Topological Map Extraction From Overhead Images

Zuoyue Li[1], Jan Dirk Wegner[2], Aurélien Lucchi[1]
ETH Zürich, Switzerland
[1]{li.zuoyue,aurelien.lucchi}@inf.ethz.ch, [2]jan.wegner@geod.baug.ethz.ch

Abstract

We propose a new approach, named PolyMapper, to circumvent the conventional pixel-wise segmentation of (aerial) images and predict objects in a vector representation directly. PolyMapper directly extracts the topological map of a city from overhead images as collections of building footprints and road networks. In order to unify the shape representation for different types of objects, we also propose a novel sequentialization method that reformulates a graph structure as closed polygons. Experiments are conducted on both existing and self-collected large-scale datasets of several cities. Our empirical results demonstrate that our end-to-end learnable model is capable of drawing polygons of building footprints and road networks that very closely approximate the structure of existing online map services, in a fully automated manner. Quantitative and qualitative comparison to the state-of-the-art also shows that our approach achieves good levels of performance. To the best of our knowledge, the automatic extraction of large-scale topological maps is a novel contribution in the remote sensing community that we believe will help develop models with more informed geometrical constraints.

1. Introduction

A fundamental research task in computer vision is pixel-accurate image segmentation, where steady progress has been measured with benchmark challenges such as [27, 12, 11]. The classical approach in this field consists of assigning a label to each image pixel describing what category it belongs to, thus yielding a labeled image as output. However, for many applications, this is not the final desired output from a user's point of view. In this paper, we will instead focus on applications that require a graph or polygon representation as output. Our interest will be in developing a method that, from an input image, directly produces a polygon representation that describes geometric objects using a vector data structure. Motivated by the success of recent works [10, 8, 5, 1], we avoid explicit pixel-wise labeling altogether, but instead directly predict polygons from images in an end-to-end learnable approach.

Figure 1: PolyMapper result for Boston overlaid on top of the original aerial imagery. Buildings and roads are directly predicted as polygons. See additional results in Fig. 10.

Our research is inspired by the insight that for many applications, image segmentation is just an intermediate step of a more comprehensive workflow that aims at a higher-level, abstract, vectorized representation of the image content. A good example is automated map generation from aerial imagery where existing research has mostly focused on aerial image segmentation such as [9, 48, 50, 30, 20, 51, 31]. We make this application our core scenario because we have access to virtually unlimited data from OpenStreetMap (OSM) [17, 16, 14] and high-resolution RGB orthophotos from Google Maps.

Usually, a full mapping pipeline consists of converting an orthophoto to a semantically meaningful raster map (*i.e.*, semantic segmentation), followed by further processing such as object shape refinement, vectorization, and map generalization techniques. Here, we turn this multi-step workflow into an end-to-end learnable deep learning architecture, PolyMapper, which outputs topological maps of buildings and roads directly, given aerial imagery as input.

Our approach performs object detection, instance segmentation, and vectorization within a unified approach that relies on modern CNNs architectures and RNNs with convolutional long-short term memory (ConvLSTM) [45] modules. As illustrated in Fig. 5, the CNN takes as input a city tile and extracts keypoints and edge evidence of building footprints and road networks, which are fed sequentially to the multi-layer ConvLSTM modules. The latter produces a vector representation for each object in a given tile. In

the case of roads, we also propose an approach that reformulates the topology of roads (typically an undirected graph) as polygons by following a maze solving algorithm that guarantees the shape consistency (sequences) of different objects (see Sec. 3.3). Finally, the roads from different tiles are connected and combined with the buildings to form a complete city map. A PolyMapper result for the city Boston is shown in Fig. 1, while the results of Chicago and Sunnyvale are illustrated in Fig. 10.

We validate our approach for the automated mapping of road networks and building footprints on the existing publicly available datasets and the new collected PolyMapper dataset. Experiment results (see Sec. 4) outperform or are are on par with the state-of-the-art, per-pixel instance segmentation methods [18, 28], and recent research that proposes custom-tailored approaches for only one of the tasks, road network prediction [32, 4] or building footprint extraction [38]. Our approach has significant advantage that it generalizes to both, building and road delineation, and could potentially be extended to other objects.

2. Related work

Building segmentation from overhead data has been a core research interest for decades and discussing all works is beyond the scope of this paper [19, 34, 20]. Before the comeback of deep learning, building footprints were often delineated with multi-step, bottom-up approaches and a combination of multi-spectral overhead imagery and airborne LiDAR, e.g., [46, 2]. A modern approach is [6] that applies a fully convolutional neural network to combine evidence from optical overhead imagery and a digital surface model to jointly reason about building footprints. Today, most building footprint delineation from a single image is often approached via semantic segmentation as part of a broader multi-class task and many works exist, e.g., [40, 24, 30, 51, 20, 31]. Microsoft recently extracted all building footprints in the US from aerial images by, first, running semantic segmentation with a CNN and second, refining footprints with a heuristic polygonization approach[1]. A current benchmark challenge that aims at extracting building footprints is [38], which we use to evaluate performance of our approach. Another large-scale dataset that includes both, building footprints and road networks is SpaceNet [49]. All processing takes place in the Amazon Cloud on satellite images of lower resolution than our aerial images in this paper.

Road network extraction in images goes back to (at least) [3], where road pixels were identified using several image processing operations at a local scale. Shortly afterwards [13] was probably the first work to explicitly incorporate topology, by searching for long 1-dimensional structures. One of the most sophisticated methods of the pre-deep learning era was introduced in [47, 23], who center their approach on marked point processes (MPP) that allows them to include elaborate priors on the connectivity and intersection geometry of roads. To the best of our knowledge, the first (non-convolutional) deep learning approach to road network extraction was proposed by [35, 36]. The authors train deep belief network to detect image patches containing roads and second network repairs small network gaps at large scale. [53] propose to model longevity and connectivity of road networks with a higher-order CRF, which is extended in [52] to sampling more flexible, road-like higher-order cliques through collections of shortest paths, and to also model buildings with higher-order cliques in [39]. [33] combine OSM and aerial images to augment maps with additional information like the road width using a MRF formulation, which scales to large regions and achieves good results at several locations world-wide. Two recent works apply deep learning to road center-line extraction in aerial images. DeepRoadMapper [32] introduces a hierarchical processing pipeline that first segments roads with CNNs, encodes end points of street segments as vertices in a graph connected with edges, thins output segments to road center-lines and repairs gaps with an augmented road graph. RoadTracer [4] uses an iterative search process guided by a CNN-based decision function to derive the road network graph directly from the output of the CNN. To the best of our knowledge, [4] is the only work, yet, that completely eliminates the intermediate, explicit pixel-wise image labeling step and outputs road center-lines directly like our method.

Polygon prediction in images has a long history with methods such as level sets [44] or active contour models [21]. While these methods follow an iterative energy minimization scheme and usually are a final component of multi-step, bottom-up workflows (e.g., [7, 15] for road network refinement), directly predicting polygons from images is a relatively new research direction. We are aware of only six works that move away from pixel-wise labeling and directly predict 2D polygons [10, 8, 4, 5, 1, 29]. Interestingly, [10, 5] apply an unsupervised strategy without making use of deep learning and achieve good results for superpixel polygons [10] and polygonal object segmentation [5]. [8] designed a semi-automated approach where a human annotator first provide bounding boxes surrounding an object of interest. A deep-learning approach consisting of an RNN coupled with a CNN, then generates a polygon outlining the target object. A recent extension of this work [1] increases the output resolution by adding a graph neural network (GNN) [43, 25]. This approach, as well as the original work of [8], still relies on user input to provide an initial bounding box around the object of interest, or to correct a predicted vertex of the polygon if needed. [29] extracts

[1]We are not aware of any scientific publication of this work and thus refer the reader to the corresponding GitHub repository that describes the workflow and shares data.

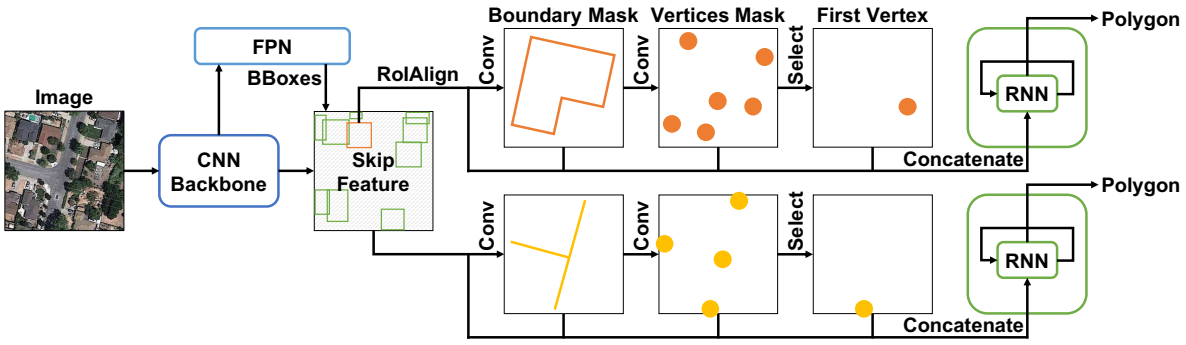


Figure 2: Workflow of our method for both building footprint and road network extraction. The only difference between road and building processing is that we use the corresponding local skip feature via RoIAlign for buildings (bounding boxes provided by FPN), but the entire feature map for roads.

building footprints by formulating active contours as a deep learning task, where a structured loss imposes learned shape priors that refine an initial extraction result.

In summary, prior works mentioned above either focus on pixel level outputs or can only handle just a single type of object. Thus, the absence of direct topological map extraction in the field of remote sensing is what motivates us to develop a fully automated, end-to-end learnable approach to detect geometrical shapes of buildings and roads in a given overhead image.

3. Method

We introduce a new, generic approach for extracting topological map in aerial images using polygons. We first start by discussing the use of polygon representations to describe objects in an image.

3.1. Polygon Representation

We represent objects as polygons. As in [8, 1], we rely on a CNN to find keypoints based on image evidence, which are then connected sequentially by an RNN. A fundamental difference of PolyMapper is that it runs fully automatically without any human intervention in contrast to [8, 1], which were originally designed for speeding up manual object annotation. All the models discussed in [8, 1] (including their "prediction mode") require a user to first draw a bounding box that contains the target object and potentially provide additional manual intervention (*e.g.*, drag/add/delete some keypoints) if the object is not correctly delineated.

We refrain from any manual intervention altogether and propose a fully automated workflow. This is however difficult for mainly two reasons: (1) multiple objects of interest can appear in a given image patch and (2) the shapes of different target objects can significantly vary. For instance, buildings are closed shapes of limited extent in the image while road networks span across entire scenes and are best described with a general graph topology. We therefore present two enhancements to address these problems and then introduce the general pipeline as shown in Fig. 2 for generating object polygons.

3.2. Multiple Targets

Prior work such as [8, 1] is only applicable when a bounding box is provided for each object of interest. These methods are therefore not able to detect objects such as multiple buildings in a given image. We first address the case of buildings by adding a bounding box detection step to partition the image into individual building instances, which allows to compute separate polygons for all buildings. To this end, we have integrated the Feature Pyramid Network (FPN) [26] into our workflow and have made it an end-to-end model. The FPN further enhances the performance of the region proposal network (RPN) used by Faster R-CNN [42] by exploiting the multi-scale, pyramidal hierarchy of CNNs and resulting in a set of so-called feature pyramids. Once images with individual buildings have been generated, the rest of the pipeline follows the generic procedure described in Sec. 3.4.

3.3. From Graphs to Polygons

The inherent topology of objects such as roads or rivers is a general graph instead of a polygon, and the vertices of this graph are not necessarily connected in a sequential manner. In order to reformulate the topology of these objects as a polygon, we follow the principle of a maze solving algorithm, the wall follower, which is also known as the left-/right-hand rule (see Fig. 3): if a maze is simply connected, then by keeping one hand in contact with one wall of the maze, the algorithm is guaranteed to reach an exit.

We apply this principle to extract road sequences. As shown in Fig. 3, the road network can be regarded as a bidirected graph. Each road segment has two directed edges with opposite directions. We assume that for a given pair of directed edges, an edge's partner is always on its left when facing the direction of travel. Suppose we are standing at an arbitrary edge and we travel according to the follow-

ing rules: (1) always walk facing the direction of the edge; (2) turn right when encountering an intersection; (3) turn around when encountering a dead end. Following this set of rules, we arrive back at the starting point after completing a full cycle (see Fig. 3b). Finally, we connect all keypoints on the way (*i.e.*, intersections and dead ends) in the order of traveling in order to obtain a "polygon" (see Fig. 3c). In this way, the vertices that are originally not sequential in the road graph become ordered.

With a larger patch size or denser road networks, multiple polygons can exist as shown in Fig. 4. However, we can only get a single polygon by following the rules described above. In order to get all the polygons in a graph, we need to traverse all the road segments twice (forward and backward). In practice, the sequence generation procedure goes as follows: we first traverse all edges in an arbitrary polygon, and for the directed edges that were not visited, we randomly select one and traverse it following the set of rules until all edges in the graph have been visited.

(a) (b) (c)

Figure 3: Maze wall follower approach to sequentialization of road topology. (a) example aerial view of a T-junction, (b) wall follower sequence, (c) resulting "polygon" with sequence order 1→2→3→2→4→2→1.

Figure 4: Road polygon extraction for a larger patch leading to one outer anticlockwise polygon (orange) and two inner clockwise polygons (blue and green).

3.4. Pipeline

CNN Part For an input image, we first use a VGG-16 without tail layers as the CNN backbone to extract skip features [41] with $\frac{1}{8}$ the size of the input image (see Fig. 2). Meanwhile, the FPN also takes features from different layers of the backbone to construct a feature pyramid and predicts multiple bounding boxes containing the buildings.

For a single building, with the skip feature map and its bounding box, followed by RoIAlign [18], the local features F are obtained. We apply convolutional layers to the feature in order to generate a heat-map mask of building boundaries B that delineate the object of interest. This is followed by additional convolutional layers outputting a mask of candidate keypoints, denoted by V. Both B and V have a size equal to $\frac{1}{8}$ the size of the input image. Among all candidate keypoints, we select those w points with the highest score in V as starting point y_0 (same as y_{-1}, see Fig. 5).

As illustrated in Fig. 2, the main procedure of road network extraction is identical to the case of buildings. We only adapt RoI definition and vertex selection to the road case. While building RoIs are sampled within an image patch, a road RoI corresponds to the entire image patch. Naturally, the generated heatmap B refers to the roads' centerlines instead of building boundaries. Vertex selection is adapted to the road topology by selecting start point candidates at image edges and choosing the one with the highest score as starting point y_0 (same as y_{-1}) to predict the unique outer polygon. Note that each segment of the outer polygon should be passed twice unless the segment is shared with an inner polygon. Thus, after the outer polygon is predicted, we choose two vertices of a segment that is passed only once as y_{-1} and y_0 (in reverse direction) to further predict a potential inner polygon.

RNN Part As illustrated in Fig. 5, the RNN outputs y_t's potential location $P(y_{t+1}|y_t, y_{t-1}, y_0)$ at each step t. We input both, y_t and y_{t-1} to compute the conditional probability distribution of y_{t+1} because it allows defining a unique direction. If given two neighboring vertices with an order in a polygon, the next vertex in this polygon is uniquely determined. Note that the distribution also involves the end signal <eos> (end of sequence), which indicates that the polygon reaches a closed shape and the prediction procedure should come to the end. The final, end vertex in a polygon thus corresponds to the very first, starting vertex y_0, which therefore has to be included at each step.

In practice, we ultimately concatenate F, B, V, y_0 (also y_{-1} for polygon prediction in the case of roads) and feed the resulting tensor to a multi-layer RNN with ConvLSTM [45] cells in order to sequentially predict the vertices that will delineate the object of interest, until it predicts the <eos> symbol. For buildings, we simply connect all sequentially predicted vertices to obtain the final building polygon. In the case of roads, the predicted polygon(s) themselves are not needed directly but rather used as a set of edges between vertices. We thus use all these individual line segments that make up the polygon(s) for further processing. Specifically, each of the predicted segments $\mathbf{e}$ is associated with a score $s_{\mathbf{e}}$ calculated as $s_{\mathbf{e}} = \int_0^1 B(\mathbf{e}(u))\mathrm{d}u \in [0, 1]$, where $\mathbf{e}(u) = ue_1 + (1-u)e_2$, B is the heatmap of centerlines, e_1 and e_2 are the two extremities of $\mathbf{e}$. We remove segments with low scores and connect the remaining segments to form the entire graph.

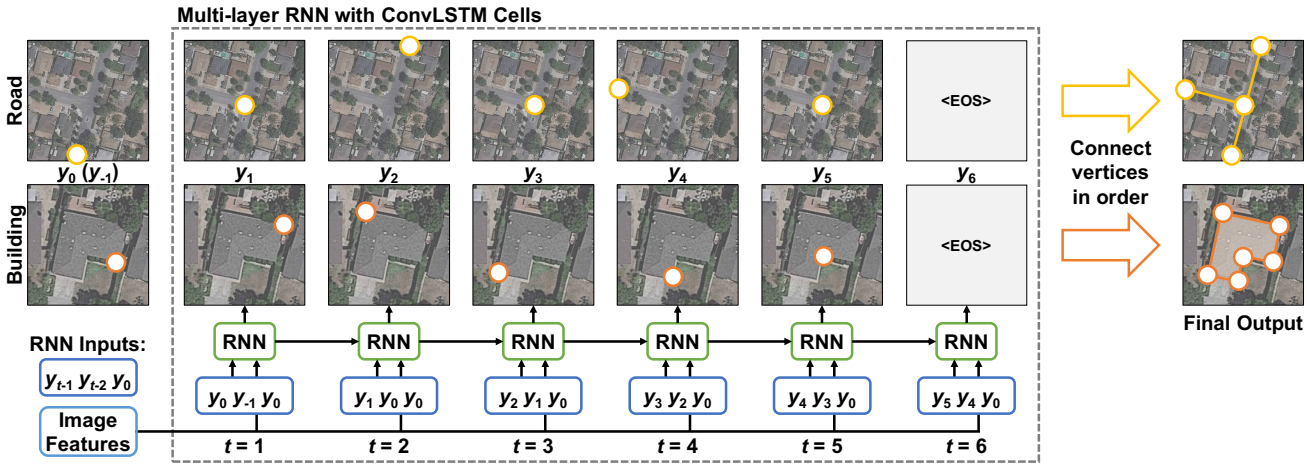


Figure 5: Keypoint sequence prediction produced by RNN for buildings and roads. At each time step t, the RNN takes the current vertex y_t and previous vertex y_{t-1} as input, as well as the first vertex y_0, and outputs a conditional probability distribution $P(y_{t+1}|y_t, y_{t-1}, y_0)$. When the polygon reaches its starting keypoint and becomes a closed shape, the end signal `<eos>` is raised. Note that the RNN also takes features generated by the CNN (see Fig. 2) as input at each time step.

3.5. Implementation Details

We set the model parameters using size 28×28 for F, B, V and y_t, and set the number of layers of the RNN to 3 (buildings) and 4 (roads). The maximum length of a sequence when training is set to be 30 for both cases. The total loss of the building case is a combined loss from the FPN, CNN and RNN parts. The FPN loss consists of a cross-entropy loss for anchor classification and a smooth L1 loss for anchor regression. The CNN loss refers to the log loss for the mask of boundary and vertices, and the RNN loss is the cross-entropy loss for the multi-class classification at each time step. In the road case, the FPN loss is excluded.

For training, we use the Adam [22] optimizer with batch size 4 and an initial learning rate of 0.0001, as well as default β_1 and β_2. We trained our model on 4 GPUs for a day for buildings and 12 hours for roads. During training, we force the order in which we visit the edges of the building polygons to be anticlockwise, while for the road polygons we follow the set of rules described in Sec. 3.3.

In the inference phase, we use beam search with a width w (which is 5 in our experiments). For building, we select top w vertices with highest probability in V as the starting vertices, then followed by a general beam search procedure. Among the w polygon candidates, we choose the one with the highest probability as the output. Similarly, for road, we select vertices at the edge of the image and then choose top w with the highest score as the starting point and follows the general beam search algorithm. After the outer polygon is predicted, we can further predict potential inner polygon(s) as mentioned in Sec. 3.4. Finally, we use a threshold of 0.7 (which was found to yield good results) in our experiments to exclude unmatched edges.

In addition, for the topological map extraction from a relatively large-scale overhead image of a city, we first divide the whole image into several patches with 50% coverage. In the training phase of the building footprints, incomplete footprints at the edge of the image are still be used, however, they are excluded in the inference scheme. In the case of roads, in order to get a complete city road network, some post-processing is performed, such as splicing road networks in adjacent patches, removing small loops of the graph and duplicated vertices and edges.

As for the efficiency, the average inference time on a single GPU is 0.38s for buildings and 0.29s for roads per image patch (300×300 pixels).

4. Experiments

We are not aware of any publicly available dataset[2] that contains labeled building footprints and road networks together with aerial images at large scale and thus create our own dataset (see Sec. 4.3). In order to compare our results to the state-of-the-art, we resort to evaluating building footprint extraction and road network delineation separately on popular task-specific datasets, crowdAI [37] and RoadTracer [4] (see Sec. 4.2).

4.1. Evaluation Measures

For building extraction, we report the standard MS COCO measures including average precision (AP, averaged over IoU thresholds), AP_{50}, AP_{75} and AP_S, AP_M, AP_L (AP at different scales). To measure the proportion of buildings detected by our approach with respect to the ground truth,

[2]Note that the only dataset that contains both, building footprints and road centerlines is SpaceNet [49], which runs on the Amazon Cloud and uses images of lower resolution than ours. In addition, we are not aware of any scientific publication of a state-of-the-art approach that uses it.

Table 1: Buildings extraction results on the crowdAI dataset [37]

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AR	AR_{50}	AR_{75}	AR_S	AR_M	AR_L
Mask R-CNN[18, 38]	41.9	67.5	48.8	12.4	58.1	51.9	47.6	70.8	55.5	18.1	65.2	63.3
PANet[28]	50.7	73.9	62.6	19.8	**68.5**	**65.8**	54.4	74.5	65.2	21.8	73.5	75.0
PolyMapper	**55.7**	**86.0**	**65.1**	**30.7**	**68.5**	58.4	**62.1**	**88.6**	**71.4**	**39.4**	**75.6**	**75.4**

(a) Mask R-CNN [18, 37] (b) PANet [28] (c) PolyMapper

Figure 6: Building footprint extraction results on 2 example patches of the crowdAI dataset [37] achieved with (a) Mask R-CNN [18, 37], (b) PANet [28], and (c) PolyMapper. Note that results in (a) and (b) are images labeled per pixel whereas PolyMapper shows polygons, as well as vertices connected with line segments.

(a) Mask R-CNN (b) PANet (c) PolyMapper

Figure 7: Comparison of pixel-wise semantic segmentation results of Mask R-CNN and PANet with our direct polygon prediction PolyMapper for an example building.

we additionally evaluate average recall (AR), which is not commonly used in previous works such as [18, 28]. Both AP and AR are evaluated using mask IoU. However, we would like to emphasize that in contrast to pixel-wise output masks produced by common methods for building footprint extraction, our outputs are polygon representations of building footprints.

Evaluating the quality of road networks in terms of its topology is a non-trivial problem. [53] propose a connectivity measure SP, which centers on evaluating shortest path distances between randomly chosen point pairs in the road graph. SP generates a large number of pairs of vertices, computes the shortest path between each two vertices in both ground truth and predicted maps, and outputs the fraction of pairs where the predicted length is equal (up to a buffer of 10%) to the ground truth, shorter (erroneous shortcut) or longer (undetected piece of road).

In addition to SP, we propose a new topology evaluation measure that compares shortest paths through graphs [53] using a measure based on average precision (AP) and average recall (AR). This allows an evaluation similar to building footprints and compares ground truth and predicted road graphs in a meaningful way. Similar to the definition in [32], we define the similarity score for the length of two shortest paths, d^* and d, in ground truth and predicted road graphs as a ratio of minimum and maximum values,

$$\mathrm{IoU}(d^*, d) = \mathrm{IoU}(d, d^*) = \frac{\min(d^*, d)}{\max(d^*, d)} \in [0, 1]. \quad (1)$$

Then, with a given IoU threshold t, we can define the weighted precision and recall as follows,

$$\mathrm{AP}^{\mathrm{IoU}=t} = \frac{\sum_i d_i \mathbb{1}[\mathrm{IoU}(d_i, d^*_{j_i}) \geq t]}{\sum_i d_i}, \quad (2)$$

$$\mathrm{AR}^{\mathrm{IoU}=t} = \frac{\sum_j d^*_j \mathbb{1}[\mathrm{IoU}(d^*_j, d_{i_j}) \geq t]}{\sum_j d^*_j}, \quad (3)$$

where $\mathbb{1}[\cdot]$ is the indicator function, d_i and $d^*_{j_i}$ refer to the i-th shortest path in the inferred map and its corresponding shortest path with index j_i in the ground truth graph, similar for d^*_j and d_{i_j}. Note that the shortest path computation is expensive and it is unfeasible to compute all possible paths exhaustively. We thus randomly sample 100 start vertices and sample 1,000 end vertices for each of them, which yields 100,000 shortest paths in total.

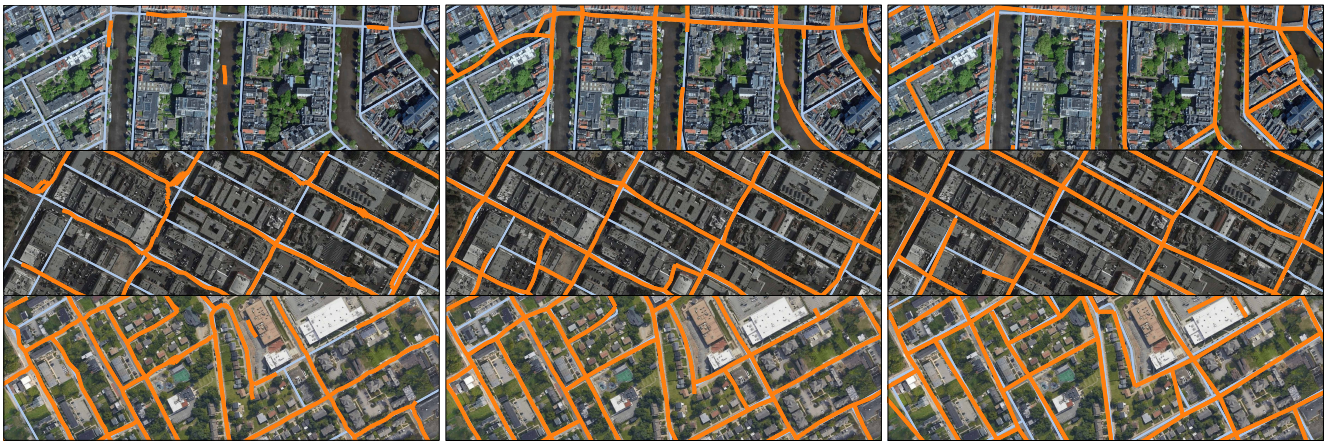
4.2. Comparison to State-of-the-art

Buildings We use the crowdAI dataset [37] to validate the building footprint extraction results and to compare to the state-of-the-arts. This large-scale dataset is split as follows. The training set consists of ∼280,000 images with ∼2,400,000 annotated building footprints. The test set contains ∼60,000 images with ∼515,000 buildings. Each individual building is annotated in a polygon format as a sequence of vertices according to MS COCO [27] standards.

We compare the performance of our model on the crowdAI dataset [37] to state-of-the-art methods Mask R-CNN [18, 38] and PANet [28]. Results in Tab. 1 show

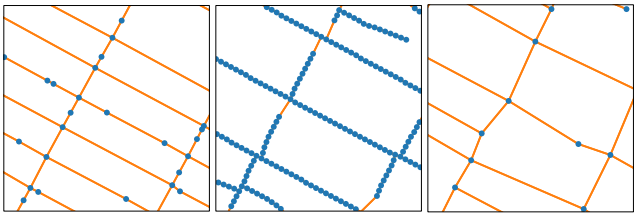
Table 2: Road network extraction results on the RoadTracer dataset [4]

Method	$SP_{\pm 5\%}$	$SP_{\pm 10\%}$	AP_{85}	AP_{90}	AP_{95}	AR_{85}	AR_{90}	AR_{95}
DeepRoadMapper [32]	11.9	15.6	35.9	28.4	19.1	58.2	45.7	27.8
RoadTracer [4]	**47.2**	**61.8**	64.9	56.6	**42.4**	**85.3**	**76.5**	**56.8**
PolyMapper	45.7	61.1	**65.5**	**57.2**	40.7	84.2	74.8	53.7



(a) DeepRoadMapper[32] (b) RoadTracer[4] (c) PolyMapper

Figure 8: Comparison of predicted road network (orange) to ground truth (blue) for subscenes of Amsterdam (top), Los Angeles (middle) and Pittsburgh (bottom) of the RoadTracer dataset [4].



(a) Ground Truth (b) RoadTracer [4] (c) PolyMapper

Figure 9: Visual comparison of graph structures. Vertices are blue and edges are orange.

that PolyMapper outperforms Mask R-CNN and PANet in all AP and AR metrics except AP_L, which refers to large buildings. We hypothesize that the inferior performance observed for large buildings is due to their large feature maps, which leads to more inaccurate location information since a vertex location may be blurred when re-sizing to a fixed size. Fig. 6 and Fig. 7 provides a qualitative comparison of the predictions of the state-of-the-art methods and PolyMapper, where polygons appear to be a more compact representation for buildings. We also see that PolyMapper learns to produce right angles on its own. As future work, we would like to explore whether imposing more geometrical constraints could further improve the results.

Roads To evaluate the road network extraction we use the dataset of [4] tailored for the RoadTracer method. We used their code to download the entire dataset and we trained our model using the same train and test split. Note that we train and test on images from 25 and 15 cities respectively. Our results thus indicate to a certain extent how well an approach generalizes to new scenes.

We compare the results of our method to the state-of-the-art methods DeepRoadMapper [32] and RoadTracer [4]. We directly take the predicted graphs for both models from [4] (who re-implemented [32]) and compute evaluation measures SP, AP and AR as shown in Tab. 2. A visual comparison of the results overlaid on top of the original images is shown in Fig. 8 whereas a comparison of the graph structures is shown in Fig. 9. PolyMapper outperforms DeepRoadMapper[32] in all measures and performs on par with RoadTracer [4].

We visually compare the PolyMapper graph structure to ground truth and RoadTracer [4] in Fig. 9. The road graph representation of PolyMapper is close to the ground truth whereas RoadTracer predicts many more vertices. We compare the overall graph complexity in terms of the total number of vertices and edges in Tab. 3 for 15 cities of the RoadTracer test set. PolyMapper has a much lower graph complexity with ∼87% less vertices and edges than RoadTracer [4] and ∼70% less than DeepRoadMapper [32].

Table 3: Comparison of graph complexity

Method	#Vertices	#Edges
DeepRoadMapper [32]	126,029	118,978
RoadTracer [4]	271,244	281,518
PolyMapper	**31,749**	**35,998**

4.3. Comparison on PolyMapper Dataset

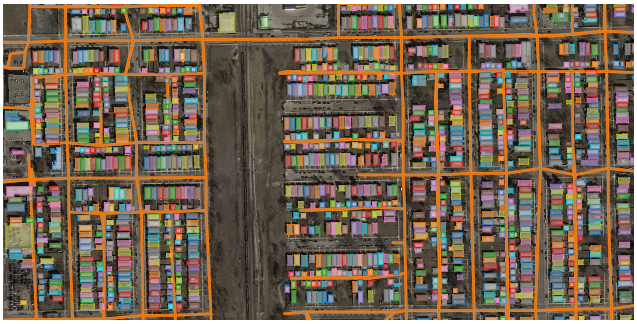
We are not aware of any publicly available dataset used by state-of-the-art methods that contains both annotations

Table 4: Evaluation on the PolyMapper dataset: Buildings

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AR	AR_{50}	AR_{75}	AR_S	AR_M	AR_L
Mask R-CNN[18, 38]	42.0	70.5	**46.7**	24.3	**55.5**	**49.9**	46.6	71.7	53.6	27.6	61.1	**60.4**
PANet[28]	42.1	71.7	46.3	25.5	54.5	47.9	47.0	72.5	54.1	29.1	60.4	57.0
PolyMapper	**44.7**	**80.5**	46.3	**31.5**	54.0	40.5	**52.8**	**84.6**	**58.0**	**39.6**	**62.7**	60.3

Table 5: Evaluation on the PolyMapper dataset: Roads

Method	$SP_{\pm 5\%}$	$SP_{\pm 10\%}$	AP_{85}	AP_{90}	AP_{95}	AR_{85}	AR_{90}	AR_{95}
DeepRoadMapper [32]	48.6	61.6	74.3	61.8	47.8	75.9	63.9	49.4
RoadTracer [4]	65.7	77.7	82.8	75.4	60.2	85.5	78.6	66.2
PolyMapper	**72.8**	**85.3**	**92.4**	**86.5**	**73.7**	**92.4**	**86.3**	**72.6**



(a) Chicago

(b) Sunnyvale

Figure 10: PolyMapper results for (a) Chicago and (b) Sunnyvale. Results for Boston are shown in Fig. 1.

of building footprints and road networks for aerial imagery. Thus we created our own dataset *following the same procedure* used to obtain the crowdAI [37] and RoadTracer [4] datasets. This new dataset contains building footprints and road networks from OSM [17, 16, 14] and aerial images from Google Maps. We collect the dataset of the three US cities Boston, Chicago, and Sunnyvale. We did not choose European cities in this work because many buildings typically share the same roof and polygonal instance segmentation is thus ill-defined (i.e. a single building in the aerial image is often split into multiple instance annotations). As for Asian cities, they usually have a lot of missing annotations in OSM. Our new PolyMapper dataset contains ∼400,000 images and each patch is of size 300×300 pixels and shows zoom level 19 (scale ∼22.57m per pixel) in Google Maps, covering 466.587km^2 with ∼3,000,000 building annotations and 8905.3km of road annotations.

Unlike RoadTracer [4] that trains its model on 25 cities and tests on 15 different cities, we train our method and baselines on each city of the new PolyMapper dataset separately. Testing of models is done on different areas of a city (same strategy as [32]) and a weighted average is computed across cities. Quantitative results are shown in Tab. 4 and 5. We also visualize the final map extraction results for some test regions in Fig. 1, 10a and 10b. For more details about the statistics of the new dataset and experiments, please refer to the supplementary material.

For roads (see Tab. 5), PolyMapper outperforms both, DeepRoadMapper [32] and RoadTracer [4] consistently across all measures (averaged across Boston, Chicago, and Sunnyvale). As for polygon building footprints extraction (see Tab. 4), PolyMapper performs on par with the pixel-wise instance segmentation approaches Mask R-CNN [18] and PANet [28], but for average precision and recall, PolyMapper still outperforms them.

5. Conclusion

We have proposed a novel approach that is able to directly extract topological map from city overhead imagery with a CNN-RNN architecture. We also propose a novel reformulation method that can sequentialize a graph structure as closed polygons to unify the shapes of different types of objects. Our empirical results on a variety of datasets demonstrate high-level of performance for delineating building footprints and road networks using raw aerial images as input. Overall, PolyMapper performs better or on par compared to state-of-the-art methods that are custom-tailored to either building or road networks extraction in pixel level. A favorable property of PolyMapper is that it produces topological structures instead of conventional per-pixel masks, which are much closer to the ones of real online map services, and are more natural and less redundant. We view our framework as a starting point for a new research direction that directly learns high-level, geometrical shape priors from raw input data through deep neural networks to predict vectorized object representations.

References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 859–868, 2018.

[2] Mohammad Awrangjeb, Mehdi Ravanbakhsh, and Clive S. Fraser. Automatic detection of residential buildings using lidar data and multispectral imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(5):457–467, 2010.

[3] Ruzena Bajcsy and Mohamad Tavakoli. Computer recognition of roads from satellite pictures. *IEEE T. Systems, Man, and Cybernetics*, 6(9):623 – 637, 1976.

[4] Favyen Bastani, Songtao He, Mohammad Alizadeh, Hari Balakrishnan, Samuel Madden, Sanjay Chawla, Sofiane Abbar, and David DeWitt. Roadtracer: Automatic extraction of road networks from aerial images. In *Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, June 2018.

[5] Jean-Philippe Bauchet and Florent Lafarge. Kippi: Kinetic polygonal partitioning of images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[6] Ksenia Bittner, Fathalrahman Adam, Shiyong Cui, Marco Körner, and Peter Reinartz. Building footprint extraction from vhr remote sensing images combined with normalized dsms using fused fully convolutional networks. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(8):2615–2629, 2018.

[7] Matthias Butenuth and Christian Heipke. Network snakes: graph-based object delineation with active contour models. *Machine Vision and Applications*, 23(1):91–109, 2012.

[8] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *CVPR*, volume 1, page 2, 2017.

[9] Mauro Dalla Mura, Jon Atli Benediktsson, Björn Waske, and Lorenzo Bruzzone. Morphological attribute profiles for the analysis of very high resolution images. *IEEE Transactions on Geoscience and Remote Sensing*, 48(10):3747–3762, 2010.

[10] Liuyun Duan and Florent Lafarge. Image partitioning into convex polygons. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3119–3127, 2015.

[11] Mark Everingham, S. M. Ali Eslami, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

[12] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[13] Martin A. Fischler, Jay Martin Tenenbaum, and H. C. Wolf. Detection of roads and linear structures in low-resolution aerial imagery using a multisource knowledge integration technique. *Computer Graphics and Image Processing*, 15:201 – 223, 1981.

[14] Jean-Francois Girres and Guillaume Touya. Quality Assessment of the French OpenStreetMap Dataset. *Transactions in GIS*, 14(4):435–459, 2010.

[15] Jens C Goepfert, Franz Rottensteiner, and Christian Heipke. Network Snakes for Adapting GIS Roads to Height Data of Different Data Sources - Performance Analysis Using ALS Data and Stereo Images. In *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume I-3, pages 209–214, 2012.

[16] Mordechai Haklay. How Good is Volunteered Geographical Information? A Comparative Study of OpenStreetMap and Ordnance Survey Datasets. *Environment and Planning B: Urban Analytics and City Science*, 37(4):682–703, 2010.

[17] Mordechai Haklay and Patrick Weber. OpenStreetMap: User-Generated Street Maps. *IEEE Pervasive Computing*, 7(4):12–18, 2008.

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[19] Christian Heipke, Hélène Mayer, Christian Wiedemann, and Olivier Jamet. Evaluation of automatic road extraction. In *3D Reconstruction and Modeling of Topographic Objects*, 1997.

[20] Pascal Kaiser, Jan Dirk Wegner, Aurélien Lucchi, Martin Jaggi, Thomas Hofmann, and Konrad Schindler. Learning aerial image segmentation from online maps. *IEEE Transactions on Geoscience and Remote Sensing*, 55(11):6054–6068, 2017.

[21] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[23] Caroline Lacoste, Xavier Descombes, and Josiane Zerubia. Point Processes for unsupervised line network extraction in remote sensing. *PAMI*, 27(10):1568 – 1579, 2005.

[24] Adrien Lagrange, Bertrand Le Saux, Anne Beaupere, Alexandre Boulch, Adrien Chan-Hon-Tong, Stéphane Herbin, Hicham Randrianarivo, and Marin Ferecatu. Benchmarking classification of earth-observation data: from learning explicit features to convolutional networks. In *International Geoscience and Remote Sensing Symposium (IGARSS)*, 2015.

[25] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. Gated graph sequence neural networks. In *ICLR*, 2016.

[26] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 936–944, 2017.

[27] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollr, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

[28] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Pro-*

ceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[29] Diego Marcos, Devis Tuia, Benjamin Kellenberger, Lisa Zhang, Min Bai, Renjie Liao, and Raquel Urtasun. Learning deep structured active contours end-to-end. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8877–8885, 2018.

[30] Dimitrios Marmanis, Konrad Schindler, Jan Dirk Wegner, and Silvano Galliani. Semantic segmentation of aerial images with an ensemble of cnns. *ISPRS Annals – ISPRS Congress*, 2016.

[31] Dimitris Marmanis, Konrad Schindler, Jan Dirk Wegner, Silvano Galliani, Mihai Datcu, and Uwe Stilla. Classification with an edge: improving semantic image segmentation with boundary detection. *ISPRS Journal of Photogrammetry and Remote Sensing*, 135:158–172, 2018.

[32] Gellert Máttyus, Wenjie Luo, and Raquel Urtasun. Deeproadmapper: Extracting road topology from aerial images. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[33] Gellért Máttyus, Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Enhancing road maps by parsing aerial images around the world. In *International Computer Vision Conference*, pages 1689–1697, 2015.

[34] Helmut Mayer, Stefan Hinz, Uwe Bacher, and Emmanuel Baltsavias. A test of automatic road extraction approaches. In *IAPRS*, volume 36(3), pages 209 – 214, 2006.

[35] Volodymyr Mnih and Geoffrey E. Hinton. Learning to detect roads in high-resolution aerial images. In *European Conference on Computer Vision*, 2010.

[36] Volodymyr Mnih and Geoffrey E. Hinton. Learning to label aerial images from noisy data. In *International Conference on Machine Learning*, 2012.

[37] Sharada Prasanna Mohanty. Crowdai dataset. `https://www.crowdai.org/challenges/mapping-challenge/dataset_files`, 2018.

[38] Sharada Prasanna Mohanty. Crowdai mapping challenge 2018: Baseline with mask rcnn. `https://github.com/crowdai/crowdai-mapping-challenge-mask-rcnn`, 2018.

[39] Javier A. Montoya-Zegarraa, Jan Dirk Wegner, Lubor Ladickyb, and Konrad Schindlera. Semantic segmentation of aerial images in urban areas with class-specific higher-order cliques. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume II(3/W4), pages 127 – 133, 2015.

[40] Sakrapee Paisitkriangkrai, Jamie Sherrah, Pranam Janney, and Anton van den Hengel. Effective semantic pixel labelling with convolutional networks and conditional random fields. In *CVPRws*, 2015.

[41] Pedro O Pinheiro, Tsung-Yi Lin, Ronan Collobert, and Piotr Dollár. Learning to refine object segments. In *European Conference on Computer Vision*, pages 75–91. Springer, 2016.

[42] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[43] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *Trans. Neur. Netw.*, 20(1):61–80, Jan. 2009.

[44] James Albert Sethian. *Level Set Methods*. Cambridge University Press, The Edinburgh Building, Cambridge CB2 2RU, UK, 1 edition, 1996.

[45] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 802–810, Cambridge, MA, USA, 2015. MIT Press.

[46] Gunho Sohn and Ian Dowman. Data fusion of high-resolution satellite imagery and lidar data for automatic building extraction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 62:43–63, 2007.

[47] Radu Stoica, Xavier Descombes, and Josiane Zerubia. A Gibbs Point Process for road extraction from remotely sensed images. *IJCV*, 57(2):121 – 136, 2004.

[48] Piotr Tokarczyk, Jan Dirk Wegner, Stefan Walk, and Konrad Schindler. Beyond hand-crafted features in remote sensing. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume II-3/W1, pages 35–40, 2013.

[49] Adam van Etten, Dave Lindenbaum, and Todd M. Bacastow. Spacenet: A remote sensing dataset and challenge series. *arXiv*, arXiv:1807.01232v2:1–21, 2018.

[50] Michele Volpi and Vittorio Ferrari. Semantic segmentation of urban scenes by learning local class interactions. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–9, 2015.

[51] Michele Volpi and Devis Tuia. Dense semantic labeling of subdecimeter resolution images with convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 55(2):881–893, 2017.

[52] Jan Dirk Wegner, Javier Alexander Montoya-Zegarra, , and Konrad Schindler. Road networks as collections of minimum cost paths. *ISPRS Journal of Photogrammetry and Remote Sensing*, 108:128 – 137, 2015.

[53] Jan Dirk Wegner, Javier A. Montoya-Zegarra, and Konrad Schindler. A higher-order crf model for road network extraction. In *CVPR*, pages 1698–1705, 2013.