# Instance Segmentation with Mask-supervised Polygonal Boundary Transformers

Justin Lazarow  Weijian Xu  Zhuowen Tu
UC San Diego
{jlazarow, wex041, ztu}@ucsd.edu

## Abstract

*In this paper, we present an end-to-end instance segmentation method that regresses a polygonal boundary for each object instance. This sparse, vectorized boundary representation for objects, while attractive in many downstream computer vision tasks, quickly runs into issues of parity that need to be addressed: parity in supervision and parity in performance when compared to existing pixel-based methods. This is due in part to object instances being annotated with ground-truth in the form of polygonal boundaries or segmentation masks, yet being evaluated in a convenient manner using only segmentation masks. Our method,* ***BoundaryFormer****, is a Transformer based architecture that directly predicts polygons yet uses instance mask segmentations as the ground-truth supervision for computing the loss. We achieve this by developing an end-to-end differentiable model that solely relies on supervision within the mask space through differentiable rasterization. BoundaryFormer matches or surpasses the Mask R-CNN method in terms of instance segmentation quality on both COCO and Cityscapes while exhibiting significantly better transferability across datasets.*

## 1. Introduction

Image segmentation [23] and scene labeling [28, 30] are amongst the most studied topics in computer vision. In addition to pixel-wise masks, representing segments/objects using vectorized boundary representations has applications and significance in many downstream tasks such as shape recognition [2], tracking [4], image understanding [31], medical imaging [26], and 3D reconstruction [5].

The recently emerged instance segmentation task (objects or areas of interest) [10] greatly propels the practical significance for object segmentation. Unlike detection, instance segmentation predicts the detailed extent of an object rather than a coarse bounding box. However, while a bounding box can be represented by only two coordinate pairs and is therefore easily turned into a regression problem, there are difficulties when deciding how to best represent and predict the segmentation of an object. These difficulties combined with a historical preference for convolutional operations has led the computer vision community to become *mask-centric*. This entails almost all segmentation models relying on a spatially dense function which outputs a binary confidence to determine whether each pixel belongs to a particular object. This is in contrast to a *boundary-centric* notion where a sparse set of structured points are predicted to denote the boundary of the object in question. Polygons are one natural choice for this structure. However, because of some inherent difficulties along with mask-centric biases, polygons have large hurdles to overcome. First, there is not an immediately obvious metric (and thus loss) for polygons. Second, the training regimes of mask-based models often relies on operations and augmentations that are *inherently difficult* to robustly and efficiently implement on polygons directly. This includes even basic operations like cropping and intersection. Finally, *evaluation* of segmentation quality is performed with respect to masks and not contours which leads to a possible mismatch in training and


Figure 1. We present a model of instance segmentation which predicts the *boundaries* of each object in the form of a polygon. This treats instance segmentation as a regression problem and allows for end-to-end differentiability of predictions. At the same time, our model requires no sacrifices in terms of resulting segmentation quality nor introduces additional supervision requirements compared algorithms which directly predict masks.

Code at https://github.com/mlpc-ucsd/BoundaryFormer.

testing objectives when predicting polygons.

Furthermore, one might question **why even predict boundaries over masks?** Despite [10] attaining high quality masks for instance segmentation, obtaining object boundaries can be more suited towards certain tasks. While masks have a large degree of flexibility, they also lack an explicit topology which can lead to degenerate or undesirable behavior (*e.g.*, uncertainty at the boundary or unexpected interior holes). Explicit boundaries, however, facilitate downstream tasks such as plane detection [18] not only in the form of a rich, top-down structural prior but also as a continuous output which can be end-to-end differentiated in a larger system.

Despite many obstacles, past efforts to predict polygons within instance segmentation have been made. Nevertheless, none have been able to *achieve parity* with baseline mask-based segmentation models across commonly used benchmarks. By *parity*, we mean a few things:
1). **Parity in supervision:** The method should require no additional sources of supervision than its mask-based counterpart. In particular, the method *should not* rely on polygons directly as a source of supervision. This is crucial since polygons are not always available and deriving polygons from a mask-based ground truth can introduce suboptimal performance or uncertainty – see Table 4.
2). **Parity in evaluation:** While polygonal boundary annotations do exist in some instance segmentation datasets, *e.g.* [16], further technical barriers await when using polygon predictions for evaluation against the ground-truth directly. This is due to the vectorized polygon representation not being unique and there existing a one-to-many representation from masks to polygons. In other words, two similar masks may have polygons that have large differences in the control points, creating a mismatch between training loss and evaluation.

Finally, we also want to deal with 3). **Parity in access**. While the predictive model itself might deviate in terms of architecture, the model should be generally considered to be a "drop-in" replacement for a mask-based segmentation head. This includes working within one and two-stage architectures, *e.g.*, with respect to either full image or RoI features.

While certain works have touched upon aspects of our model [9, 14], none has yet to provide a fully polygon based solution that is accessible to a myriad of architectures and matches performance of mask-based architectures on standard datasets. We believe providing a clearer picture into the capabilities of polygons in providing a performant and end-to-end differentiable segmentation pipeline could be helpful to further development within the field. We outline our contributions below:

1. In this paper, we present a new instance segmentation method, BoundaryFormer, a Transformer based approach for predicting **an object's boundary as a polygon directly**. Our model outperforms the strong baseline Mask R-CNN [10] on the MS-COCO [16] dataset and achieves competitive results on Cityscapes [7] when training from scratch while significantly outperforming it when transferring from a COCO-based initialization. To the best of our knowledge, this is the first time a method with polygonal outputs has matched or exceeded Mask R-CNN on the MS-COCO dataset. Furthermore, BoundaryFormer does so without compromising its ability to be trained end-to-end.

2. BoundaryFormer uses pixel-wise masks as ground-truth for supervision and evaluation by utilizing a novel **differentiable rasterization** module. Therefore, BoundaryFormer *adds no new supervision* requirements over Mask R-CNN [10].

3. By only relying on masks as a source of supervision, our model can be placed as a **drop-in** replacement for the mask-based segmentation head of R-CNN [10] as either a full image-based component or an RoI-based component. Furthermore, it can be adopted in other common architectures including FCOS [29].

## 2. Related Work

We highlight past work which has predicted polygonal (or contour-based) segmentations and make special note of those which have adopted rasterization as a form of supervision for their models.

### 2.1. Point-based Losses

We first discuss contour-based segmentation algorithms which rely primarily on a matching between predicted points and ground-truth points sampled from a known polygon. Therefore, these methods *require* access to some polygonal ground-truth. As one of the first works built on modern architectures, DeepSnake [25] considers an initial octagonal polygon derived from four extreme points, after which an iterative process of refinement is carried out using circular convolutions. The resulting refinements are supervised using an $L_1$ loss against a uniform ground-truth polygon. On the other hand, PolarMask [34] models polygons in a polar representation along with an approximation to IoU as supervisory signal.

Applying a direct distance loss in a Transformer-based architecture for line segmentation detection [35] and pose recognition [13] exists, but they have the direct ground-truth supervision on the points.

PolyTransform [14] uses an off the shelf mask-based segmentation pipeline to predict an initial binary mask of an object from which highly accurate initial polygons(s) can be derived using an non-differentiable border following algorithm. These polygons are then deformed using Transform-

ers [32] after which a point-based loss is used for supervisory signal.

Building off of DeepSnake, DANCE [20] considers an FCOS [29] detector to produce an initial box proposal. Points are uniformly sampled along this box to produce an initial polygon which can be deformed using an attention-like process. This initial box contour allows for a novel ground-truth matching which can be more easily optimized. In addition, the attentive process plus predicted edge maps (which relies on access to panoptic annotations) improve performance.

### 2.2. Mask-based Losses

We next consider contour-based models which entirely or in-part rely on rasterized forms of predicted polygons as supervision. ACDRNet [9] provides a system which takes crops of objects of interest as input and iteratively deforms an initial contour by predicting a dense heatmap of offsets in feature-space. A 3D neural renderer [11] is applied to a triangulation of the predictions against the ground-truth masks using an MSE loss. In addition, ACDRNet relies on two additional losses: a balloon like loss to force expansion of the contour and a curvature term. While an interesting proof of concept, the authors do not consider integration into an actual detection pipeline and performance on the standard MS-COCO [16] is not considered.

CurveGCN [17] predicts polygonal boundaries through a graph convolutional neural network from an initial contour proposal. For the bulk of training, they use an ordinary Chamfer loss, however, they do note that fine-tuning their model with respect to a differentiable accuracy metric, i.e. triangulating the polygon into a mesh and supervising with respect to masks using a differentiable renderer [22] leads to improved results.

Lastly, some recent work [6] considers using boundary information as an additional supervision for instance segmentation, however, they still predict masks directly and not polygons.

## 3. Method

### 3.1. Setting

Instance segmentation considers an input image $I \in \mathbb{R}^{H \times W \times 3}$ and is tasked with producing $N$ ordered instances $O_i$, $1 \leq i \leq N$. Most modern benchmarks consider instances as a tuple $(M_i, c_i)$ where $M_i$ is a per-pixel binary mask denoting membership by the object and where $c_i \in \{1, \ldots, C\}$ is the predicted class of the object.

Since $M_i$ is a binary mask, most segmentation models find it natural to predict some downsampled version of $M_i$. $M_i$ is predicted as a discrete grid of continuous confidence values $M_i(x, y)$ with $x \in \{1, \ldots, X'\}$, $y \in \{1, \ldots, Y'\}$ for a mask size of $X' \times Y'$. This phrases mask prediction as a *classification problem*. At inference time, $M_i$ is transformed into a binary mask with an ordinary decision rule (usually $M_i(x, y) > 0.5$). Therefore, without approximations or relaxations, $M_i$ is *not differentiable* for downstream components relying on $M_i(x, y)$.

In this work, we phrase the prediction of $M_i$ rather as a *regression problem*. This decomposes the prediction of $M_i$ into two parts. First, we predict $K$ vertices $V_i = \{(x_0^i, y_0^i), \ldots, (x_{K-1}^i, y_{K-1}^i\}$ which define the boundary of a polygon in 2D under a fixed ordering. Then, we produce $M_i$ through rasterization to the desired mask size $X' \times Y'$. $M_i$ must be produced because instance segmentation relies on masks for evaluation. This leads to the following choices: how should $V_i$ be predicted and more importantly, how should $V_i$ be supervised with respect to the ground truth. We present our approach to this problem as **BoundaryFormer**.

### 3.2. Instance Segmentation with Mask-supervised Boundary Regression Transformers

We design BoundaryFormer as a component that can be added on to existing detection-based frameworks. While we believe BoundaryFormer is generally applicable, we consider a more concrete setup for the sake of presentation. In particular, we assume a detector built upon a standard FPN architecture (this includes FCOS [29] or R-CNN [10]) which produces feature maps $F = \{P_2, \ldots, P_5\}$ in decreasing resolution from the image $I$. From these features, the detector proposes $N$ objects in the form of boxes $B_i = (l_i, t_i, w_i, h_i)$ which denote the left edge, top edge, width, and height of the box respectively. Each $B_i$ corresponds to some ground-truth mask $M_i$ where $M_i$ is expected to be clipped to $B_i$. We use $B_i$ as a means to initialize an ellipsoidal polygon $V_i(0)$ inscribed in $B_i$, similar to other contour-based methods [14, 20, 25]. From $V_i(0)$, our model iteratively refines this shape $L$ times by $V_i(j+1) = g_j(F, V_i(j))$ to produce a final prediction $V_i(L)$.

We visualize a concrete implementation in Figure 2. Since we are dealing with point sets, we implement $g$ with Transformers [32] using two kinds of attention. Each vertex within the polygon $V_i$ corresponds to some point embedding $P_i^k$ where $1 \leq k \leq K$ and includes a "point encoding" modeled off of the usual Transformer sine positional encoding. The first kind of attention allows each $P_i^k$ to attend to all other point embeddings $P_i^{k'}$ within the same object. The second allows each $P_i^k$ to attend to the image features $F$. While the first kind of attention is implemented using ordinary (quadratic) self-attention, we implement the point to image feature attention using Deformable Attention [36] which significantly reduces the computational cost. Furthermore, this allows multi-scale *across* levels $P_2$ through $P_5$. Finally, for each embedding $P_i^k$, $g_j$ predicts a 2D offset

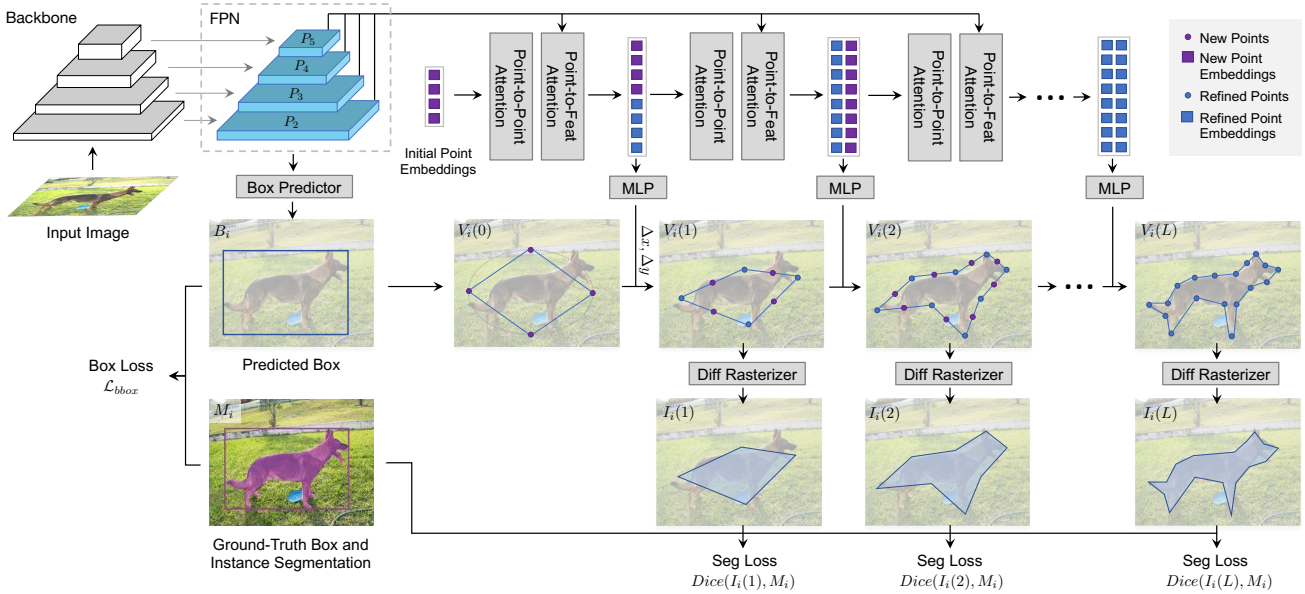



Figure 2. **Illustration of BoundaryFormer:** Given an image, multi-scale features are collected and boxes for each object are proposed by the underlying detector. We initialize a proposed polygon for each box using a simple ellipse. At each layer, attention between points within the polygon as well as attention from points to the image features is performed in order to refine the polygon. New points are inserted between existing ones in order to efficiently produce higher resolution polygons as the refinement progresses. After each refinement, the predicted polygon is rasterized in a differentiable manner and compared to the ground-truth mask using a mask-based loss.

$(\Delta x, \Delta y)$ to refine the vertex using an MLP.

### 3.3. Mask-based Supervision

Given the predictive model, we expect a final output polygon $V_i$ consisting of $K$ points for each proposed box $B_i$ from the underlying detector. However, in order to train such a model, we must provide a means of supervision. Most previous work has relied upon point-based matching losses where some scheme to assign a predicted point to a point in the *polygonal ground-truth* is devised. These might include: the Chamfer distance [14], permutation-based matching [17], and assignment rules that depend on the initial contour [20]. However, we believe these approaches are undesirable and unnecessary. First, they require access to ground-truth polygons which are not always available and attempting to derive polygonal contours from masks is a noisy process (Table 4). These ground-truth polygons might need to be sampled (up or down) against heuristics in order to satisfy the requirements of point-based losses. Second, we *evaluate* instance segmentation quality with respect to *masks* (i.e. COCO mask AP [16]), and therefore it's not guaranteed that these metrics are optimizing the desired metric. Lastly, many essential operations *are non-trivial* to implement on polygons directly. This includes: clipping to a box (required for RoI-like operations), intersection, and even union. However, the mask-based counterparts are simple, highly optimized, and differentiable in existing frameworks.

Therefore, we require our model to only require mask-based supervision and furthermore, to require it in the exact same sense as an ordinary mask-based segmentation model. We transparently handle this transformation from polygon space to mask space by the usage of a differentiable 2D rasterizer. Because of the dynamic nature of rasterization, we are afforded more flexibility than a purely mask-based model. We now describe specific details of the rasterization process.

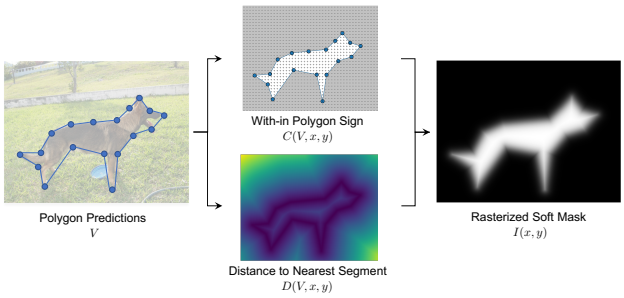



Figure 3. **Illustration of differentiable rasterizer:** We illustrate the transformation from polygonal point predictions to a differentiably rasterized mask: pixels are given signs based on being within the polygon or not, projections onto the nearest segment are computed, and Equation 1 determines the final rasterized value.

#### 3.3.1 A polygon-specific rasterizer

While previous work [9, 17] has used 3D renderers (leaving the depth component constant) to produce masks from polygons in a differentiable manner, these methods require the additional step of triangulation. We find this to be un-

necessary, especially accounting for the time necessary for triangulation and additional choice of triangulation method required. Rather, we use [19] as inspiration to design a rasterizer that operates on polygons directly rather than triangulated meshes.

The pipeline of the proposed differentiable rasterizer is shown in Figure 3. Consider a polygon with vertices $V$ and a desired rasterization pixel size of $X' \times Y'$. For each pixel $(x, y)$ where $0 \leq x < X', 0 \leq y < Y'$, we use a PnP algorithm [1] to determine whether the pixel at position $(x, y)$ lies within the polygon as $C(V, x, y)$, where $C = 1$ if it lies within and $-1$ if it does not. Then, each pixel $(x, y)$ is projected onto the closest segment on the polygon's boundary and this distance is recorded as $D(V, x, y)$. Finally, following [19], we model each pixel's contribution to the rasterized image as a sigmoidal function (with some sharpness $\tau$) according to the associated distance:

$$I(x, y) = \sigma\left(\frac{C(V, x, y) * D(V, x, y)}{\tau}\right) \tag{1}$$

Therefore, this rasterizer provides signal solely from the boundary without the need to consider derived mesh points.

The rasterizer and backwards pass are implemented entirely in CUDA. This affords us efficiency to train solely with the differentiable rasterizer for the entirety of training. We find the Dice [24] loss to be critical to the success of training with rasterized masks, although it's likely similar losses, *e.g.*, Lovasz-Softmax [3] perform equally well.

#### 3.3.2 Alignment

**Alignment with ground-truth:** We emphasize that we use the exact same ground-truth masks (although possibly at differing resolutions) as an ordinary R-CNN pipeline. Thus, it is imperative that we ensure our differentiable rasterization is "aligned" to the method of rasterization built into the COCO API [16]. Due to differences in what is considered a "pixel", we ensure that we subtract a half pixel on all coordinates before passing them to our rasterizer. This significantly improves alignment and thus performance, *e.g.*, mask AP **drops** from 36.1 AP to 35.3 AP when not aligned.
**Alignment with architecture:** When building off of existing architectures, *e.g.*, FPN [15], operations often have a bias towards alignment with convolutions. We find that when we attempt to use BoundaryFormer in an RoI-less setting, *i.e.*, directly attending to the full image-based features, it is essential to perform a global pooling-like operation to significantly increase performance — without this, mask AP **drops** from 36.1 AP to 34.2 AP. We hypothesize this might be due to aliasing in the deconvolutional process of the FPN.

### 3.4. Coarse to fine upsampling

By having each point dedicated to a single point query embedding, we afford a great deal of flexibility to our model. However, since often we might have a large number $O$ of objects or object proposals and $K$ control points for each object, this can require computation on the order of $O * K * L$ if we include $L$ layers. At the same time, it's unlikely that we truly need $K$ vertices until the prediction is itself more accurate. Therefore, we consider a base number of control points $B$ (usually 8) and upsample the points by $2\times$ at each layer. Specifically, given $V_i(l)$ consisting of $K(l)$ points, we first apply $g_l(G, V_i(l))$ to get refinements to produce $V_i(l+1)$ still consisting of $K(l)$ points. Then, between each point $(x_j, y_j)$ and $(x_{j+1}, y_{j+1})$, we insert a new point at the midpoint $\left(\frac{x_j + x_{j+1}}{2}, \frac{y_j + y_{j+1}}{2}\right)$. We do not average the corresponding point embeddings $P_j$ and $P_{j+1}$ to initialize the new point and instead insert the corresponding "learned" point embedding . Overall, this speeds up training and memory consumption by about $1.5\times$ versus using the same number of points at each layer.

### 3.5. Loss

We finally present the loss of our polygon-based boundary prediction model trained jointly with a box-based detector. Suppose the detector is trained against $\mathcal{L}_{bbox}$. We denote the subset of foreground-matched boxes as $\{B_i \mid 1 \leq i \leq R\}$ and associate ground-truth mask $M_i$ of resolution $X' \times Y'$ to each. If our predictive model of polygons consists of $L$ layers, then we have dense outputs corresponding to $V_i(l)$ with $1 \leq l \leq L$. Then, using $I_i(l)$ to denote the (differentiably) rasterized version of $V_i(l)$, we define a total loss:

$$\mathcal{L} = \mathcal{L}_{bbox} + \sum_{l}^{L} \sum_{i=1}^{R} Dice\left(I_i(l), M_i\right) \tag{2}$$

This loss provides *parity in supervision* since like a standard instance segmentation model, it relies only on access to ground-truth masks $M_i$.

## 4. Experiments

We evaluate the *parity in performance* of BoundaryFormer across the commonly used MS-COCO dataset [16] and Cityscapes dataset [7]. We additionally provide experiments for transferring models from MS-COCO to Cityscapes to illustrate possible differences based on the underlying representation learned. We focus on comparisons with Mask R-CNN [10] as the standard baseline for instance segmentation, although, we calibrate our results with other contour-based and masked-based models. Furthermore, we consider inclusion of BoundaryFormer in both single stage, FCOS [29], and two stage approaches, Faster R-CNN [27].

### 4.1. Training Details

All models are trained in an end-to-end fashion with respect to the entire network. We aim to keep underlying

backbones and architectures as close as possible to Mask R-CNN [10] to compare as fairly as possible. Therefore, unless specified, we use a ResNet50-FPN backbone for all models and follow the exact same settings as Mask-RCNN in Detectron2 [33] with only the "mask head" of Mask R-CNN replaced with the architecture outlined in Figure 2. While Mask R-CNN *requires* RoI pooling to predict masks, this feature is optional in our approach and frees us from constraints that such a grid-based approach imposes. Rather, we present results *without* the need of RoI pooling such that the point to image features attention is with respect to the entirety of $\{P_2, \ldots, P_5\}$. One notable exception to standard training is that instead of SGD, we train all models with the Adam [12] optimizer according to the settings outlined in Swin [21] due to our use of Transformers (except a weight decay of 0.20 for larger models).

For details specific to BoundaryFormer, we train all models with coarse to fine upsampling over 4 layers of Transformers [32] to produce a final output of 64 points on COCO and 128 on Cityscapes. All other parameters with respect to deformable attention follow the same settings of Deformable DETR's [36] decoder layer.

For rasterization, we find that $\tau = 0.1$ (denoting rasterization smoothness) generally performs well. Additionally, we rasterize polygons during training to a fixed $X' \times Y' = 64 \times 64$ resolution. When performing rasterization, we *only rasterize the predicted polygon within its box*, not with respect to the entire image. This emulates the behavior of RoI pooling. Like Mask R-CNN [10], the ground-truth mask is clipped to this box.

**Inference:** At inference time, we follow the same inference procedures as the underlying detector. We rely on rasterization from the COCO API directly [16] rather than our own rasterizer since differentiability is not required.

## 4.2. COCO

COCO [16] is a large-scale dataset containing 118K training images of natural scenes with 80 annotated foreground classes. It has historically been a relatively difficult dataset for contour or boundary-based models to achieve parity in performance with mask-based approaches. As far as we can tell, our model *is the first to achieve parity in mask quality to Mask R-CNN on COCO*.

We detail these comparisons in Tables 1 and 2. We note that BoundaryFormer achieves a slight edge in mask quality over Mask R-CNN and is competitive with the boundary-preserving variant [6] while significantly outperforming the best contour-based method [20]. At the same time, we observe that both models attain the same box performance — indicating that both tasks (mask or polygon prediction) provide multi-task training benefits to the underlying detector when trained end-to-end. While we find the best results using R-CNN as an underlying architecture, BoundaryFormer can still match Mask R-CNN when using FCOS. Performance appears to be slightly worse despite performing better with respect to box AP. We hypothesize this might be due to FCOS using only P3-P7 within its FPN whereas R-CNN includes P2 and thus might learn enhanced features relevant to boundary prediction.

| Method | Backbone | Detector | AP | $AP_{50}$ | $AP_{bbox}$ |
|---|---|---|---|---|---|
| Mask R-CNN [10] | R50-FPN | R-CNN | 35.2 | 56.3 | 38.6 |
| Mask R-CNN* [10] | R50-FPN | R-CNN | 35.8 | 56.8 | 38.8 |
| BMask R-CNN [6] | R50-FPN | R-CNN | 36.6 | 56.7 | 39.4 |
| BMask R-CNN* [6] | R50-FPN | R-CNN | 36.4 | 56.3 | 37.8 |
| DANCE [20] | R50-FPN | FCOS | 34.5 | 55.3 | 40.2 |
| BoundaryFormer (ours) | R50-FPN | FCOS | 35.8 | 55.7 | 40.2 |
| BoundaryFormer (ours) | R50-FPN | R-CNN | 36.1 | 56.7 | 38.8 |

Table 1. **Results on MS-COCO val:** We compare BoundaryFormer to the state of the art in contour/boundary prediction [20] as well as mask-based counterparts. * indicates re-trained with Adam [12].

## 4.3. Cityscapes

Cityscapes [7] consists of a diverse set of street scenes across European cities. The dataset is relatively small compared to COCO — consisting of only 2975 training images, however, each image is of a high resolution and annotation quality. Furthermore, it is notable for its large amount of occlusion [10] which causes a significant number of instances to be fragmented into multiple simple polygons. This creates problems for boundary prediction which usually only consists of predicting and matching to a single ground-truth polygon. While we compare to other methods which include mitigations (*e.g.*, initialization from masks [14]), we emphasize that BoundaryFormer *uses no special handling* and does not deviate from the standard training process.

We highlight the performance of our method with respect to both ImageNet and COCO initializations. COCO initializations are made using a model trained under the standard 1x training schedule, i.e., the models in Table 1. Mask R-CNN [10] and BoundaryFormer results are the average of three runs to account for well-known instabilities in Cityscapes training.

We observe that BoundaryFormer is still able to achieve parity with Mask R-CNN despite being at a significant disadvantage by predicting a single polygon for fragmented objects. Surprisingly, when initializing BoundaryFormer from COCO, it significantly exceeds Mask R-CNN in final performance which might indicate a polygonal representation transfers well.

## 4.4. Ablation studies

We provide ablations for the hyperparameters of both the supervision (the rasterization process) and some model-specific hyperparameters/justifications.

**Polygonal ground-truth from masks?** While we promote

| Method | Backbone | Detector | Schedule | Output | Supervision | AP | $AP_{50}$ | $AP_{75}$ | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Mask R-CNN [10] | R50-FPN | R-CNN | 1× | masks | masks | 36.1 | 57.3 | 38.7 | 19.7 | 38.0 | 47.1 |
| Mask R-CNN [10] | R101-FPN | R-CNN | 3× | masks | masks | 39.2 | 61.1 | 42.3 | 22.4 | 41.4 | 50.7 |
| BMask R-CNN [6] | R50-FPN | R-CNN | 1× | masks | masks | 36.6 | 57.0 | 39.6 | 19.4 | 38.7 | 48.0 |
| BMask R-CNN [6] | R101-FPN | R-CNN | 1× | masks | masks | 38.3 | 59.2 | 41.3 | 20.3 | 40.8 | 50.2 |
| DeepSnake [25] | R50-FPN | CenterNet | - | polygons | polygons | 30.5 | - | - | - | - | - |
| PolarMask [25] | R101-FPN | FCOS | 2× | polygons | polygons | 32.1 | 53.7 | 33.1 | 14.7 | 33.8 | 45.3 |
| DANCE [20] | R50-FPN | FCOS | 1× | polygons | polygons + panoptic | 34.6 | 55.9 | 36.4 | 19.3 | 37.2 | 43.9 |
| DANCE [20] | R101-FPN | FCOS | 3× | polygons | polygons + panoptic | 38.1 | 60.2 | 40.5 | 21.5 | 40.7 | 48.8 |
| BoundaryFormer (ours) | R50-FPN | R-CNN | 1× | polygons | masks | 36.4 | 57.2 | 39.0 | 19.6 | 38.6 | 47.9 |
| BoundaryFormer (ours) | R101-FPN | R-CNN | 1× | polygons | masks | 37.7 | 58.8 | 40.5 | 20.4 | 40.2 | 49.0 |
| BoundaryFormer (ours) | R101-FPN | R-CNN | 3× | polygons | masks | 39.4 | 60.9 | 42.6 | 22.1 | 42.0 | 51.2 |

Table 2. **Results on MS-COCO test-dev:** We evaluate and compare BoundaryFormer on the MS-COCO test-dev set with larger backbones and longer training schedules, showing that its performance scales as expected while staying competitive with strong mask-based baselines.

| Method | model init | poly init | e2e | sup | AP | $AP_{50}$ |
|---|---|---|---|---|---|---|
| Mask R-CNN* [10] | ImageNet | N/A | - | masks | 34.2 | 60.7 |
| Mask R-CNN* [10] | COCO | N/A | - | masks | 36.5 | 62.0 |
| DeepSnake [25] | ImageNet | extreme pts | ✓ | poly | 28.2 | - |
| UPSNet [14] | COCO | pred masks | - | masks | 37.8 | - |
| PolyTransform [14] | COCO | pred masks | - | both | 40.2 | - |
| BoundaryFormer | ImageNet | ellipse | ✓ | masks | 34.7 | 60.8 |
| BoundaryFormer | COCO | ellipse | ✓ | masks | 38.3 | 62.9 |

Table 3. **Results on Cityscapes val:** We establish that BoundaryFormer can maintain parity with Mask R-CNN even on the difficult Cityscapes dataset. Additionally, when using a model initialized from COCO, BoundaryFormer shows improved transfer ability over Mask R-CNN and is competitive with PolyTransform, which requires the use of a mask head to initialize its contours. Lastly, BoundaryFormer is the only model that both relies solely on ground-truth masks for supervision and is end-to-end differentiable. * indicates re-trained with Adam [12].

the usage of masks directly as supervision of our model for both performance and flexibility, contour models that require access to polygonal ground-truth could resort to generating contours from masks themselves when masks are the only available annotation source. Therefore, we re-generate the COCO training dataset (where polygonal ground-truth *is* available) and replace the existing segmentations with those generated using a border following algorithm (*i.e.*, `cv2.findContours`) in order to retrain DANCE [20]. In Table 4, we observe sharp decreases in resulting performance as measured by the unmodified MS-COCO *val* set — indicating that generating polygons from masks can introduce errors. While it's plausible this could be improved, we believe it acts as an unnecessary barrier in producing an accurate, certain model.

| DANCE [20] | Polygons (verbatim) | Polygons (from masks) |
|---|---|---|
| Mask AP | 34.5 | 23.1 |

Table 4. **Comparison in training a point-based supervised model using verbatim polygons from annotators or those generated using standard algorithms from masks.**

**Rasterization smoothness ($\tau$):** The rasterization smoothness dictates the (inverse) steepness of the sigmoidal function in Equation 1. While it must be sufficiently large to allow good gradient flow, it should also be small enough to accurately reflect what will be predicted at inference time, *i.e.*, the *hard* rasterization of the predicted polygons. We find that lower values of $\tau$ are required (i.e. sharper), however, within that lower range, the performance is generally robust with values around $\tau = 0.1$ to be sufficient. Larger values, *e.g.*, $\tau = 1.0$ lack sufficient sharpness for optimal results and mask AP **drops** to 35.6.

**Rasterization resolution ($X' \times Y'$):** Like Mask R-CNN, each instance is supervised at a fixed resolution with respect to some mask loss. We investigate the impact of the choice of resolution on resulting performance. In our experiments, we find that $64 \times 64$ performs well with performance saturating at larger resolutions. However, this might be a consequence of the lower quality annotations in COCO and these characteristics might change on higher quality annotations, *e.g.*, LVIS [8].

| $X' \times Y'$ | $14 \times 14$ | $20 \times 20$ | $40 \times 40$ | $64 \times 64$ | $80 \times 80$ |
|---|---|---|---|---|---|
| Mask AP | 34.5 | 35.5 | 35.9 | 36.1 | 36.1 |

**Number of control points/layers ($K, L$):** We investigate the tradeoff in performance given the number of initial points, number of layers, and whether to use a coarse to fine strategy (i.e., $K_1 \neq K_L$). We find that even with only two layers, the model still performs reasonably well — suffering only a 1 point drop in Mask AP. Furthermore, we find using less points to lead to similar drop, however, we observe no degradation in performance when comparing our coarse to fine strategy to a dense one.

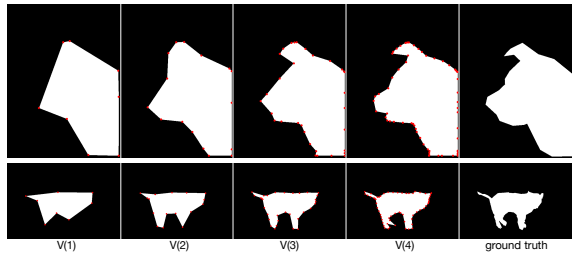
| $L/K_1/K_L$ | 2/32/64 | 3/16/64 | 4/4/32 | 4/8/64 | 4/64/64 | 4/16/128 |
|---|---|---|---|---|---|---|
| Mask AP | 35.0 | 35.7 | 35.2 | 36.1 | 36.1 | 36.2 |

## 5. Qualitative Analysis

We briefly investigate the general qualitative quality of predicted polygons in Table 4. Furthermore, we observe

Figure 4. **BoundaryFormer is able to predict high quality instance segmentations over the MS-COCO dataset:** Predicted boundary points are shown in black. The resulting rasterized mask is overlayed onto the instance in color.

some robustness to fragmented objects in Figure 5b where multiple polygons are required for to make a perfect prediction. Nonetheless, BoundaryFormer has implicitly learned a reasonable single-polygon approximation — likely owing to its supervision in the form of masks. Finally, we exhibit the approximations our coarse to fine strategy learns in Figure 5a and improvements in quality it makes as more and more points become available.




(a)

(b)

Figure 5. **Progressive refinement and fragment robustness (a)** illustrates the progressive improvement in boundary quality over layers. **(b)** shows some implicitly learned robustness to fragmented objects (ground-truth is shown in the top-left).

## 6. Conclusion

We presented BoundaryFormer, a simple baseline for regressing instance segmentations as polygonal boundaries rather than predicting dense masks. Despite regressing polygons, this model relies on supervision *only through masks*. Combined with a strong point-based architecture and supervision in pixel space, BoundaryFormer can match and outperform mask-based counterparts across a variety of datasets. As a result, we believe that many tasks that have historically been mask-based, *e.g.*, semantic and panoptic segmentation can now be revisited. Furthermore, we hope downstream tasks that have relied on non-differentiable mask predictions can now consider using the sparse representation of a polygon and the end-to-end differentiability it provides. Finally, we believe further research can alleviate some limitations: predicting fragmented objects faithfully, incorporating additional mask-based advancements, and finding more efficient architectures. While this work *predicts* a sparse representation, it nonetheless requires the full, dense form of an object's mask for supervision — raising the issue of whether the supervision could be made sparser and more boundary-centric.

**Societal impact:** Our work is concerned with predicting extents of objects within images. Like other systems of detection and recognition, both bias learned from datasets as well as misuse could cause societal harm. However, we believe our work is consistent with the existing risks in advancing such systems.

**Acknowledgement:** This work is supported by NSF Award IIS-1717431 and NSF Award IIS-2127544. We thank Qualcomm Inc. for the Qualcomm Faculty award.

## References


[1] Pnpoly - point inclusion in polygon test. https://wrf.ecse.rpi.edu/Research/Short_Notes/pnpoly.html. Accessed: 2021-10-30. 5

[2] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE transactions on pattern analysis and machine intelligence*, 24(4):509–522, 2002. 1

[3] 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. 5

[4] Andrew Blake and Michael Isard. *Active contours: the application of techniques from graphics, vision, control theory and statistics to visual tracking of shapes in motion*. Springer Science & Business Media, 2012. 1

[5] Anne-Laure Chauve, Patrick Labatut, and Jean-Philippe Pons. Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data. In *CVPR*, pages 1261–1268, 2010. 1

[6] Tianheng Cheng, Xinggang Wang, Lichao Huang, and Wenyu Liu. Boundary-preserving mask r-cnn. In *ECCV*, pages 660–676, 2020. 3, 6, 7

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 2, 5, 6

[8] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5356–5364, 2019. 7

[9] Shir Gur, Tal Shaharabany, and Lior Wolf. End to end trainable active contours via differentiable rendering. *arXiv preprint arXiv:1912.00367*, 2019. 2, 3, 4

[10] 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. 1, 2, 3, 5, 6, 7

[11] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6, 7

[13] Ke Li, Shijie Wang, Xiang Zhang, Yifan Xu, Weijian Xu, and Zhuowen Tu. Pose recognition with cascade transformers. In *CVPR*, pages 1944–1953, 2021. 2

[14] 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, 3, 4, 6, 7

[15] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 5

[16] 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. 2, 3, 4, 5, 6

[17] 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. 3, 4

[18] Chen Liu, Jimei Yang, Duygu Ceylan, Ersin Yumer, and Yasutaka Furukawa. Planenet: Piece-wise planar reconstruction from a single rgb image. In *CVPR*, 2018. 2

[19] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019. 5

[20] Zichen Liu, Jun Hao Liew, Xiangyu Chen, and Jiashi Feng. Dance: A deep attentive contour model for efficient instance segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 345–354, January 2021. 3, 4, 6, 7

[21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 6

[22] Matthew M Loper and Michael J Black. Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision*, pages 154–169. Springer, 2014. 3

[23] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, volume 2, pages 416–423, 2001. 1

[24] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. IEEE, 2016. 5

[25] Sida Peng, Wen Jiang, Huaijin Pi, Xiuli Li, Hujun Bao, and Xiaowei Zhou. Deep snake for real-time instance segmentation. In *CVPR*, 2020. 2, 3, 7

[26] Dzung L Pham, Chenyang Xu, and Jerry L Prince. Current methods in medical image segmentation. *Annual review of biomedical engineering*, 2(1):315–337, 2000. 1

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015. 5

[28] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006. 1

[29] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019. 2, 3, 5

[30] Zhuowen Tu. Auto-context and its application to high-level vision tasks. In *CVPR*, 2008. 1

[31] Zhuowen Tu, Xiangrong Chen, Alan L Yuille, and Song-Chun Zhu. Image parsing: Unifying segmentation, detection, and recognition. *International Journal of computer vision*, 63(2):113–140, 2005. 1

[32] 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. 3, 6

[33] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 6

[34] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12193–12202, 2020. 2

[35] Yifan Xu, Weijian Xu, David Cheung, and Zhuowen Tu. Line segment detection using transformers without edges. In *ICCV*, pages 4257–4266, 2021. 2

[36] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3, 6