PolyWorld: Polygonal Building Extraction with Graph Neural Networks in Satellite Images

Stefano Zorzi [1 3]
stefano.zorzi@icg.tugraz.at

Shabab Bazrafkan [2]
sbazrafkan@blackshark.ai

Stefan Habenschuss [2]
shabenschuss@blackshark.ai

Friedrich Fraundorfer [1]
fraundorfer@icg.tugraz.at

[1] Graz University of Technology
[2] Blackshark.ai [3] VRVis

Abstract

While most state-of-the-art instance segmentation methods produce binary segmentation masks, geographic and cartographic applications typically require precise vector polygons of extracted objects instead of rasterized output. This paper introduces PolyWorld, a neural network that directly extracts building vertices from an image and connects them correctly to create precise polygons. The model predicts the connection strength between each pair of vertices using a graph neural network and estimates the assignments by solving a differentiable optimal transport problem. Moreover, the vertex positions are optimized by minimizing a combined segmentation and polygonal angle difference loss. PolyWorld significantly outperforms the state of the art in building polygonization and achieves not only notable quantitative results, but also produces visually pleasing building polygons. Code and trained weights are publicly available at https://github.com/zorzi-s/PolyWorldPretrainedNetwork.

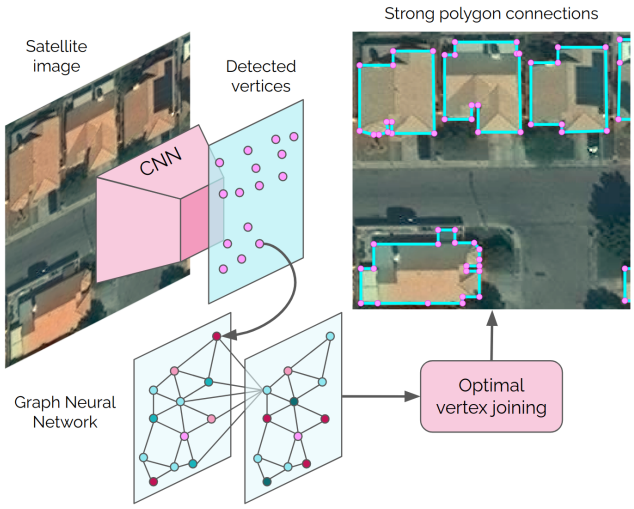

Figure 1. **Polygonal object extraction with PolyWorld**. The method uses a CNN backbone to detect vertex candidates from an image, and aggregates the information of the visual descriptors exploiting a graph neural network. The connections between vertices are generated solving a differentiable optimal transport problem.

1. Introduction

The extraction of vector representations of building polygons from aerial and satellite imagery has been growing in importance in many remote sensing applications, such as cartography, city modelling and reconstruction, as well as map generation. Most building extraction and polygonization methods rely on the vectorization of probability maps produced by a segmentation network. These approaches are not end-to-end learned, which means that imperfections and artifacts produced by the segmentation model are carried through the entire pipeline with the consequent generation of unregular polygons.

In this paper, we present a new way of tackling the building polygonization problem. Rather than learning a segmentation network which is then followed by a polygonization method, we propose a novel neural network architecture called PolyWorld that detects building corners from a satellite image and uses a learned matching procedure to connect them in order to form polygons. Thereby, our method allows the generation of valid polygons in an end-to-end fashion.

PolyWorld extracts positions and visual descriptors of building corners using a Convolutional Neural Network (CNN) and generates polygons by evaluating whether the connections between vertices are valid. This procedure finds the best connection assignment between the detected vertex descriptors, which means that every corner must be matched with the subsequent vertex of the polygon. The connections between polygon vertices can be represented as the solution of a linear sum assignment problem. In PolyWorld, an important role is played by a Graph Neural Net-

work (GNN) that propagates global information through all the vertex embeddings, increasing the descriptors' distinctiveness. Moreover, it refines the position of the detected corners in order to minimize the combined segmentation and polygonal angle difference loss. PolyWorld demonstrates superior performance compared to the state of the art building extraction and polygonization methods, not only achieving higher segmentation and detection results, but also producing more regular and clean building polygons.

2. Related work

Since building detection and segmentation from satellite images has been of major research interest throughout the last few decades, discussing all work is beyond the scope of this paper. In this section we therefore focus on the most relevant contributions in different related categories.

Building segmentation: Before the great success of deep learning methods, building footprint delineation was mainly done with multi-step, bottom-up approaches by combining multi-spectral overhead images and airborne LIDAR data [3, 31]. Nowadays, deep learning-based methods are state-of-the-art, mainly addressing the problem by refining raster footprints via heuristic polygonization approaches computed by powerful semantic or instance segmentation networks [11–15, 21]. The majority of these segmentation models are trained with cross entropy, soft intersection over union, or Focal based losses [4, 18, 28, 34], achieving high scores in terms of intersection over union, recall, and precision, but mostly generating irregular building outlines that are neither visually pleasing, nor employable in most cartographic applications. A typical problem of semantic and instance segmentation networks is, in fact, the inability of outlining straight building walls and sharp corners in presence of ground truth noise, e.g. misalignment between a segmentation mask and an intensity image. Some publications, therefore, suggest to post-process the segmented building footprints in order to align the segmentation outlines to the actual building contours described in the intensity image. DSAC [24] employs an Active Contour Model to integrate geometrical priors and constraints in the segmentation process, while DARNet [7] proposes a loss function that encourages the contours to match the building boundaries. Another technique to make the building contours more regular and realistic is to combine adversarial and regularized losses [35, 36, 40].

Polygon prediction: Standard semantic and instance segmentation networks are easy to train and generate accurate segmentation masks, but most remote sensing applications that involve building layers require segmentation data in vector format rather than rasterized masks. Object detection and polygonization methods found in literature can be classified into two categories.

The first category includes methods that perform the vectorization of grid-like information, e.g. the probability map produced by a segmentation network. In [38] the authors corrected the segmentation masks produced with Mask R-CNN [13] by first simplifying the detected boundaries using the Douglas-Peucker algorithm [9] and subsequently refining the resulting polygons using a Minimum Descriptor Length method [32]. More recently, Chen et al. [6] suggested to regularize the segmentation produced with a CNN via quantizing the histogram of building boundaries in angle space, which can be achieved by exploiting a Relative Angle Gradient Transform. Zorzi et al. [39] applied three different networks in series to perform the extraction and polygonization. Their method uses a CNN to generate building segmentation; then, it performs a regularization on the raster data by applying an autoencoder trained with regularized [35, 36] and adversarial losses, and finally detects building corners using a third CNN. The polygonization is performed by ordering the detected corners following the regularized boundaries. All these methods are developed with the idea of decomposing the building extraction and polygonization problem into smaller tasks that can be tackled individually. As a result, most of these approaches are computationally heavy, they lack of parallelization and their hyperparameters must be carefully tuned in order to achieve the desired results. Most importantly, since they are composed of a sequence of blocks, these methods can accumulate errors through their pipeline, which can harm the quality of the final polygonization. The current state of the art in the field is achieved by the Frame Field Learning (FFL) method [10], which generates a vector field that encodes useful boundary information alongside the corresponding segmentation mask. Moreover, the contour is optimized to be aligned to the frame field using an Active Skeleton Model.

The second category is represented by methods that directly learn a vector representation. PolyTransform [17] initializes a polygon for every object instance and refines the vertex positions using a Transformer network [37]. Curve GCN [20] learns a graph convolutional network to deform polygons in an iterative manner. Some networks also utilize recurrent neural networks (RNN) to extract polygons vertex by vertex, e.g. Polygon-RNN [5] and Polygon-RNN++ [1]. Also PolyMapper [16] applies a RNN to predict building and road vertices one by one. All these methods directly process polygon parameters but they are typically more difficult to train and they need multiple iterations during inference. Moreover they have troubles dealing with complex building shapes, e.g. structures having curved walls or holes in their shape. PolyWorld, which is presented in this paper, fits well into the second category of direct polygon prediction, although the employed architecture and general idea fundamentally differs from all existing work.

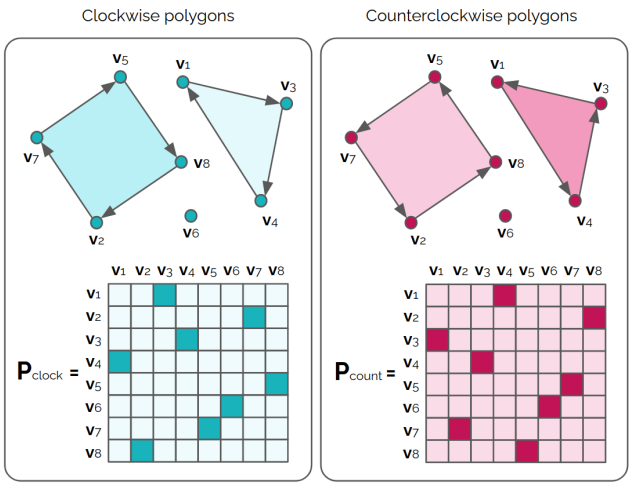

Figure 2. In PolyWorld, the connections between polygon vertices are described with a permutation matrix. The i-th row of the permutation matrix P_{clock} or P_{count} indicates the index of the next clockwise or counterclockwise vertex connected to v_i. Please note that the permutation matrix of the clockwise oriented polygons P_{clock} is the transpose of the permutation matrix of the counterclockwise oriented polygons P_{count}.

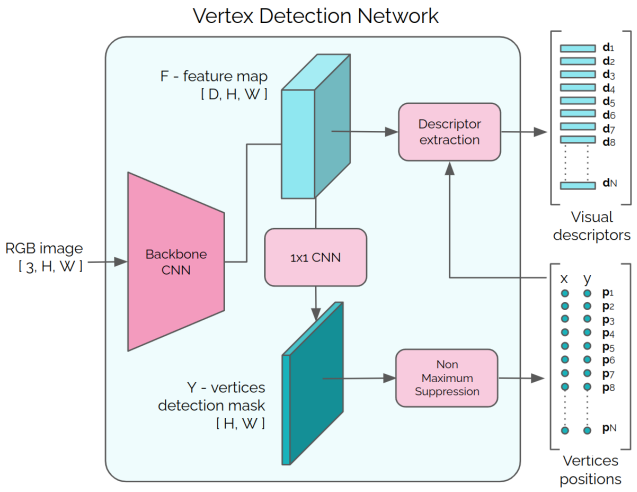

Figure 3. The Vertex Detection Network of PolyWorld. A backbone CNN receives the intensity image and returns a feature map and a vertex detection mask. A Non Maximum Suppression (NMS) algorithm removes undesired vertices and returns N locations that correspond to the highest peaks in the detection mask. The visual descriptors are then extracted from the feature map at every location provided by the NMS.

3. The PolyWorld Architecture

The main idea behind PolyWorld is to represent building polygons in the scene as a set of vertices connected according to a permutation matrix, as illustrated in Figure 2. Each corner of the polygon is associated to a specific row of the permutation matrix that indicates the next clockwise vertex. The permutation matrix must fulfill certain polygonal constraints: ① every vertex corresponds to at most one clockwise connection and one counterclockwise connection; ② the permutation matrix of the clockwise oriented polygons is the transpose of the counterclockwise permutation matrix; ③ a vertex having its entry in the diagonal of the permutation matrix can be discarded since, in reality, there are no building polygons having a single corner, e.g. vertex v_6 in Fig. 2.

PolyWorld is composed of three blocks: a Vertex Detection Network that extracts a set of possible building corner candidates, an Attentional Graph Neural Network that aggregates information through the vertices and refines their position, and an Optimal Connection Network that generates the connections between vertices. Given the input image, the model provides the position of the detected building corners and a valid permutation matrix.

3.1. Vertex Detection Network

The vertex detection network is depicted in Figure 3. The module receives an image $I \in \mathbb{R}^{3\times H\times W}$ as input, it forward propagates I through a fully convolutional backbone, and returns a D-dimensional feature map $F \in \mathbb{R}^{D\times H\times W}$. The vertex detection mask $Y \in \mathbb{R}^{H\times W}$ is obtained by propagating the features F through a 1×1 convolutional layer. The detection mask Y is then filtered using a Non Maximum Suppression algorithm with kernel size of 3, in order to retain the most relevant peaks. The *positions* p of the N highest peaks are then used to extract N *visual descriptors* $d \in \mathbb{R}^D$ from the feature map F. Vertex positions consist of x and y image coordinates $p_i := (x, y)_i$. During training, the backbone not only learns to produce a feature map F useful to segment building corners but also learns to embed an abstract representation of the latter. During training, this information is constrained to represent the building vertex by matching with the other detected corners.

3.2. Attentional Graph Neural Network

Besides the position and the visual appearance of a building corner, considering other contextual information is essential to describe it in a more rich and distinctive way. Capturing relationships between its position and appearance with other vertices in the image can be helpful to link it with corners having the same roof style, having a compatible shape and pose for the matching, or simply with adjacent corners. Motivated by this consideration, we design the next PolyWorld block using an attentional graph neural network that computes a set of *matching descriptors* $m_i \in \mathbb{R}^D$ by learning short and long term vertex relationships given the vertex positions p and the visual descriptors v extracted by the vertex detection network. Moreover, this block also estimates a *positional offset* $t_i \in \mathbb{R}^2$ in order to refine the

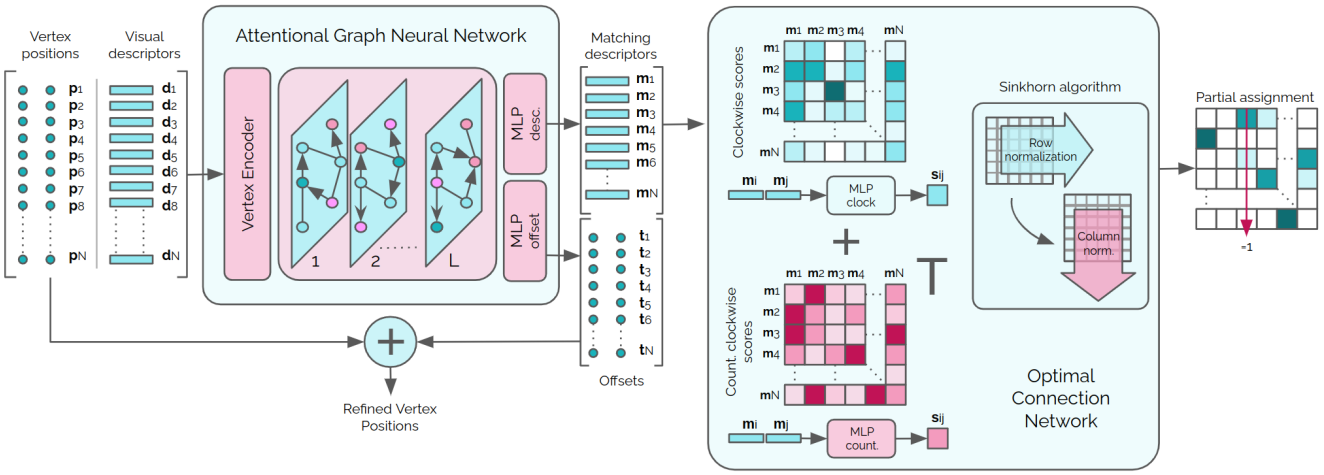

Figure 4. The Attentional Graph Neural Network and the Optimal Connection Network of PolyWorld. The first module uses a vertex encoder to map *vertex positions* p and *visual descriptors* d into a single vector, and uses L self-attention layers to increase their distinctiveness. The module returns a set of *offsets* t and the *matching descriptors* m. The offsets are used to refine the vertex positions, while m are propagated through the optimal connection network that creates a $N \times N$ score matrix and generates the permutation matrix using the Sinkhorn algorithm.

vertex positions, optimizing the corner angle and the footprint segmentation. As we will show in the following chapters, aggregating features from all the detected vertices and refining the vertex positions leads not only to improved segmentation scores, but also to more realistic building polygons.

3.2.1 Vertex Encoder

Before forward propagating through the graph neural network, positions p and visual descriptors d are merged by a Multilayer Perceptron (MLP).

$$d'_i = \text{MLP}_{enc}\left([d_i||p_i]\right) \tag{1}$$

MLP_{enc} receives the concatenation $[\cdot||\cdot]$ of p_i and d_i and returns a new descriptor $d'_i \in \mathbb{R}^D$ that encodes positional and visual information together.

3.2.2 Self Attention Network

The aggregation is performed by a self-attention mechanism [37] that propagates the information across vertices, increasing their contextual information.

Given the intermediate descriptors $x \in \mathbb{R}^{D\times N}$, the model employs a linear projection to produce a query $Q(x)$, a key $K(x)$, and a value $V(x)$. The weights between the nodes are computed taking the softmax over the dot product $Q(x)K(x)^\top$. The result is then multiplied with the values $V(x)$ in order to propagate the information across all the vertices. The attention mechanism can be written as:

$$A = \text{softmax}\left(\frac{Q(x)\cdot K(x)^\top}{\sqrt{n_k}}\right)V(x) \tag{2}$$

where the normalization term n_k is the dimension of the queries and keys.

This operation is repeated for a fixed numbers of layers L. The message $A^{(l)} \in \mathbb{R}^{D\times N}$ is the attention result at layer l and it is used to update the vertex descriptors at every step. We indicate $a_i^{(l)}$ the i-th column of $A^{(l)}$, that represents the attention message relative to the i-th vertex of the graph. In every layer the vertex descriptors are updated as follows:

$$x_i^{(l+1)} = \text{MLP}^{(l)}\left(\left[x_i^{(l)}||a_i^{(l)}\right]\right) \tag{3}$$

The embeddings received by the the first attention layer are the descriptors produced by the vertex encoder $d' = x^{(l=1)}$. Finally, the embedding of the i-th vertex produced by the last attention layer $x_i^{(L)}$ is decomposed in two components: a *matching descriptor* $m_i \in \mathbb{R}^D$ and a *positional offset* $t_i \in \mathbb{R}^2$.

$$m_i = \text{MLP}_{match}\left(x_i^{(L)}\right) \tag{4}$$

$$t_i = \text{MLP}_{offset}\left(x_i^{(L)}\right) \tag{5}$$

The matching descriptors are used further to generate a valid combination of connections between the vertices, while the offsets are combined with the vertex positions as follows:

$$\hat{p}_i = p_i + \gamma \cdot t_i \tag{6}$$

where γ is a factor that regulates the correction radius since the offsets are generated through a HardTanh activation function and the values range between -1 and 1.

3.3. Optimal Connection Network

The last block of PolyWorld is the optimal connection network that connects the vertices generating a permutation matrix $P \in \mathbb{R}^{N \times N}$. The assignment can be obtained by calculating a score matrix $S \in \mathbb{R}^{N \times N}$ for all possible vertex pairs and maximizing the overall score $\sum_{i,j} P_{i,j} S_{i,j}$.

Given two matching descriptors m_i and m_j encoding the information of two distinct vertices, we exploit MLP_{clock} to detect whether the clockwise connection $m_i \rightarrow m_j$ is possible. The network receives the concatenation of the two descriptors and returns a high score value if the connection between them is strong; e.g. if m_i represents the top-left corner of an orange roof, it is likely that m_j is the next clockwise vertex if it represents a top-right corner of an orange roof.

$$s^{clock}_{i \rightarrow j} = \text{MLP}_{clock}\left([m_i || m_j]\right) \tag{7}$$

Vice versa we estimate how strong is the counterclockwise connection $m_i \rightarrow m_j$ exploiting a second network MLP_{count}.

$$s^{count}_{i \rightarrow j} = \text{MLP}_{count}\left([m_i || m_j]\right) \tag{8}$$

By enforcing constraint ②, we can establish a consistency check between the clockwise and the counterclockwise path of vertices. The final score matrix S is calculated as the combination of the clockwise score matrix S_{clock} and the transpose version of the counterclockwise score matrix S_{count}:

$$S = S_{clock} + S^{\top}_{count} \tag{9}$$

The double path consistency ensures to have stronger matches, better connections and, ultimately, higher polygon quality.

As a final step, we use the Sinkhorn algorithm [8, 27, 29, 30] to find the optimal assignment matrix P given the score matrix S. The Sinkhorn is a GPU efficient and differentiable version of the Hungarian algorithm [26], used to solve linear sum assignment problems, and it consists of normalizing rows and columns of $\exp(S)$ for a certain amount of iterations.

4. Losses

Detection: We train the corner detection as a segmentation task using weighted binary cross-entropy loss:

$$\begin{aligned} \mathcal{L}_{det} = &- \omega \cdot \sum_{i=1}^{H} \sum_{j=1}^{W} \bar{Y}_{i,j} \cdot \log\left(Y_{i,j}\right) \\ &- \sum_{i=1}^{H} \sum_{j=1}^{W} (1 - \bar{Y}_{i,j}) \cdot \log\left(1 - Y_{i,j}\right) \end{aligned} \tag{10}$$

The ground truth $\bar{Y}$ is a sparse array of zeros. Pixels indicating the presence of a building corner have a value of one. Since the segmentation is heavily unbalanced for the foreground pixels, we use the factor ω to counterbalance positive samples.

Matching: The attentional graph neural network and the optimal connection network of PolyWorld are fully differentiable which allows us to backpropagate from the generated partial assignment to the backbone that generates the visual descriptors. This path is trained in a supervised manner from the ground truth permutation matrix $\bar{P}$ using cross entropy loss:

$$\mathcal{L}_{match} = - \sum_{i=1}^{N} \sum_{j=1}^{N} \bar{P}_{i,j} \cdot \log\left(P_{i,j}\right) \tag{11}$$

Due to the iterative normalization through rows and columns made by the Sinkhorn algorithm, minimizing the negative log-likelihood of the positive matches of P leads to simultaneously maximizing the precision and the recall of the matching.

Positional refinement: Due to low image resolution, ground truth misalignments, or wrong building labelling, the position of the vertices provided by the vertex detection network is not optimal in practice. The subsequent matching procedure, therefore, could produce polygons having corner angles different from the ground truth, altering the visual appeal of the extracted polygons. In order to repress this phenomenon, we minimize the difference between the corner angles of the predicted polygons and the ground truth polygons.

We indicate with $\mathcal{C}$ the function that converts a permutation matrix and vertex positions to a list of polygons $\mathcal{P}$. The polygons predicted by PolyWorld and the ground truth polygons are then $\mathcal{P} = \mathcal{C}\left(\hat{p}, P\right)$ and $\bar{\mathcal{P}} = \mathcal{C}\left(\bar{p}, \bar{P}\right)$, respectively. Indicating with $\mathcal{P}_k$ the k-th polygon instance extracted from the image and composed of a set of clockwise ordered vertex positions, we formulate the angle loss as:

$$\begin{aligned} \mathcal{L}_{angle} &= \sum_{k=1}^{K} \sum_{(u \rightarrow v \rightarrow w)} 1 - \exp\left(-\sigma \cdot \left|\Delta_{k,(u,v,w)}\right|\right) \\ \Delta_{k,(u,v,w)} &= \angle\left(\hat{p}_u, \hat{p}_v, \hat{p}_w\right)_k - \angle\left(\bar{p}_u, \bar{p}_v, \bar{p}_w\right)_k \end{aligned} \tag{12}$$

where $(u \twoheadrightarrow v \twoheadrightarrow w)$ denotes the indices of any three consecutive vertices in polygon $\mathcal{P}_k$ and $\bar{\mathcal{P}}_k$. The strength of the loss term is regulated by the factor σ, while $\angle(\hat{p}_u, \hat{p}_v, \hat{p}_w)_k$ and $\angle(\bar{p}_u, \bar{p}_v, \bar{p}_w)_k$ indicate the angle at the v-th vertex of the polygon $\mathcal{P}_k$ and $\bar{\mathcal{P}}_k$, respectively.

Even if the network is encouraged to fix corner angles, $\mathcal{L}_{angle}$ potentially induces unexpected modifications of the polygon shapes since it leaves some degrees of freedom to the network on how to warp the vertices. In our experiments the network stretched the polygons in undesired ways while respecting the angle criterion, potentially producing misaligned footprints. PolyWorld fixes this issue by minimizing a segmentation loss between the ground truth and predicted polygons. This refinement loss not only inhibits unwanted effects of $\mathcal{L}_{angle}$, but it also increases segmentation scores as documented in the next sections.

We generate the footprint mask of the predicted polygons exploiting a Differentiable Polygon Rendering method [33]. It is the soft version of the winding number algorithm, that checks whether a pixel location x is inside the polygon $\mathcal{P}_k$ with the equation:

$$W(x, \mathcal{P}_k) = \sum_{(u \twoheadrightarrow v)} \frac{\lambda \cdot \det(\overline{\hat{p}_u x}, \overline{\hat{p}_v x})_k}{1 + \left|\lambda \cdot \det(\overline{\hat{p}_u x}, \overline{\hat{p}_v x})_k\right|} \cdot \angle(\hat{p}_u, x, \hat{p}_v)_k \tag{13}$$

where $(u \twoheadrightarrow v)$ are the indices of any two consecutive vertices of $\mathcal{P}_k$, $\det(\cdot)$ is the determinant of vectors $\overline{\hat{p}_u q}$ and $\overline{\hat{p}_v q}$, and the value λ fixes the smoothness of the raster contours.

Calculating the winding number for every pixel location in the image, we generate the raster mask $M_k \in \mathbb{R}^{H \times W}$ of the polygon $\mathcal{P}_k$. The segmentation loss $\mathcal{L}_{seg}$ is finally calculated as the soft intersection over union [28] between the ground truth segmentation mask $\bar{M}$ and the combination of extracted polygon masks:

$$\mathcal{L}_{seg} = \text{softIoU}\left(\sum_{k=1}^{K} M_k, \bar{M}\right) \tag{14}$$

Since the NMS block is not differentiable, the only way for the network to minimize $\mathcal{L}_{seg}$ and $\mathcal{L}_{angle}$ is to generate a proper set of offsets t for Equation 6.

5. Implementation details

Training and inference: The NMS algorithm extracts a list of $N = 256$ vertex positions p with the highest detection confidence. During training, these positions are not directly used to extract the descriptors d from the features F, but they are first sorted to match the nearest neighboring ground truth point. After sorting, p_i is the closest vertex to the ground truth point $\bar{p}_i$. This procedure ensures to have index consistency between the positions p and the ground truth permutation matrix $\bar{P}$. In reality, the number of extracted points N is always greater than the number of building corners in the image, therefore the vertices that do not minimize the distance with any of the ground truth points have their entry assigned to the diagonal of $\bar{P}$. PolyWorld is trained from scratch linearly combining detection, matching and refinement losses: $\mathcal{L}_{det} + \mathcal{L}_{match} + \mathcal{L}_{angle} + \mathcal{L}_{seg}$. Rather than learning the matching branch at the early training stage, we prefer to first pretrain the vertex detection network only using $\mathcal{L}_{det}$. When it extracts sufficiently accurate building corners, we keep training the full PolyWorld architecture with the complete loss. During inference, vertices that have their entry in the diagonal of the permutation matrix are discarded (constraint ③).

Architecture: As backbone PolyWorld uses a Residual U-Net model [2]. The descriptor dimension and the intermediate representations of the attentional graph neural network have the same size $D = 64$. We use $L = 4$ self attention layers having 4 parallel heads each. In Equation 6, we use $\gamma = 0.05$, allowing a maximum offset radius of 5% of the image size. Increasing γ further does not improve the results. We use $\omega = 100$ in Equation 10, while, in Equation 13, the value of λ is set to 10^3 as suggested in [33]. During training, the permutation matrix P is calculated by performing $T = 100$ Sinkhorn iterations, whereas during inference the exact linear sum assignment result is determined using the Hungarian algorithm on the CPU. With this configuration a forward pass takes on average 24 ms per image (320×320 pixels) on a NVIDIA GTX 3090 and an AMD Ryzen7 3700X.

6. Experiments

Dataset: Building extraction and polygonization networks require ground truth polygonal annotations in order to be trained. Therefore, we perform all our experiments using the CrowdAI Mapping Challenge dataset [25], which is composed of over 280k satellite images of size 300×300 pixels for training and 60k images for testing. In order to avoid pooling issues in the backbone, we upsample the images to 320×320 pixels. The dataset provides the polygon annotations in MS COCO format [19].

Evaluation metrics: We evaluate and compare the results of PolyWorld computing classical segmentation and detection metrics, such as Intersection over Union (IoU), and MS COCO [19] Average Precision (AP) and Average Recall (AR). In order to evaluate the regularity of the extracted building contours, we also calculate the Max Tangent Angle Error [10]. This metric compares the tangent angles of the predicted and ground truth polygons, penalizing building contours not aligned with the ground truth.

In general, simple polygonization methods applied to the raster output of classical segmentation networks produce

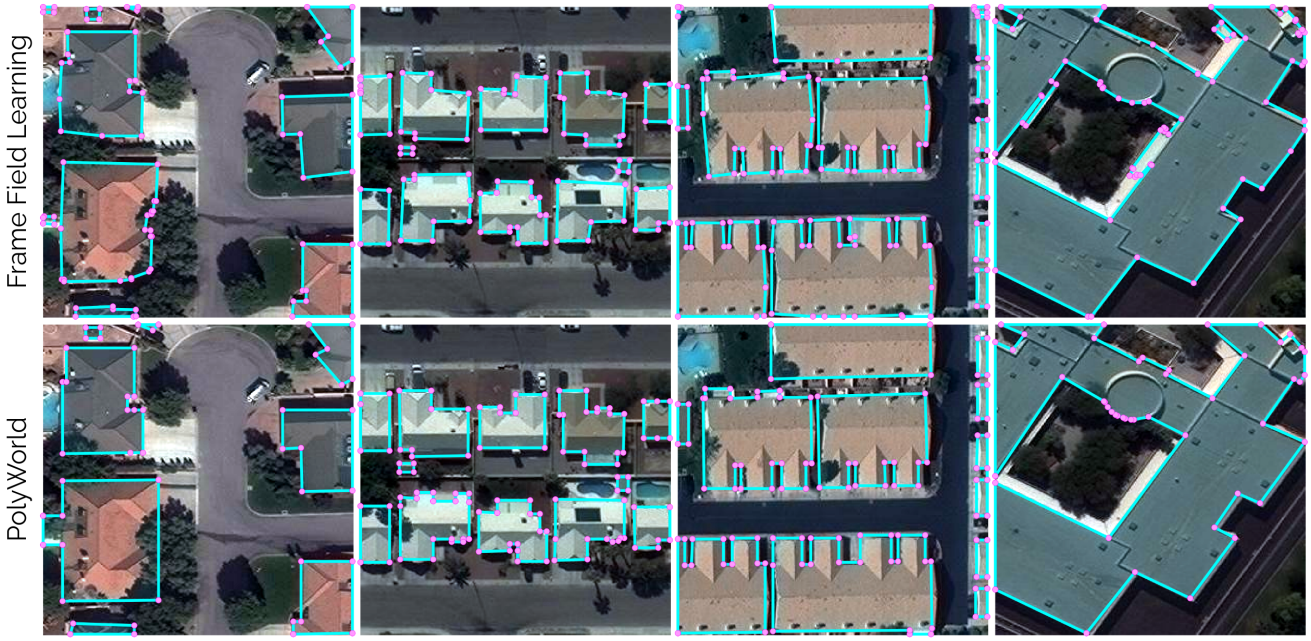

Figure 5. Examples of building extraction and polygonization on CrowdAI test dataset. Top row: Frame Field Learning [10] approach with Res101-UNet as backbone and ACM polygonization. Bottom row: PolyWorld results.

unregular polygons with a high amount of redundant vertices. On the other hand, building extraction and polygonization methods tend to reduce the segmentation scores in favour of more regular and realistic footprints. Since the goal of the proposed method is to generate high quality building polygons ready to be used on geographical applications, we introduce the *complexity aware IoU (C-IoU)* metric computed as follows:

$$\text{C-IoU}(A, \bar{A}) = \text{IoU}(A, \bar{A}) \cdot (1 - \text{RD}(N_A, N_{\bar{A}})) \quad (15)$$

where the first term $\text{IoU}(A, \bar{A})$ indicates the intersection over union between the predicted polygon raster mask A and the ground truth segmentation $\bar{A}$. The second term $\text{RD}(N_A, N_{\bar{A}}) = |N_A - N_{\bar{A}}|/(N_A + N_{\bar{A}})$ is the relative difference between the number of extracted vertices N_A in the image used to produce the raster A, and the number of ground truth vertices $N_{\bar{A}}$. The metric aims to favor polygonizations with a complexity similar to ground truth, penalizing both oversimplified building shapes and polygons with redundant vertices. Ideally, a method achieves a high C-IoU score if it manages to balance the trade off between segmentation accuracy and polygonization complexity.

Results: Qualitative results of experiments conducted on the CrowdAI [25] dataset can be observed in Figure 5. The images represent different kind of urban areas and they are sorted by building complexity from left to right. We compare the results of PolyWorld with the Frame Field Learning (FFL) method [10] that represents the state of the art on building extraction and polygonization. Both FFL and PolyWorld generalize well in every kind of building, but PolyWorld produces overall cleaner and more linear geometries without developing undesired artifacts. It is interesting to note that PolyWorld can better deal with hard object occlusions, estimating the position of the hidden corners and connecting them producing more regular and realistic footprints, as shown on the left image. The robustness of the vertex detection and matching process is shown on the right images, where PolyWorld does not have issues in generating polygons of complex buildings with curved walls or inner courtyards. More images can be found in the supplementary material.

In Table 1 we report the MS COCO metrics results using the test-set of CrowdAI. We computed the scores of PolyWorld considering and discarding the positional offsets used to correct the vertex positions ("offset on" and "offset off"). Our approach is compared with FFL, PolyMapper [16], and two general-purpose instance segmentation networks: Mask R-CNN [13] and PANet [21]. For the FFL method, we report the results of the model trained with and without frame field output, and with different polygonization approaches: "mask" is raster segmentation, "simple poly" refers to the marching squares [22] contour detector followed by the Douglas–Peucker [9] simplification, and "ACM poly" refers to the Active Contour Model [10] polygonization. The results of PolyWorld show state-of-the-art precision and recall performances despite the fact that the refinement offsets have been ignored. When the vertex po-

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 [13]	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 [21]	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 [16]	55.7	86.0	65.1	30.7	68.5	58.4	62.1	88.6	71.4	39.4	75.6	75.4
FFL (no field), mask	57.8	84.0	66.9	33.8	74.1	80.7	67.0	90.4	76.9	46.2	79.7	85.7
FFL (no field), simple poly	61.1	87.4	71.2	35.1	74.5	82.3	64.7	89.4	74.1	41.7	77.9	85.7
FFL (with field), mask	57.7	83.8	66.3	33.8	73.8	81.0	68.1	91.0	77.7	47.5	80.0	86.7
FFL (with field), simple poly	61.7	87.6	**71.4**	35.7	74.9	83.0	65.4	89.8	74.6	42.5	78.6	85.8
FFL (with field), ACM poly [10]	61.3	87.4	70.6	33.9	75.1	83.1	64.9	89.4	73.9	41.2	78.7	85.9
PolyWorld (offset off)	58.7	86.9	64.5	31.8	80.1	85.9	71.7	92.6	79.9	47.4	85.7	94.0
PolyWorld (offset on)	**63.3**	**88.6**	70.5	**37.2**	**83.6**	**87.7**	**75.4**	**93.5**	**83.1**	**52.5**	**88.7**	**95.2**

Table 1. MS COCO [19] results on the CrowdAI test dataset [25] for all the building extraction and polygonization experiments. The results of PolyWorld are calculated discarding the correction offsets (offset off), and refining the vertex positions (offset on). FFL refers to the Frame Field Learning [10] method. The results are computed with and without frame field estimation. "mask" refers to the pure segmentation produced by the model. "simple poly" refers to the Douglas–Peucker polygon simplification [9], and "ACM poly" refers to the Active Contour Model [10] polygonization method.

Method	IoU	C-IoU	MTA	N ratio
FFL (no field), simple poly	83.9	23.6	51.8°	5.96
FFL (with field), simple poly	84.0	30.1	48.2°	2.31
FFL (with field), ACM poly	84.1	73.7	33.5°	1.13
PolyWorld (offset off)	89.9	86.9	35.0°	0.93
PolyWorld (offset on)	**91.3**	**88.2**	**32.9°**	0.93

Table 2. *Intersection over union (IoU)*, *mean tangent angle error (MTA)*, and *complexity aware IoU (C-IoU)* results on the test-set of the CrowdAI dataset [25]. The last column reports the ratio between the number of detected vertices and the number of ground truth vertices.

sition refinement is enabled, all the scores improve by a considerable margin, demonstrating the effectiveness of the refinement losses. Another interesting fact to mention is that PolyWorld uses considerably fewer points to describe the buildings compared to the FFL approach. In the 60k test images of CrowdAI dataset, the ground truth counts a total of about 4.4M vertices. PolyWorld extracts 4.2M polygon vertices in the test-set, compared to the 5.1M extracted by FFL with ACM polygonization. Nevertheless, our approach is able to achieve better segmentation scores, suggesting that the PolyWorld vertex extraction is more efficient.

In Table 2 we report the intersection over union, mean tangent angle error, and complexity aware IoU results. Again, there is a noticeable improvement in all the metrics exploiting the vertex position refinement. Even though the ACM polygonization of FFL significantly outperforms the Douglas–Peucker polygonization in terms of MTA and C-IoU, the full PolyWorld method manages to overtake all the FFL results.

7. Limitations and future work

In our future work we want to demonstrate the capability of PolyWorld to generalize and produce accurate polygons on large scale data sets with a number of unseen conditions. This will include the Inria segmentation dataset [23] with Open Street Map annotations since it contains varied areas captured from different cities around the globe, and includes adjacent buildings with common corners. From a technical point of view, the case of common corners could be efficiently solved using PolyWorld by generalizing the vertex detection network to multiclass segmentation, detecting the number of vertices located in the same position, and sampling the visual descriptor multiple times from the feature map if a shared corner is detected. Another limitation of PolyWorld concerns buildings with holes. Since the permutation matrix does not carry the information to bind outer and inner rings to the same shape, a post processing step might be required to generate multi-polygons.

8. Conclusion

We presented PolyWorld, a novel method capable of elegantly extracting building polygons from satellite and aerial images in an end-to-end manner. The evaluation results experimentally prove the power and effectiveness of self-attention graph neural networks for matching and positional refinement of detected building vertices. By solving an optimal transport problem, our method provides strong and reliable vertex connections and implicitly avoids redundant points. Our experiments show that PolyWorld significantly outperforms existing building extraction approaches, enabling highly accurate and regular building footprints, which fulfill the strict requirements of geographic and cartographic applications.

Acknowledgments

Thanks to VRVis for financing the project. VRVis is funded by BMK, BMDW, Styria, SFG, Tyrol and Vienna Business Agency in the scope of COMET - Competence Centers for Excellent Technologies (879730) which is managed by FFG.

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

[2] Md Zahangir Alom, Chris Yakopcic, Mahmudul Hasan, Tarek M Taha, and Vijayan K Asari. Recurrent residual u-net for medical image segmentation. *Journal of Medical Imaging*, 6(1):014006, 2019. 6

[3] 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. 2

[4] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4413–4421, 2018. 2

[5] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5230–5238, 2017. 2

[6] Yuhao Chen, Yifan Wu, Linlin Xu, and Alexander Wong. Quantization in relative gradient angle domain for building polygon estimation. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 8360–8367. IEEE, 2021. 2

[7] Dominic Cheng, Renjie Liao, Sanja Fidler, and Raquel Urtasun. Darnet: Deep active ray network for building segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7431–7439, 2019. 2

[8] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013. 5

[9] David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: the international journal for geographic information and geovisualization*, 10(2):112–122, 1973. 2, 7, 8

[10] Nicolas Girard, Dmitriy Smirnov, Justin Solomon, and Yuliya Tarabalka. Polygonal building extraction by frame field learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5891–5900, 2021. 2, 6, 7, 8

[11] Sergey Golovanov, Rauf Kurbanov, Aleksey Artamonov, Alex Davydow, and Sergey Nikolenko. Building detection from satellite imagery using a composite loss function. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 229–232, 2018. 2

[12] Ryuhei Hamaguchi and Shuhei Hikosaka. Building detection from satellite imagery using ensemble of size-specific detectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 187–191, 2018. 2

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2, 7, 8

[14] Vladimir Iglovikov, Selim Seferbekov, Alexander Buslaev, and Alexey Shvets. Ternausnetv2: Fully convolutional network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 233–237, 2018. 2

[15] Vladimir Iglovikov and Alexey Shvets. Ternausnet: U-net with vgg11 encoder pre-trained on imagenet for image segmentation. *arXiv preprint arXiv:1801.05746*, 2018. 2

[16] Zuoyue Li, Jan Dirk Wegner, and Aurélien Lucchi. Topological map extraction from overhead images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1715–1724, 2019. 2, 7, 8

[17] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. Polytransform: Deep polygon transformer for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9131–9140, 2020. 2

[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 2

[19] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 6, 8

[20] Huan Ling, Jun Gao, Amlan Kar, Wenzheng Chen, and Sanja Fidler. Fast interactive object annotation with curve-gcn. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5257–5266, 2019. 2

[21] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018. 2, 7, 8

[22] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 7

[23] Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2017. 8

[24] Diego Marcos, Devis Tuia, Benjamin Kellenberger, Lisa Zhang, Min Bai, Renjie Liao, and Raquel Urtasun. Learning deep structured active contours end-to-end. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8877–8885, 2018. 2

[25] S. P. Mohanty. *CrowdAI mapping challenge 2018 dataset*, 2019 (accessed November 10, 2019). https://www.crowdai.org/challenges/mapping-challenge. 6, 7, 8

[26] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957. 5

[27] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019. 5

[28] Md Atiqur Rahman and Yang Wang. Optimizing intersection-over-union in deep neural networks for image segmentation. In *International symposium on visual computing*, pages 234–244. Springer, 2016. 2, 6

[29] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 5

[30] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967. 5

[31] 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(1):43–63, 2007. 2

[32] Gunho Sohn, Yoonseok Jwa, Jaewook Jung, and H Kim. An implicit regularization for 3d building rooftop modeling using airborne lidar data. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 1(3):305–310, 2012. 2

[33] Sinisa Stekovic, Mahdi Rad, Friedrich Fraundorfer, and Vincent Lepetit. Montefloor: Extending mcts for reconstructing accurate large-scale floor plans. *arXiv preprint arXiv:2103.11161*, 2021. 6

[34] Carole H Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Deep learning in medical image analysis and multimodal learning for clinical decision support*, pages 240–248. Springer, 2017. 2

[35] Meng Tang, Abdelaziz Djelouah, Federico Perazzi, Yuri Boykov, and Christopher Schroers. Normalized cut loss for weakly-supervised cnn segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1818–1827, 2018. 2

[36] Meng Tang, Federico Perazzi, Abdelaziz Djelouah, Ismail Ben Ayed, Christopher Schroers, and Yuri Boykov. On regularized losses for weakly-supervised cnn segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 507–522, 2018. 2

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 2, 4

[38] Kang Zhao, Jungwon Kang, Jaewook Jung, and Gunho Sohn. Building extraction from satellite images using mask r-cnn with building boundary regularization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 247–251, 2018. 2

[39] Stefano Zorzi, Ksenia Bittner, and Friedrich Fraundorfer. Machine-learned regularization and polygonization of building segmentation masks. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 3098–3105. IEEE, 2021. 2

[40] Stefano Zorzi and Friedrich Fraundorfer. Regularization of building boundaries in satellite images using adversarial and regularized losses. In *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 5140–5143. IEEE, 2019. 2