

PolyMapper: Extracting City Maps using Polygons

Zuoyue Li

Department of Computer Science
ETH Zürich, Switzerland
zuli@student.ethz.ch

Jan Dirk Wegner

EcoVision Lab, PRS Group
ETH Zürich, Switzerland
jan.wegner@geod.baug.ethz.ch

Aurélien Lucchi

Department of Computer Science
ETH Zürich, Switzerland
aurelien.lucchi@inf.ethz.ch

Abstract

We propose a method to leapfrog pixel-wise, semantic segmentation of (aerial) images and predict objects in a vector representation directly. *PolyMapper* predicts maps of cities from aerial images as collections of polygons with a learnable framework. Instead of the usual multi-step procedure of semantic segmentation, shape improvement, conversion to polygons, and polygon refinement, our approach learns mappings with a single network architecture and directly outputs maps. We demonstrate that our method is capable of drawing polygons of buildings and road networks that very closely approximate the structure of existing on-line maps such as OpenStreetMap, and it does so in a fully automated manner. Validation on existing and novel large scale data sets of several cities show that our approach achieves good levels of performance.

1. Introduction

A fundamental research task in computer vision is pixel-accurate semantic image segmentation, where steady progress has been measured with benchmark challenges such as [24, 11, 10]. However, current pixel-wise labeling methods are rather inflexible in that they typically take a pixel image as input and assign a label to each pixel describing what category it does belong to, thus yielding a (labeled) image as output. Our interest in this paper is in developing a more flexible method that, from an input image, produces a graph representation that directly describes geometric objects using a vector data structure. Motivated by the success of recent works [9, 7, 5, 1], we avoid explicit pixel-wise labeling altogether, but instead directly predict polygons from images in an end-to-end learnable approach.

Our research is motivated by the insight that for many applications, semantic segmentation is just an intermediate step of a more comprehensive work-flow 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 fo-



Figure 1: *PolyMapper* result for Sunnyvale overlaid on top of the original aerial imagery. Buildings and roads are directly predicted as polygons using the *PolyMapper* network architecture. See additional results in Fig. 6.

cused on aerial image segmentation such as [8, 43, 45, 27, 18, 46, 26]. We make this application our core scenario because we have access to virtually unlimited data from OpenStreetMap [15, 14, 13] (OSM) and high-resolution RGB orthophotos from Google Maps. Usually, a full mapping pipeline consists of pre-processing the raw aerial imagery (e.g., bundle block adjustment, orthorectification, etc.), converting the orthophoto to a semantically meaningful raster map (i.e., semantic segmentation), followed by further processing like object shape refinement, vectorization, and map generalization techniques. Here, we turn this multi-step workflow into a unified, end-to-end learnable deep learning architecture, *PolyMapper*, which outputs maps of buildings and roads directly, given aerial imagery as input.

Our method is inspired by the recent success of the sequential PolygonRNN [7] and PolygonRNN++ [1] approaches. However, instead of a semi-automated approach where a human annotator provides bounding boxes and segments objects manually, we propose a fully automated workflow without any human intervention by adopting a feature pyramid network (FPN) [23, 16]. Object detection, instance segmentation, and vectorization are thus accomplished within a single, end-to-end learnable approach that relies on modern CNNs architectures and RNNs with convolutional long-short term memory [40] (ConvLSTM) mod-

ules. The CNN part takes as input a city tile and extracts keypoints and edge evidence of building footprints and roads, which are fed sequentially to the ConvLSTM [40] modules. The latter produces a vector representation for each object in a given tile. Finally, the roads from different tiles are connected and combined with the buildings to form a complete city map. A *PolyMapper* result for Sunnyvale is shown in Fig. 1, while the results of Boston and Chicago are illustrated in Fig. 6.

We validate our approach for the automated mapping of road networks and building footprints on a new *PolyMapper* dataset as well as on existing datasets. Our results are on par with the state-of-the-art instance segmentation methods [16, 25], and recent research that proposes custom-tailored approaches for only one of the tasks, road network prediction [29, 4] or building footprint extraction [34]. Our unified approach has the significant advantage that it generalizes to both, building and road delineation, and it 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 [17, 30, 18]. 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., [41, 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, for example, [36, 21, 27, 46, 18, 26]. Microsoft recently extracted all buildings footprints in the US from aerial images by, first, running semantic segmentation with a CNN and second, refining footprints with a heuristic polygonization approach¹. A current benchmark challenge that aims at extracting building footprints is [34], which we use to evaluate performance of our approach. Another large-scale dataset that includes both, building footprints and road networks is SpaceNet [44]. 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 [12] was probably the first work to explicitly incor-

porate topology, by searching for long 1-dimensional structures. One of the most sophisticated methods of the pre-deep learning era was introduced in [42, 20], 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 [31, 32]. The authors train deep belief network to detect image patches containing roads and second network repairs small network gaps at large scale. [47] propose to model longevity and connectivity of road networks with a higher-order CRF, which is extended in [48] to sampling more flexible, road-like higher-order cliques through collections of shortest paths, and to also model buildings with higher-order cliques in [35]. [28] 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 [29] 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.

Direct polygon prediction in images is a relatively new research direction. We are aware of only five works that move away from pixel-wise labeling and directly predict 2D polygons [9, 7, 4, 5, 1]. Interestingly, [9, 5] apply an unsupervised strategy without making use of deep learning and achieve good results for super-pixel polygons [9] and polygonal object segmentation [5]. [7] design an end-to-end learnable workflow that consists of an RNN intertwined with a CNN, which generates a polygon outlining the target object within a user-defined bounding box. The CNN part learns image evidence, such as keypoints and connecting edges, which are connected sequentially by the RNN to come up with a closed object polygon. A recent extension of this work [1] increases the output resolution by adding a graph neural network [39, 22] (GNN). This approach as well as the original work of [7] - 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. Our aim is to instead develop a fully automated approach to detect multiple objects in a given image.

¹We are not aware of any scientific publication of this work and thus refer the reader to the corresponding github website that describes the workflow and shares data: <https://github.com/Microsoft/USBuildingFootprints>.

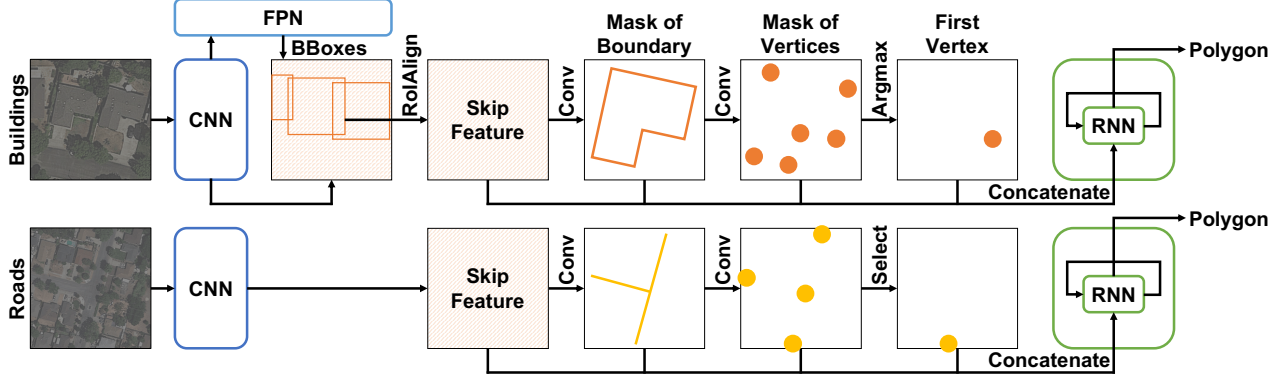


Figure 2: Workflow of our method for buildings (top) and road topology (bottom). The only difference between these two pipelines is that the top one for buildings has an additional feature pyramid network (FPN) for object detection to generate bounding boxes containing individual buildings.

3. Method

We introduce a new, generic approach for extracting arbitrary objects in aerial images using polygons. We first discuss the use of polygon representations to describe such objects and then turn our discussion to the specific cases of buildings and roads.

3.1. Polygon Representation

We represent objects as polygons. As in [7, 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 [7, 1], which were originally designed for speeding up *manual* object annotation. [7, 1] requires 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*. What makes formulating a concise solution hard for both, buildings and road networks are their different shape properties. 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 thus have to resort to different keypoint connecting strategies to handle buildings and roads with the same architecture as shown in Fig. 2.

3.2. Buildings

Although a building can be directly described as a polygon, as mentioned above, Polygon-RNN [7, 1] is only applicable when the bounding box containing the object is given. Thus, it can not extract multiple buildings in an image. As illustrated in Fig. 2 (top), we add bounding box detection to partition the image into individual building instances, which allows to compute separate polygons for all

buildings. To this end, we add a Feature Pyramid Network (FPN) [23] to our workflow. The FPN further enhances the performance of the original region proposal network (RPN) used by Faster R-CNN [38] 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 Section 3.4.

3.3. Roads

The topology of roads 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 road topology as a polygon, we follow the principle of a maze solving algorithm, the wall follower, which is also known as the left-hand rule or the right-hand rule (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.



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

The road topology in a small part of a city (Fig. 3a) can be viewed as a *simply connected maze*. Suppose we are entering this region from a road entrance at the edge of the image and we always travel in accordance with the following rules: (1) always walk on the right side of the road;

(2) turn right when encountering an intersection; (3) turn around if encountering a dead end or another road entrance which lies at the edge of the image. Following this rule-set, we will finally arrive back at the starting point after completing a full cycle (Fig. 3b). We only have to connect all keypoints on the way (i.e., intersections, dead ends and entrances/exits) in the order of traveling and we end up with an "empty" polygon (i.e., a polygon with zero area) most of the time (Fig. 3c). In this way, the vertices that are originally not sequential in the road graph become ordered (typically anti-clockwise).

With an increasing patch size or denser road networks, multiple polygons can exist as shown in Fig. 4. Following the rule-set, the *empty* polygon assumption does no longer hold. Instead, we obtain a unique *outer* polygon with anti-clockwise ordering and (at least one) *inner* polygon(s) with clockwise ordering. Still, the order of the keypoint sequence of each polygon is uniquely determined (Fig. 4) and thus can be used for training the RNN.



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

3.4. Pipeline

CNN for Buildings For each input image, we first use a VGG-16 without tail layers like [7] to extract a set of skip features [37] with $\frac{1}{8}$ the size of the input image (Fig. 2 (top)). Meanwhile, the FPN also takes features from different layers of VGG-16 to construct a feature pyramid to output several bounding boxes. With these two steps together, followed by RoIAlign [16], the local skip features F can be then 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 the one with the highest score in V as starting point y_0 (same as y_{-1}).

CNN for Roads As illustrated in Fig. 2 (bottom), 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 (Fig. 2 (top)), 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 edge of the outer polygon should be passed twice (one forward, one backward) unless the edge is shared with an inner polygon. Thus, after the outer polygon is predicted, we choose two vertices of an edge that is passed only once as y_{-1} and y_0 (in reverse direction) to further predict a potential inner polygon.

RNN 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 $\langle \text{eos} \rangle$, which indicates that the polygon reaches a closed shape. 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 inner polygon prediction) and feed the resulting tensor to a multi-layer RNN with ConvLSTM cells in order to sequentially predict the vertices that will delineate the object of interest, until it predicts the $\langle \text{eos} \rangle$ (end of sequence) 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. Each of the predicted segments e is associated with a score s_e calculated as $s_e = \int_0^1 B(e(u))du \in [0, 1]$, where B is the heatmap of centerlines, $e(u) = ue_1 + (1-u)e_2$, e_1 and e_2 are the two extremities of e . We remove segments with low scores (we used 0.7 in our experiments which was found to yield good results) and connect the remaining segments to form the entire graph.

3.5. Implementation Details

For model parameters, we follow [7] using size 28×28 for all F, B, V, y_t , but increase the number of layers of 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 pipeline 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

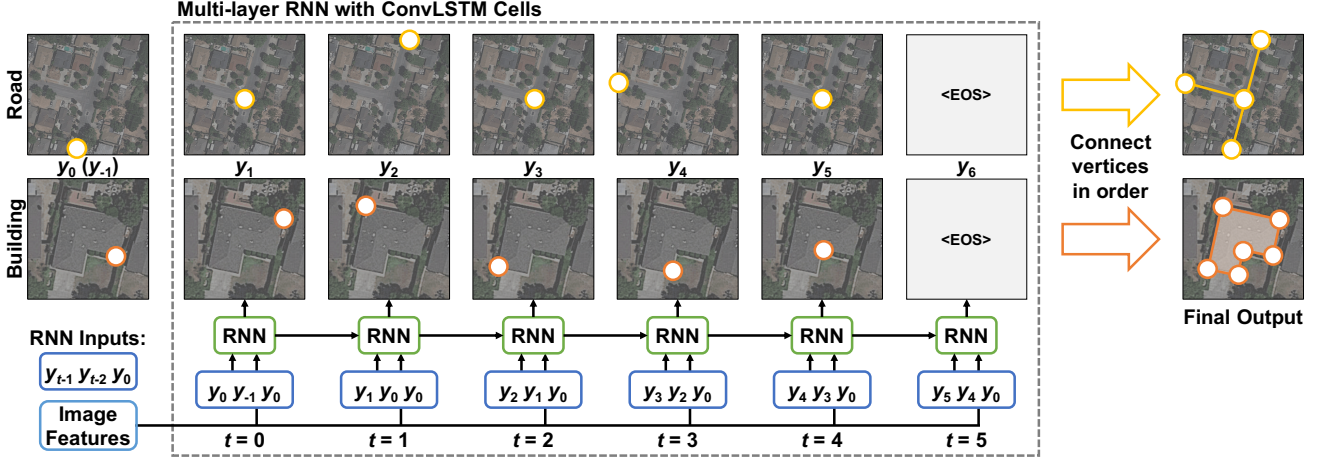


Figure 5: Keypoint sequence prediction produced by RNN for buildings and roads. At each time step t , the RNN takes the current vertex y_t and previous vertex y_{t-1} as input, as well as the first vertex y_0 , and outputs a conditional probability distribution $P(y_{t+1} | y_t, y_{t-1}, y_0)$. When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

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. As for the road pipeline, the loss remains the same but the FPN is excluded.

For training, we use the Adam [19] optimizer with batch size 4 for both roads and buildings and an initial learning rate of $1e^{-4}$, as well as $\beta_1 = 0.9, \beta_2 = 0.999$. We trained our model on 4 GPUs for a day for buildings and 12 hours for roads. During training we force the order of building polygons and the *outer* road polygons to be anti-clockwise, while the order of *inner* road polygons must be clockwise.

In the inference phase, similar to [1] we use beam search with a width w ($w = 5$ 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 Section 3.4.

4. Experiments

We are not aware of any publicly available dataset that contains labeled building footprints and road networks together with aerial images at large scale and thus create our own dataset². This new dataset contains building footprints

²Note that the only dataset that contains both, building footprints and road centerlines is SpaceNet [44], which runs on the Amazon Cloud and uses satellite images of lower resolution than our aerial images. In addition, we are not aware of any scientific publication of a state-of-the-art approach that uses it.

and road networks from OSM [15, 14, 13] and Google Maps aerial images of three US cities, namely Boston, Chicago, and Sunnyvale. [18] find that the lower labeling accuracy of OSM maps (i.e., slight mismatches to Google Maps aerial imagery especially at object boundaries) can be compensated to a large part by using massive amounts of training data, which encourages us to generate an OSM/Google Maps aerial imagery dataset. In total, this new dataset contains 300,000 training images and 15,000 test images, 100,000 per city (Boston, Sunnyvale, Chicago) for training, and 5,000 for testing. Each image patch is of size 300×300 pixels and shows zoom level 19 in Google Maps. 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 different datasets, crowdAI [33] and RoadTracer [4].

4.1. Evaluation Measures

For building extraction, we report the standard MS COCO measures including mean of average precision (mAP, 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, we additionally evaluate average recall (AR), which is not commonly used in previous works such as [16, 25]. Both AP and AR are evaluating using mask IoU. However, we would like to emphasize that in contrast to output images 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. [47] propose a connectivity measure SP, which centers on evaluating shortest path

Table 1: Evaluation on the *PolyMapper* dataset: Buildings

City	mAP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L	mAR	AR ₅₀	AR ₇₅	AR _S	AR _M	AR _L
Boston	21.9	58.8	9.4	10.0	27.9	35.9	31.0	66.2	24.5	14.4	39.2	53.9
Chicago	48.5	85.8	51.2	38.0	58.4	40.3	55.6	88.0	62.2	45.7	65.7	59.8
Sunnyvale	45.7	76.2	50.1	17.5	54.9	32.5	54.2	81.8	61.0	25.9	63.3	57.3

Table 2: Evaluation on the *PolyMapper* dataset: Roads

City	SP _{±5%}	SP _{±10%}	AP ₈₅	AP ₉₀	AP ₉₅	AR ₈₅	AR ₉₀	AR ₉₅
Boston	59.6	80.0	88.0	81.2	61.3	87.8	80.7	59.1
Chicago	89.8	95.5	98.4	96.0	90.1	98.5	96.0	88.5
Sunnyvale	69.1	80.3	90.8	82.2	69.8	90.8	82.1	70.2

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 maps 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 based on the core idea of comparing shortest paths through graphs [47] and associate it with 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 [29], we define the similarity score for the length of two shortest paths, d^* and d , in ground truth and predicted road graphs respectively as a ratio of minimum and maximum values:

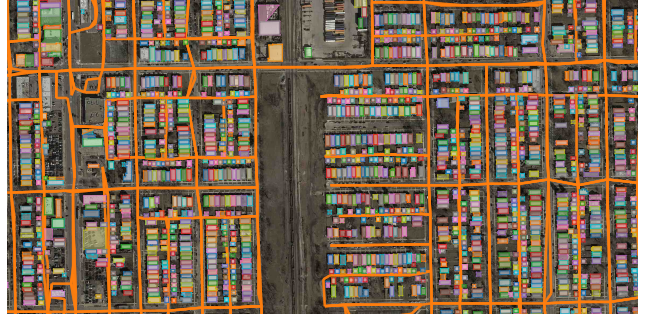
$$\text{IoU}(d^*, d) = \text{IoU}(d, d^*) = \frac{\min(d^*, d)}{\max(d^*, d)} \in [0, 1]. \quad (1)$$

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

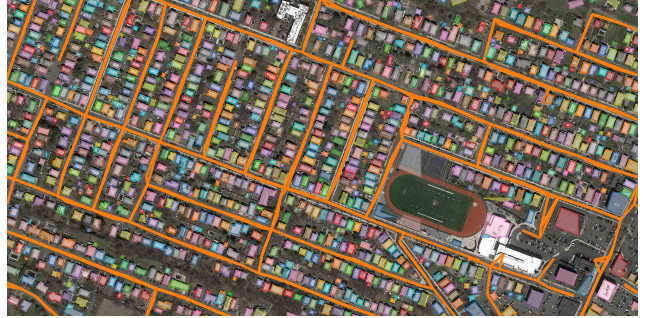
$$\text{AP}^{\text{IoU}=t} = \frac{\sum_i d_i \mathbb{I}[\text{IoU}(d_i, d_{j_i}^*) \geq t]}{\sum_i d_i}, \quad (2)$$

$$\text{AR}^{\text{IoU}=t} = \frac{\sum_j d_j^* \mathbb{I}[\text{IoU}(d_j^*, d_{i_j}) \geq t]}{\sum_j d_j^*}, \quad (3)$$

where $\mathbb{I}[\cdot]$ is the indicator function, d_i and $d_{j_i}^*$ refer to the i -th shortest path in the inferred map its corresponding shortest path with index j_i in the ground truth graph, similar for d_j^* and d_{i_j} . In comparison to the original SP measure, this new measure allows estimates of precision and recall similar to what is done for buildings. 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. This yields 100,000 shortest paths in total that we use to compute AP and AR.



(a) Chicago



(b) Boston

Figure 6: *PolyMapper* results for (a) Chicago and (b) Boston. Results for Sunnyvale are shown in Fig. 1. Roads are represented as polylines and buildings as polygons.

4.2. Results

We evaluate our approach on the new *PolyMapper* dataset of Chicago, Boston, and Sunnyvale. Quantitative results for buildings are shown in Tab. 1 and for road network extraction in Tab. 2. We visualize results for a test region of Sunnyvale in Fig. 1, Chicago in Fig. 6a and Boston in Fig. 6b. In general, *PolyMapper* achieves the best results on Chicago with $\text{AP}_{50} = 85.8$ & $\text{AR}_{50} = 88.0$ for buildings and $\text{AP}_{85} = 98.4$ & $\text{AR}_{85} = 98.5$ for roads. Also, the original SP measure is highest for Chicago ($\text{SP}_{\pm 5\%} = 89.8$ & $\text{SP}_{\pm 10\%} = 95.5$). The results on Sunnyvale are close, while Boston seems a much harder case. All measures drop to $\text{AP}_{50} = 58.8$ & $\text{AR}_{50} = 66.2$ for buildings and

Table 3: Buildings extraction results on the crowdAI dataset [33]

Method	mAP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L	mAR	AR ₅₀	AR ₇₅	AR _S	AR _M	AR _L
Mask R-CNN[16, 34]	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[25]	50.7	73.9	62.6	19.8	68.5	65.8	54.4	74.5	65.2	21.8	73.5	75.0
<i>PolyMapper</i>	55.7	86.0	65.1	30.7	68.5	58.4	62.1	88.6	71.4	39.4	75.6	75.4



Figure 7: Building footprint extraction results on 4 example patches of the crowdAI dataset [33] achieved with (a) Mask R-CNN [16, 33], (b) PANet [25], 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.

$AP_{85} = 88.0$ & $AR_{85} = 87.8$ for roads. In particular, $SP_{\pm 5\%} = 59.6$ is 30 percent points inferior to the result for Chicago. Boston seems a much harder case due to its road network layout that is much less grid-structured in comparison to Chicago and Sunnyvale while Boston buildings show a larger variety of shapes.

4.3. Comparison to State-of-the-art

Buildings We use the crowdAI dataset [33] to validate the building footprint extraction results and to compare to the state-of-the-art. This large-scale dataset is split into a train and a test set. The training set consists of $\approx 281,000$ images with $\approx 2.4 \times 10^6$ annotated building footprints. The test set contains $\approx 60,000$ images with $\approx 515,000$ buildings. Each individual building is annotated in a polygon format as a sequence of vertices according to MS COCO [24] standards. For all experiments on this dataset, we train our model using exactly the same train and test split as competing methods [34].

We compare the performance of our model on the crowdAI dataset [33] to state-of-the-art methods Mask R-

CNN [16] and PANet [25]. Results in Tab. 3 show that our (*PolyMapper*) outperforms Mask R-CNN in all AP and AR metrics. *PolyMapper* outperforms PANet in all AP and AR metrics except AP_L , which refers to large buildings. It turns out that the distribution of buildings into small (S), medium (M), and large (L) is 37.1%, 60.6%, 2.3%. Since only a tiny portion (2.3%) of all buildings falls into the large category, we view inferior performance (by 7.4 percent points) for AP_L as acceptable. Interestingly, *PolyMapper* achieves better building extraction results on crowdAI than on the *PolyMapper* dataset. Both, AP and AR improve by ≈ 7 percent points over the best results on Chicago. Fig. 7 provides a qualitative comparison of the predictions of the state-of-the-art methods and *PolyMapper*, where we can conclude that the polygon is a better and more compact representation for building footprints.

Roads To evaluate the road network extraction we use the dataset of [4] tailored for the RoadTracer method, which combines Google Maps imagery with road networks from OSM. We apply their scripts to download the entire dataset and train our model using the same train and test split.

Table 4: 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 [29]	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

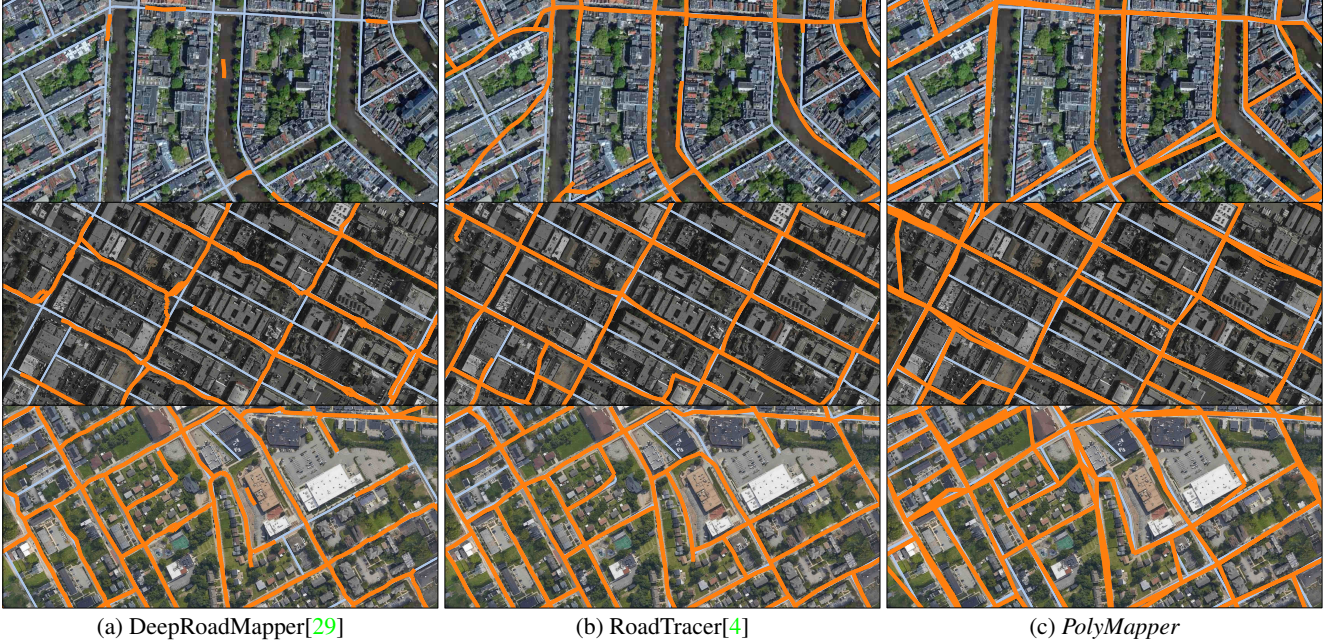


Figure 8: Comparison of predicted road network (orange) to ground truth (blue) for subscenes of Amsterdam (top), Los Angeles (middle) and Pittsburgh (bottom) of the RoadTracer dataset [4].

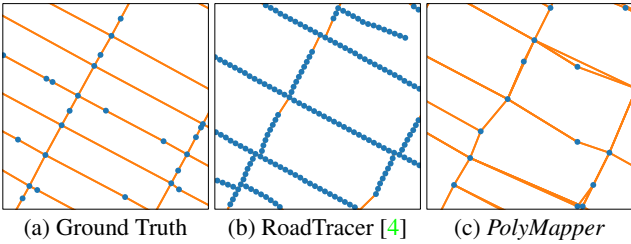


Figure 9: Visual comparison graph structures (vertices blue and edges orange): (a) ground truth, (b) result of RoadTracer [4], (c) PolyMapper.

An interesting property closer to a realistic application setting is that training and testing are done on imagery of different cities. Our results thus indicate to a certain extent how well an approach generalizes to new scenes. We compare the results of our method to state-of-the-art methods DeepRoadMapper[29] and RoadTracer[4]. We directly take the predicted graphs for both models from [4] (who re-implemented [29]) and compute evaluation measures SP, AP and AR as shown in Tab. 4. A visual comparison of results overlaid on top of the original images is shown in Fig. 8 whereas a comparison of the graph structures is illustrated in Fig. 9. *PolyMapper* outperforms DeepRoadMapper[29] in all measures and performs on par

with RoadTracer [4]. We visually compare the *PolyMapper* graph structure to the ground truth and RoadTracer [4] in Fig. 9. *PolyMapper* shows a structure close to the OSM ground truth in terms of its road graph representation whereas RoadTracer predicts many more vertices.

5. Conclusion

We have proposed a novel, unified approach that directly outputs polygons with a CNN-RNN architecture. Our empirical results on a variety of datasets demonstrates 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 network extraction. A favorable property of *PolyMapper* is that it produces graph structures which are close to the ones of real online maps such as OSM. 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 such as images or 3D point clouds to predict vectorized object representations for the mobile internet. We expect that incorporating such high-level priors could further enhance the performance of *PolyMapper*.

References

- [1] D. Acuna, H. Ling, A. Kar, and S. 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. 1, 2, 3, 5
- [2] M. Awrangjeb, M. Ravanbakhsh, and C. Fraser. Automatic detection of residential buildings using lidar data and multispectral imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(5):457–467, 2010. 2
- [3] R. Bajcsy and M. Tavakoli. Computer recognition of roads from satellite pictures. *IEEE T. Systems, Man, and Cybernetics*, 6(9):623 – 637, 1976. 2
- [4] F. Bastani, S. He, M. Alizadeh, H. Balakrishnan, S. Madden, S. Chawla, S. Abbar, and D. DeWitt. RoadTracer: Automatic Extraction of Road Networks from Aerial Images. In *Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, June 2018. 2, 5, 7, 8
- [5] J.-P. Bauchet and F. Lafarge. Kippi: Kinetic polygonal partitioning of images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2
- [6] K. Bittner, F. Adam, S. Cui, M. Körner, and P. 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. 2
- [7] L. Castrejon, K. Kundu, R. Urtasun, and S. Fidler. Annotating object instances with a polygon-rnn. In *CVPR*, volume 1, page 2, 2017. 1, 2, 3, 4
- [8] M. Dalla Mura, J. Benediktsson, B. Waske, and L. Bruzzone. Morphological attribute profiles for the analysis of very high resolution images. *IEEE Transactions on Geoscience and Remote Sensing*, 48(10):3747–3762, 2010. 1
- [9] L. Duan and F. Lafarge. Image partitioning into convex polygons. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3119–3127, 2015. 1, 2
- [10] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015. 1
- [11] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 1
- [12] M. Fischler, J. Tenenbaum, and H. 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. 2
- [13] J.-F. Girres and G. Touya. Quality Assessment of the French OpenStreetMap Dataset. *Transactions in GIS*, 14(4):435–459, 2010. 1, 5
- [14] M. 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. 1, 5
- [15] M. Haklay and P. Weber. OpenStreetMap: User-Generated Street Maps. *IEEE Pervasive Computing*, 7(4):12–18, 2008. 1, 5
- [16] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017. 1, 2, 4, 5, 7
- [17] C. Heipke, H. Mayer, and C. Wiedemann. Evaluation of automatic road extraction. In *3D Reconstruction and Modeling of Topographic Objects*, 1997. 2
- [18] P. Kaiser, J. D. Wegner, A. Lucchi, M. Jaggi, T. Hofmann, and K. Schindler. Learning aerial image segmentation from online maps. *IEEE Transactions on Geoscience and Remote Sensing*, 55(11):6054–6068, 2017. 1, 2, 5
- [19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 5
- [20] C. Lacoste, X. Descombes, and J. Zerubia. Point Processes for unsupervised line network extraction in remote sensing. *PAMI*, 27(10):1568 – 1579, 2005. 2
- [21] A. Lagrange, B. Le Saux, A. Beupere, A. Boulch, A. Chan-Hon-Tong, S. Herbin, H. Randrianarivo, and M. Ferecatu. Benchmarking classification of earth-observation data: from learning explicit features to convolutional networks. In *International Geoscience and Remote Sensing Symposium (IGARSS)*, 2015. 2
- [22] Y. Li, D. Tarlow, M. Brockschmidt, and R. S. Zemel. Gated graph sequence neural networks. In *ICLR*, 2016. 2
- [23] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 936–944, 2017. 1, 3
- [24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014. 1, 7
- [25] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 5, 7
- [26] D. Marmanis, K. Schindler, J. Wegner, S. Galliani, M. Datcu, and U. Stilla. Classification with an edge: improving semantic image segmentation with boundary detection. *ISPRS Journal of Photogrammetry and Remote Sensing*, 135:158–172, 2018. 1, 2
- [27] D. Marmanis, K. Schindler, J. D. Wegner, and S. Galliani. Semantic segmentation of aerial images with an ensemble of cnns. *ISPRS Annals – ISPRS Congress*, 2016. 1, 2
- [28] G. Mátyus, S. W. anf Sanja Fidler, and R. Urtasun. Enhancing road maps by parsing aerial images around the world. In *International Computer Vision Conference*, pages 1689–1697, 2015. 2
- [29] G. Mátyus, W. Luo, and R. Urtasun. Deeproadmapper: Extracting road topology from aerial images. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2, 6, 8
- [30] H. Mayer, S. Hinz, U. Bacher, and E. Baltsavias. A test of automatic road extraction approaches. In *IAPRS*, volume 36(3), pages 209 – 214, 2006. 2

- [31] V. Mnih and G. E. Hinton. Learning to detect roads in high-resolution aerial images. In *European Conference on Computer Vision*, 2010. 2
- [32] V. Mnih and G. E. Hinton. Learning to label aerial images from noisy data. In *International Conference on Machine Learning*, 2012. 2
- [33] S. P. Mohanty. Crowdai dataset. https://www.crowdai.org/challenges/mapping-challenge/dataset_files, 2018. 5, 7
- [34] S. P. Mohanty. Crowdai mapping challenge 2018 : Baseline with mask rcnn. <https://github.com/crowdai/crowdai-mapping-challenge-mask-rcnn>, 2018. 2, 7
- [35] J. Montoya, J. Wegner, L. Ladický, and K. Schindler. 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. 2
- [36] S. Paisitkriangkrai, J. Sherrah, P. Janney, and A. van den Hengel. Effective semantic pixel labelling with convolutional networks and conditional random fields. In *CVPRws*, 2015. 2
- [37] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *European Conference on Computer Vision*, pages 75–91. Springer, 2016. 4
- [38] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 3
- [39] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *Trans. Neur. Netw.*, 20(1):61–80, Jan. 2009. 2
- [40] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. 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. 1, 2
- [41] G. Sohn and I. 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. 2
- [42] R. Stoica, X. Descombes, and J. Zerubia. A Gibbs Point Process for road extraction from remotely sensed images. *IJCV*, 57(2):121 – 136, 2004. 2
- [43] P. Tokarczyk, J. Wegner, S. Walk, and K. 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. 1
- [44] A. van Etten, D. Lindenbaum, and T. Bacastow. Spacenet: A remote sensing dataset and challenge series. *arXiv*, arXiv:1807.01232v2:1–21, 2018. 2, 5
- [45] M. Volpi and V. 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. 1
- [46] M. Volpi and D. Tuia. Dense semantic labeling of sub-decimeter resolution images with convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 55(2):881–893, 2017. 1, 2
- [47] J. Wegner, J. Montoya, and K. Schindler. A higher-order crf model for road network extraction. In *CVPR*, pages 1698–1705, 2013. 2, 5, 6
- [48] J. Wegner, J. Montoya, and K. Schindler. Road networks as collections of minimum cost paths. *ISPRS Journal of Photogrammetry and Remote Sensing*, 108:128 – 137, 2015. 2