along with ground-truth segmentations to a discriminator and then to train them jointly. Unfortunately, we will show that this approach is too global for the discriminator to learn to detect local topological errors and for the segmentation network to avoid making them. To remedy this, we introduce the following two key modifications:

- **Spatially aware labels.** Labeling a delineation as globally correct or incorrect is too coarse. As shown in Fig. 2, the discriminator returns a label pyramid that describes what portions of the road network are correct at several different scales.
- **Dynamically assigned labels.** These correctness decisions are not made *a priori*. Instead, the segmentations produced by the generator are evaluated for correctness in the course of the training procedure.

Our main contribution is therefore a novel AL strategy for imroving connectivity constraints on the output of road segmentation networks. We will refer to it as *TopoAL*. We will use the RoadTracer dataset [2] to show that it compares favorably to state-of-the-art methods.

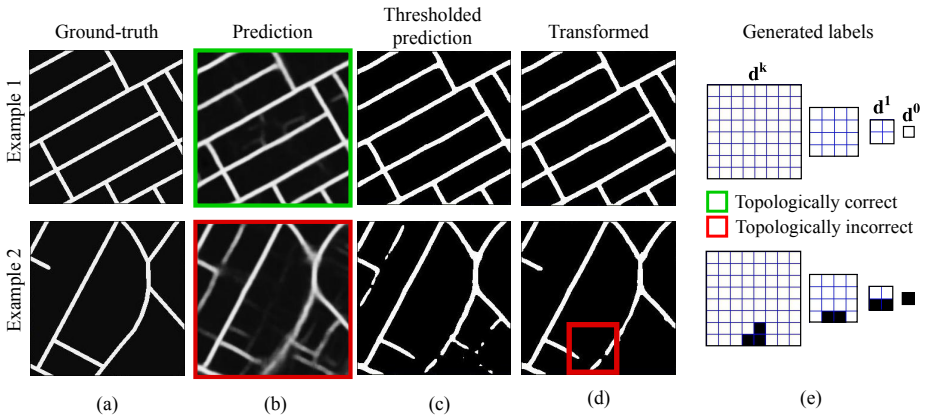

Fig. 2: **Label generation for discriminator training.** Two example patches with the corresponding multi-scale labels. One patch is topologically correct, the other contains an interruption. The labels are generated from the processing of ground-truth and predicted masks. (**a**) Ground truth road masks and corresponding (**b**) predictions from segmentation network. (**c**) Thresholded predicted masks. (**d**) Masks generated by multiplying (c) with the dilated form of (a). (**e**) Spatial-aware labels made by comparing (a) with (d). White and black boxes denote label 1 (correct topology) and 0 (topology error).

2 Related Work

Most of the early approaches to extracting road networks relied on handcrafted features and prior knowledge about road geometry to optimize complex objective functions [1, 16, 26, 31, 6, 27], to quote only the most recent ones. They have now been mostly superseded by deep learning techniques. One of the first such method is the approach of [20] in which image patches were fed as input to a fully connected neural network. Due to memory constraints, only limited context information could be exploited. The advent of convolutional neural networks (CNNs) opened the door to increasing the size of the receptive fields.

Many state-of-the-art approaches formulate road extraction as a segmentation problem and rely on encoder-decoder architectures such as U-Net [25], D-LinkNet [35], or recurrent versions of these architectures [33, 30]. While these approaches feature large receptive fields, none of them explicitly takes into account the connectivity of the resulting binary masks. Several recent approaches have attempted to remedy this. A topology-aware loss function was introduced in [22, 23] while the use of an additional centerline extraction module was proposed in [33]. In [3], orientation prediction is used as an auxiliary task to improve the connectivity of the predicted masks. [18] introduced a post-processing algorithm to reconstruct the graph from segmentation outputs by reasoning about missing connections and applying a number of heuristics. A unified approach to segmentation and connectivity reasoning was presented in [21]. It uses a seg-

mentation network and a classifier that shares the same encoder representation. The classifier is used to reason out the connectivity in the segmentation output. Finally, [17] proposed an approach to predict road segments in the form of vector representation instead of pixel-wise segmentations. Their approach use a CNN to extract key-points and edge evidences from a given patch, which were then fed sequentially to a recurrent neural network (RNN) to produce vector representations of the underlying road segments.

A radically different approach is to directly build the graph without segmenting. This is typically done iteratively by adding road segments one by one. In [28], a CNN is trained to predict the local connectivity among the central pixel of an input patch and its border points. To reconstruct the road network corresponding to the whole image, the algorithm iteratively performs patch-wise identification of input and exit points and the associated connections. In [2], a CNN-based decision function is used to guide an iterative search process, which starts from a search point known to be part of the road network. At each step, the CNN takes the point and the already reconstructed roads in its neighborhood as input and outputs the decision to walk a fixed distance along a certain direction or to stop and return to previous search point. In a similar spirit, a Neural Turtle Graphics can be used to iteratively generate new nodes and the corresponding edges connecting to the existing nodes [9]. This approach relies on RNNs instead of CNNs, as in [28, 2].

Both segmentation based approaches and graph-based approaches have some clear vulnerabilities. The former are subject to small-scale topological errors in the form of missing connections. The latter, though free from topological errors, are vulnerable to error propagation due to the iterative reconstruction policy. We will show that our proposed approach is less affected by these difficulties as compared to other state-of-the-art algorithms.

Some of the recents works have proposed to use multiple discriminators in generative adversarial networks setups [7, 13, 29]. A domain adaptation technique for semantic segmentation is proposed in [7]. They divide the discriminator input into different spatial regions, and associate different classifiers to each region. The concepts of multi-scale discriminator [29] and local-global discriminators [13] were introduced to examine the input data at different context levels. Unlike all these methods, our approach differs in multiple aspects including label generation, the use of a single network architecture, and the input characteristics that the discriminator has to learn.

3 Method

As shown in Fig. 1, our approach closely traces the structure of a GAN [12]. For this reason and simplicity in the terminology, we refer to the segmentation network as the generator and to the evaluator of the delineated predictions as the discriminator. In a traditional AL, the discriminator would assign a binary label to the segmentations it produces. However, this is not suitable for our purpose because the predicted mask can be correct almost everywhere except

for few locations that result in poor connectivity. To properly account for this and to provide spatially-aware supervision, our discriminator predicts the more sophisticated labels depicted by Fig. 2. They are computed online by splitting the image into increasingly fine partitions and then labeling each element of these partitions as valid or not given the ground-truth data. The labels corresponding to ground truth masks have the same structure but are uniformly valid and the corresponding loss function takes all labels at all scales into account.

The combination of using these more sophisticated labels and computing them dynamically during training, instead of fixing them *a priori* as is usually done is what gives our approach its edge. We now formalize it and describe its individual components in turn.

3.1 Formalization

Let $\mathbf{x} \in \mathbb{R}^{H \cdot W \cdot C}$ be a C-channel input image of size $H \times W$, and let $\mathbf{y} \in \{0,1\}^{H \cdot W}$ be the corresponding ground-truth road mask, with 1 indicating pixels corresponding to a road and 0 indicating the background pixels. Let us consider a segmentation network G that takes $\mathbf{x}$ as the input and outputs a *probability map* $\hat{\mathbf{y}} = G(\mathbf{x}) \in [0,1]^{H \cdot W}$. For any given pixel i, $\hat{\mathbf{y}}_i$ is taken to be the probability that it is a road pixel. The weights of the network are typically learned by minimizing the pixel-wise binary cross-entropy (BCE) loss

$$\mathcal{L}_{bce}(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_i \left[(1 - \mathbf{y}_i) \cdot \log(1 - \hat{\mathbf{y}}_i) \; + \mathbf{y}_i \cdot \log \hat{\mathbf{y}}_i \right]. \tag{1}$$

Such a loss function penalizes mistakes everywhere equally regardless of their influence on the underlying geometry of the predicted road network. To remedy this, we use the segmentation network G as the generator of the GAN of Fig. 1 and define a discriminator network D whose role is to identify topological errors in the generator output. G and D are trained by making them play a game: The weights of G are optimized to generate segmentations that D cannot distinguish from ground-truth ones and the weights of D are optimized to make that distinction as well as possible.

3.2 Discriminator Network: *D*

In a traditional GAN, D is trained to return a binary label that is 1 if a sample is a ground-truth one and 0 if it is one generated by the G network. We will show in the result section that this does not help much in our case, mostly because the errors that G makes are local and that a single binary label is not enough to characterize them. Instead, we designed D to classify different portions of the predicted masks at different scales as correct or not. This pyramid approach allows the discriminator to provide both local and global supervision to the generator and to model spatial dependencies between neighboring locations. We now describe the label generation and the discriminator architecture in detail.

Scale Space Labels. The most challenging aspect of road network reconstruction is recovering the connectivity of the network by avoiding topological errors. They most often manifest themselves as short breaks in road segments that spoil an otherwise mostly correct binary mask. An effective discriminator must therefore detect and localize such mistakes so that the generator can fix them.

To this end, we define spatially-aware labels as follows. We consider a pyramid of increasingly zoomed-out versions of the predicted mask, as shown in Fig. 2(e). At level k of the resulting pyramid, we divide the mask into non-overlapping patches of size $H_k \times W_k$ and associate a binary value to each one, depending on the topological correctness within it. The label matrix generated at level k is of size $\frac{H}{H_k} \times \frac{W}{W_k}$. The collection of such label matrices can be regarded as a representation of topological correctness at different scales and locations.

In practice, our input images are of size 256×256 and we have used four levels with $H_k = W_k = \{256, 128, 64, 32\}$. At one extreme of the range, we assign a single label to the whole image and, at the other, we assess the correctness within 32×32 patches. As shown in Fig. 2(e), for a road mask that has topological error, the values in the label matrices are neither all zeros nor all ones. Instead, they encode fine- or coarse-level locations of topological errors in the generators output. This allows the discriminator to effectively use multi-scale information.

Dynamic Label Assignment. There is no way to know *a priori* in which of the patches discussed above the topology is correct. Therefore, unlike in traditional GANs, we cannot predefine the labels we assign to the generator output. Instead, we must do this dynamically for each prediction made by the generator.

For example, the patches outlined in green or red in Fig. 2 are deemed correct or incorrect, respectively. To make this assessment, we compare the probability map produced by the generator to the ground-truth within the patch of interest. Alongside, to match with the labelling strategy, we use a transformed form of $\hat{\mathbf{y}}$ to construct the corresponding input to the discriminator. To generate the inputs and labels for the discriminator training we use the following steps:

1. *Differentiable Thresholding.* We binarize the probability map produced by the generator. To preserve differentiability, we use STE [4, 34] that thresholds during the forward pass while behaving as the identity function during the backward pass. We set the threshold to be 0.5 in our experiments.
2. *Multiplication by the dilated ground truth.* The generator can produce false negatives—road pixels that are classified as background as highlighted in Fig. 2(d)—and false positives—background pixels that are classified as road pixels as highlighted in Fig. 2(c). The false negatives are those that cause disconnections and break the connectivity of the network. Furthermore, it is not unusual for some real roads to be missing from the ground-truth. We have therefore found empirically that ignoring the false positives and focusing on the false negatives to be beneficial. Before feeding the thresholded prediction to the input of the discriminator, we therefore multiply it with a dilated version of the ground truth mask. The dilation accounts for the fact that the centerline locations are not always precise in the ground truth. We set

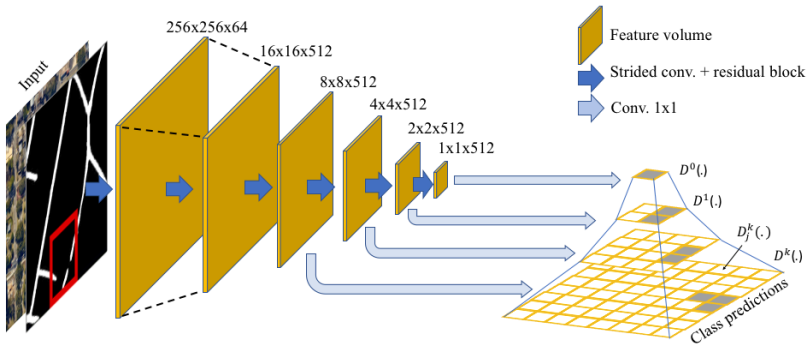

Fig. 3: **Discriminator.** The discriminator is a single fully convolutional network that downsample the input to the desired scales and locations. Each downsampling stage is composed of a convolutional layer with stride factor 2, followed by a residual block. The first convolution layer is set to have 64 feature channels. The channel number doubles after every strided convolution until it reaches 512. The class prediction are produced from the latest stages using a 1×1 convolutional layer.

the dilation factor to 3. Let us denote the resulting mask as $T_0(\hat{\mathbf{y}})$. Two examples are shown in Fig. 2(d).

3. *Concatenating with Input Image.* This final step is used to generate the complete discriminator input. The ground-truth road masks may contain genuine interruptions, for example because of road dead ends near to other road sections as in the top-left corner of second row of Fig. 2 (d). To help the discriminator distinguish these from unwarranted ones (i.e., erroneous interruptions of the predicted road network as in the red box in Fig. 2 (d)), we also feed it the input image $\mathbf{x}$ so that it can examine the context in which these interruptions occur. To this end, we concatenate $\mathbf{x}$ with the road mask before feeding it to the discriminator. We will refer to the discriminator formed by concatenating $T_0(\hat{\mathbf{y}})$ with $\mathbf{x}$ as $T(\hat{\mathbf{y}})$.
4. *Assigning Label Values.* To generate the label values, we compare the ground truth skeleton with the prediction $T_0(\hat{\mathbf{y}})$ and count the false nagatives (number of pixels in the skeleton that are not covered in $T_0(\hat{\mathbf{y}})$). We use the count of false negatives to identify the cell that is likely to contain topological errors. We assign zero to these patches and a 1 otherwise.

The operations corresponding to STE, dilation, and concatenation are represented by the pink, green, and blue boxes in Fig. 1. Detailed illustration on the proposed label generation scheme can be found in Fig. 2.

Architecture. To implement D, we use a fully convolutional network similar to PatchGAN [14], but with four outputs, each having a different resolution. As a result, D outputs a pyramid of probability maps having the same structure as the spatially aware labels. Fig.3 depicts its architecture. It comprises eight

stages that downsample to $\frac{1}{256}^{th}$ of the input resolution. Each stage is made of a convolutional layer with stride factor 2 followed by a residual block. The first convolution layer has 64 feature channels. The channel number doubles after every strided convolution until stage 4; the number of channels are kept at 512 afterward. The final residual block is followed by a single convolutional layer to output the prediction at the last stage. Features from the output of residual blocks at stages 5-7 are passed through a single convolutional layer to generate predictions at respective resolutions.

3.3 Generator Network: *G*

One could use any standard segmentation network as the generator G. To demonstrate this, we experimented with two different ones, a standard *UNet* [25] and a recurrent version [30] of it, which we will refer to as *DRU*.

UNet is a fully convolutional encoder-decoder network with skip connections that serves as the backbone of several recent road extraction algorithms [22, 21]. We follow the standard *UNet* design with four levels. Each one comprises two convolutional operations followed by max pooling. We set the first convolutional unit to have 64 feature channels. *DRU* [30] is a *UNet* with Dual-gated Recurrent Units. It performs recursion on the input-output and in multiple internal states of the network to improve the overall performance with only minimal increments in model size. In the original paper, a lightweight *UNet* architecture with 32 channels was used. Here we use 64 channels as in the standard *UNet*.

By combining our proposed discriminator with the two generator networks *UNet* and *DRU*, we build two different methods and dub them as *UNet-TopoAL* and *DRU-TopoAL* respectively.

3.4 Training

To train the generator G and discriminator D, we follow the usual GAN approach, which is to alternatively minimize the loss functions with respect to the generator and the discriminator. The difference, however, resides in the discriminator that takes as input $T(\hat{\mathbf{y}})$, the generator's output transformed as described before, and $c[\mathbf{y}, \mathbf{x}]$ formed by concatenating the ground truth mask $\mathbf{y}$ with $\mathbf{x}$. We train the discriminator to minimize the BCE loss between $T(\hat{\mathbf{y}})$ outputs and the corresponding spatially aware labels. We write the loss function as

$$\mathcal{L}_{D1}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_k \mathcal{L}_{bce}(D^k(T(\hat{\mathbf{y}})), \mathbf{d}^k) + \sum_j -\log D_j^k(c[\mathbf{y}, \mathbf{x}]) \ , \tag{2}$$

where $k \in \{0, 1, 2, 3\}$ is used to index the four outputs from the discriminator. D_j^k refers to j^{th} element of k^{th} discriminator output, and $\mathbf{d}^k$ is the spatial-aware labels for the level k.

The generator is trained using a combination of BCE Loss and adversarial loss given by

$$\mathcal{L}_{G1}(\hat{\mathbf{y}}, \mathbf{y}) = \mathcal{L}_{bce}(\hat{\mathbf{y}}, \mathbf{y}) + \lambda_A \sum_k \sum_j -\log D_j^k(T(\hat{\mathbf{y}})) \ , \tag{3}$$

where λ_A is a scalar weight that controls the influence of adversarial loss. In our experiments we set λ_A to be 0.005, an empirically found optimal value.

Note that if we use a single discriminator output at level k = 0 in Eq. 2, predefined label value of 0 for the generator output, that is, $d^0 = 0$, and $(\hat{\mathbf{y}}, \mathbf{y})$ as the two inputs to the discriminator network, this reduces to a standard GAN. We will use this as a baseline that we will refer to as *VanillaGAN*.

4 Experiments

We now describe the dataset we have tested our approach on, the performance measures we used to assess the quality of the reconstructions, and baselines we used for comparison purposes. We then show that our approach can be used to enhance the performance of two of them and report our performance against that of the others. Additional experimental results including performance comparison on the DeepGlobe Dataset [10] and the ablation studies revealing the importance of each component of our approach are provided in the supplementary material.

4.1 Datasets, Metrics, and Baselines

Dataset. We perform our experiments on the RoadTracer dataset [2], which is one of the most recently published, largest, and most challenging road network dataset. It contains high-resolution satellite images covering the urban areas of forty cities in six different countries. The labels are generated using OpenStreetMap in the form of graphs. It covers areas featuring highways, urban roads, and rural paths, which results in extreme appearance variations. The roads are often occluded by trees, buildings, and shadows, making it difficult for segmentation approaches to preserve topology. Finally, the set of 25 training cities and 15 testing cities are totally disjoint, which makes generalization more difficult than if the training and testing images were from the same city.

Metrics. Many metrics have been developed to compare the estimated road networks to that of the ground truth. These metrics can be broadly classified into two categories, pixel-based and topology-based. We use both kinds for the sake of completeness.

- **Correctness/Completeness/Quality (*CCQ*).** This is a pixel-based metrics intended to measure the similarity between a skeletonized prediction and the corresponding annotation. In segmentation tasks, predictions are often evaluated using precision = $\frac{|\mathrm{TP}|}{|\mathrm{PP}|}$, recall = $\frac{|\mathrm{TP}|}{|\mathrm{AP}|}$, and intersection-over-union = $\frac{\mathrm{TP}}{\mathrm{PP} \cup \mathrm{AP}}$, where PP is the set of foreground pixels in the prediction, AP is the set of foreground pixels in the ground truth, and TP = PP $\cap$ AP. To account for the shift between predictions and the ground truth, precision, recall, and intersection-over-union have been relaxed [32]. The resulting quantities are named correctness, completeness, and quality respectively. To assess the performance using a single metric we report the values of quality (qual.). In our experiments, we set the allowable pixel shift to 2 pixels.

- **Too Long/Too Short (*TLTS*).** *TLTS* [31] compares the lengths of the shortest paths between randomly sampled ground-truth nodes matched to the prediction. If the length of the path in the predicted graph is within 5% of that of the path in the ground-truth the path is declared to be correct. We use the percentage of correct paths, denoted as *TLTS*-corr, to assess the prediction quality. We set the threshold defining if nodes from the ground-truth are matched to the prediction to 25.
- **Average Path Length Similarity (*APLS*).** *APLS* [11] aggregates the differences in optimal path lengths between nodes in the ground truth and predicted graphs. The Average Path Length Similarity is computed as

$$1 - \frac{1}{N} \sum \min \left\{ 1, \frac{L(a,b) - L(\hat{a},\hat{b})}{L(a,b)} \right\} \tag{4}$$

  where a and b are nodes in the ground truth graph, $\hat{a}$ and $\hat{b}$ are the corresponding nodes in the predicted graph, N is the number of nodes in ground truth, and L denotes the length of the shortest path connecting them. To penalize false positives, the same procedure is repeated by swapping ground-truth and prediction. The final score is the harmonic mean of the two.
- **Junction (*JUNCT*).** *JUNCT* [2] compares the degree of corresponding nodes with at least three incident edges, called junctions. The correspondences are established greedily by matching closest nodes. For each ground-truth junction v that is matched to a predicted junction u, the per-junction recall $f_{u,correct}$ is computed by taking into account the fraction of edges incident on v that are also captured around u. The same operation is performed to compute the per-junction 1-precision $f_{v,error}$ which is the fraction of edges incident on u that do not appear around v. For this metric we report the f1-score compute using $f_{u,correct}$ as recall and $1 - f_{v,error}$ as precision. We used the implementation provided in [2] with default parameters.
- **Holes and Marbles (*H&M*).** This metric first extracts small subgraphs from the ground-truth and match them to the prediction. Then, compares sets of locations in the two subgraphs accessible by traveling a predefined distance away from a randomly sampled point. Virtual control points, namely holes, are dropped at regular intervals along the paths in the ground truth graph. The same process is repeated in the predicted graph, these control points are called marbles. A hole is said to be matched if it lies sufficiently close to one of the marbles. The process is repeated for many subgraphs and the total count of matched and unmatched points are then used to compute precision and recall. To asses the prediction quality using a single value, we report the f1-score as in [5]. We set the threshold defining if nodes from the ground-truth are matched to the prediction to 25. We extracted subgraph of radius 300 and sampled 1000 of them for each sample.

Baselines and Variants. We use the following approaches that are briefly described in Section 2 as baselines.

- *UNet* [25]: Fully-convolutional network with skip connections.

- *UNet-VGG* [22]: Fully-convolutional network with skip connections and topology-aware loss function.
- *DRU* [30]: Fully-convolutional network with skip connections and recursion.
- *Seg-Path* [21]: Two-branch network that jointly learns to segment linear structures and to classify candidate connections.
- *MultiBranch* [3]: Recursive architecture jointly trained for road segmentation and orientation estimation.
- *RCNN-UNet* [33]: Fully convolutional network with recursive convolutional layers.
- *DeepRoadMapper* [18]: A fully convolutional network based segmentation followed by heuristics based post-processing to generate the graph.
- *Roadtracer* [2]: Graph constructed iteratively with new node locations being selected by a convolutional network.

We compare these baselines against three variants of our approach introduced in Sections 3.3 and 3.4.

- *DRU-TopoAL*: We use the network of *DRU* [30] as our generator.
- *UNet-TopoAL*: We use the network of *UNet* [25] as our generator.
- *UNet-VanillaGAN*: We use the network of *UNet* [25] as our generator and replace our sophisticated discriminator by a simple one that returns a simple binary flag for each input mask.

4.2 Implementation details

To train our *TopoAL* networks we rendered the RoadTracer ground-truth graphs to half resolution to generate pixel-wise annotations. We have experimented with different input sizes and observed that half-resolution produced the best results on the Roadtracer dataset when the network is trained using binary cross-entropy loss (BCE) alone. We take this to mean that the higher resolution details of this dataset are not key to producing globally correct topologies, and the half-resolution provides the optimal trade-off between the required details in the input image and the effective context available to the network. As input, we used 256×256 patches randomly cropped from the training images. To improve the generalization of learned network, we employed data augmentations in the form of random horizontal flip, vertical flip, scaling and rotation. To train both the generator and discriminator, we used the Adam optimizer [15] with a 10^{-4} learning rate and a batch size of 4. All the models are implemented in Pytorch [24]. To construct the dynamic labels, a 32x32 patch is declared topologically incorrect if it contains an erroneous road interruption of at least 4 pixels long. Larger patches are declared incorrect if they contain an incorrect 32x32 sub-patch. We selected this threshold based on visual inspection to separate road interruptions from misalignment of the predicted and annotated roads.

We trained *UNet*, *UNet-VGG* and *DRU* using the same settings as *TopoAL*, as described in Section 3.3. For *DRU* and *DRU-TopoAL*, we set the number of

		Pixel-based	Topology-aware			
	Method	CCQ qual.	TLTS corr.	APLS	JUNCT f1	*H&M* f1
RoadTracer	*UNet* [25]	0.632	0.323	0.619	0.792	0.737
	UNet-VanillaGAN	0.636	0.328	0.607	0.776	0.748
	UNet-TopoAL (Ours)	**0.658**	**0.388**	**0.666**	**0.808**	**0.767**
	DRU [30]	0.656	0.437	0.697	0.821	0.768
	DRU-TopoAL (Ours)	**0.657**	**0.480**	**0.725**	**0.837**	**0.787**

Table 1: Quantitative comparison between baselines segmentation networks, our improved versions, and *UNet-VanillaGAN*. Our *TopoAL* approach improves the baselines on all metrics. On the other hand, *UNet-VanillaGAN* performance is only comparable to that of the baseline network *UNet*.

recursions to 3 and use the sum of losses corresponding to outputs from all the recursions. For *DRU-TopoAL*, outputs from all the recursions is used for the discriminator training, and the total adversarial loss is computed as the sum of losses corresponding to all the recursions. We retrained the *MultiBranch* network on the RoadTracer dataset using the code provided by the authors with default settings. For *DeepRoadMapper*, we use the results shared by the authors of [2]. For all other methods we use the predicted road networks made publicly available by the authors.

4.3 Boosting the Performance of Existing Architectures

In Table 1, we compare *UNet* and *DRU* against *UNet-TopoAL* and *DRU-TopoAL*, which use *UNet* and *DRU* as their generators. In both cases, we can see our scheme consistently boosts their performance, especially the *TLTS*, *APLS*, *JUNCT* and *H&M* metrics that are designed to assess topological correctness. The first row of Fig. 4 provides corresponding qualitative results. We also report the performance of *UNet-VanillaGAN*, which implements a standard GAN, in Table 1. It can be noted that it is not better than its generator *UNet*. Therefore, using a GAN by itself does not bring the same improvements as *TopoAL*. As is evident from the first row of Fig. 4, adding our topology loss not only refine the predictions but also improves the topological correctness. On the other hand, predictions from *UNet-VanillaGAN* does not respect the topological correctness.

4.4 Comparison against the State-of-the-Art

We now turn to compare our results to those of all the baselines discussed above and report the results in Table 2. We provide corresponding qualitative results in the last three rows of Fig. 4.

DRU-TopoAL does best on three of the four topology-aware metrics and is second on the fourth, without any of the post-processing that *Seg-Path* perform.

		Pixel-based	Topology-aware			
	Method	CCQ qual.	TLTS corr.	APLS	JUNCT f1	*H&M* f1
RoadTracer	*DeepRoadMapper* [18]	0.435	0.069	0.247	0.514	0.469
	Roadtracer [2]	0.477	0.420	0.591	0.812	0.714
	RCNN-UNet [33]	0.628	0.201	0.474	0.790	0.701
	MultiBranch [3]	**0.659**	0.439	0.682	0.798	0.765
	Seg-Path [21]	0.535	**0.489**	0.679	0.754	0.688
	UNet-VGG [22]	0.636	0.328	0.607	0.776	0.748
	UNet [25]	0.632	0.323	0.619	0.792	0.737
	DRU [30]	0.656	0.437	0.697	0.821	0.768
	UNet-TopoAL (Ours)	**0.659**	0.388	0.666	0.808	0.767
	DRU-TopoAL (Ours)	0.657	0.480	**0.725**	**0.837**	**0.787**

Table 2: Quantitative comparison between state-of-the-art road network reconstruction algorithms and our proposition. Our proposition *DRU-TopoAL* produces the best results in four out of five metrics while it comes second to *Seg-Path* in *TLTS*.

For the pixel-based measures, *DRU-TopoAL* performs honorably but does not truly dominate the other algorithms. This is not surprising because *TopoAL* targets small interruptions in road segment. They are few in numbers but critical in terms of topological correctness. Since CCQ only performs pixel-wise comparisons, it is relatively insensitive to the kind of errors we detect and fix.

As indicated by the example in the second row of Fig. 4, adding VGG loss (UNet-VGG) helps to suppress spurious prediction errors but does not lead to significant improvement in the prediction of true roads. From the third row of Fig. 4 one can observe that, despite the multi-tasking strategy, *MultiBranch* fails to predict the roads at occluded regions (highlighted in blue), a limitation that was reported in [3]. On the other hand, *UNet-TopoAL* and *DRU-TopoAL* depicts progressive improvements towards filling up such gaps. In comparison with the proposed methods, *Roadtracer* and *Seg-Path* that use trained networks to enforce connectivity explicitly, either fails to predict some road segments completely (highlighted in blue) or result in big false positives (highlighted in green) that maintains connectivity to the other parts of the predicted network. The last row of Fig. 4 is an exceptional case wherein trees mostly occlude true roads.[1] As is evident, among the segmentation methods, *DRU-TopoAL* shows the most promising result against occlusion effects, while maintaining favorable performance against *Seg-Path*.

[1] Ground truth mask does not show most of the actual roads because of the omission noise.

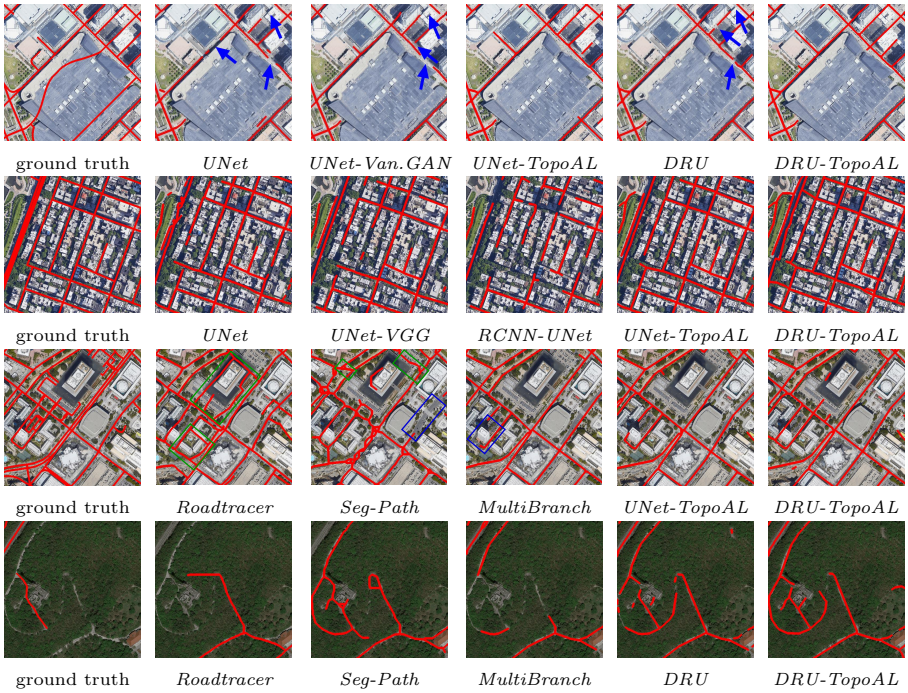

Fig. 4: **Road extraction.** Ground truth and predicted roads are marked in red. Our *TopoAL* approach improves the topological correctness over their respective generators when used by themselves (1^{st} row), performs better than the segmentation baselines (2^{nd} row) and connectivity-based methods (3^{rd} row), and generalizes well to challenging cases (4^{th} row).

5 Conclusion

In this paper, we have proposed an AL strategy that is tailored for extracting networks of curvilinear structures. Its key ingredient is a discriminator that, instead of returning a simple yes or no answer, return a spatially-meaningful descriptor of which parts of the images are well modeled and which are not. The corresponding decisions are made at run-time as opposed to be taken *a priori* as in traditional GANs. As a result, we can outperform the state-of-the-art without having to resort to particularly complicated architectures.

Networks of curvilinear structures—blood vessels, dendrites and axons, bronchi, among others—are prevalent in biomedical imagery and recovering their connectivity is also crucial. In future work, we will therefore extend this approach to delineation in 3D image stacks.

References

1. Barzohar, M., Cooper, D.B.: Automatic finding of main roads in aerial images by using geometric-stochastic models and estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence **18**(7), 707–721 (1996)
2. Bastani, F., He, S., Alizadeh, M., Balakrishnan, H., Madden, S., Chawla, S., Abbar, S., Dewitt, D.: Roadtracer: Automatic Extraction of Road Networks from Aerial Images. In: Conference on Computer Vision and Pattern Recognition (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: Conference on Computer Vision and Pattern Recognition (June 2019)
4. Bengio, Y., Léonard, N., Courville, A.: Estimating or Propagating Gradients through Stochastic Neurons for Conditional Computation. In: arXiv Preprint (2013)
5. Biagioni, J., Eriksson, J.: Inferring road maps from global positioning system traces: Survey and comparative evaluation. Transportation Research Record: Journal of the Transportation Research Board **2291**, 61–71 (12 2012)
6. Chai, D., Forstner, W., Lafarge, F.: Recovering Line-Networks in Images by Junction-Point Processes. In: Conference on Computer Vision and Pattern Recognition (2013)
7. Chen, Y., Li, W., Van Gool, L.: ROAD: Reality Oriented Adaptation for Semantic Segmentation of Urban Scenes. In: Conference on Computer Vision and Pattern Recognition. pp. 7892–7901 (2018)
8. 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 Trans. Geoscience and Remote Sensing **55**(6), 3322–3337 (2017)
9. Chu, H., Li, D., Acuna, D., Kar, A., Shugrina, M., Wei, X., Liu, M., Torralba, A., Fidler, S.: Neural Turtle Graphics for Modeling City Road Layouts. In: International Conference on Computer Vision (2019)
10. Demir, I., Koperski, K., Lindenbaum, D., Pang, G., Huang, J., Basu, S., Hughes, F., Tuia, D., Raskar, R.: Deepglobe 2018: A Challenge to Parse the Earth through Satellite Images. In: Conference on Computer Vision and Pattern Recognition (June 2018)
11. Etten, A.V., Lindenbaum, D., Bacastow, T.: Spacenet: A remote sensing dataset and challenge series. CoRR **abs/1807.01232** (2018), http://arxiv.org/abs/1807.01232
12. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems (2014)
13. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and locally consistent image completion. ACM Transactions on Graphics **36**(4), 1–14 (2017)
14. Isola, P., Zhu, J., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Conference on Computer Vision and Pattern Recognition (2017)
15. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. In: International Conference on Learning Representations (2015)
16. Laptev, I., Mayer, H., Lindeberg, T., Eckstein, W., Steger, C., Baumgartner, A.: Automatic Extraction of Roads from Aerial Images Based on Scale Space and Snakes. Machine Vision and Applications (2000)

17. Li, Z., Wegner, J., Lucchi, A.: Topological map extraction from overhead images. In: International Conference on Computer Vision (2019)
18. Máttyus, G., Luo, W., Urtasun, R.: Deeproadmapper: Extracting Road Topology from Aerial Images. In: International Conference on Computer Vision. pp. 3458–3466 (2017)
19. Mnih, V., Hinton, G.: Learning to Label Aerial Images from Noisy Data. In: International Conference on Machine Learning (2012)
20. Mnih, V., Hinton, G.: Learning to Detect Roads in High-Resolution Aerial Images. In: European Conference on Computer Vision. pp. 210–223 (2010)
21. Mosińska, A., Kozinski, M., Fua, P.: Joint Segmentation and Path Classification of Curvilinear Structures. IEEE Transactions on Pattern Analysis and Machine Intelligence (2019)
22. Mosińska, A., Marquez-Neila, P., Kozinski, M., Fua, P.: Beyond the Pixel-Wise Loss for Topology-Aware Delineation. In: Conference on Computer Vision and Pattern Recognition. pp. 3136–3145 (2018)
23. Máttyus, G., Urtasun, R.: Matching adversarial networks. In: Conference on Computer Vision and Pattern Recognition (2018)
24. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NIPS-W (2017)
25. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Conference on Medical Image Computing and Computer Assisted Intervention. pp. 234–241 (2015)
26. Stoica, R., Descombes, X., Zerubia, J.: A Gibbs Point Process for Road Extraction from Remotely Sensed Images. International Journal of Computer Vision **57**(2), 121–136 (2004)
27. Turetken, E., Benmansour, F., Andres, B., Pfister, H., Fua, P.: Reconstructing Loopy Curvilinear Structures Using Integer Programming. In: Conference on Computer Vision and Pattern Recognition (June 2013)
28. Ventura, C., Pont-Tuset, J., Caelles, S., Maninis, K., Gool, L.V.: Iterative Deep Learning for Network Topology Extraction. In: British Machine Vision Conference (2018)
29. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. In: Conference on Computer Vision and Pattern Recognition (2018)
30. Wang, W., Yu, K., Hugonot, J., Fua, P., Salzmann, M.: Recurrent U-Net for Resource-Constrained Segmentation. In: International Conference on Computer Vision (2019)
31. Wegner, J., Montoya-Zegarra, J., Schindler, K.: A Higher-Order CRF Model for Road Network Extraction. In: Conference on Computer Vision and Pattern Recognition. pp. 1698–1705 (2013)
32. Wiedemann, C., Heipke, C., Mayer, H., Jamet, O.: Empirical Evaluation of Automatically Extracted Road Axes. In: Empirical Evaluation Techniques in Computer Vision. pp. 172–187 (1998)
33. Yang, X., Li, X., Ye, Y., Lau, R.Y.K., Zhang, X., Huang, X.: Road Detection and Centerline Extraction via Deep Recurrent Convolutional Neural Network U-Net. IEEE Transactions on Geoscience and Remote Sensing pp. 1–12 (2019)
34. Yin, P., Lyu, J., Zhang, S., Osher, S.J., Qi, Y., Xin, J.: Understanding straight-through estimator in training activation quantized neural nets. In: International Conference on Learning Representations (2019)

35. 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 (2018)