Box2Poly: Memory-Efficient Polygon Prediction of Arbitrarily Shaped and Rotated Text

Xuyang Chen[1,2]**, Dong Wang**[1*]**, Konrad Schindler**[3]**, Mingwei Sun**[1,4]**,**
Yongliang Wang[1]**, Nicolo Savioli**[1]**, Liqiu Meng**[2]

[1]Riemann Lab, Huawei
[2]Technical University of Munich
[3]ETH Zurich
[4]Wuhan University
{xuyang.chen, wangdong136, sunmingwei2, wangyongliang775,
nicolo.savioli}@huawei.com,schindler@ethz.ch,liqiu.meng@tum.de

Abstract

Recently, Transformer-based text detection techniques have sought to predict polygons by encoding the coordinates of individual boundary vertices using distinct query features. However, this approach incurs a significant memory overhead and struggles to effectively capture the intricate relationships between vertices belonging to the same instance. Consequently, irregular text layouts often lead to the prediction of outlined vertices, diminishing the quality of results. To address these challenges, we present an innovative approach rooted in Sparse R-CNN: a cascade decoding pipeline for polygon prediction. Our method ensures precision by iteratively refining polygon predictions, considering both the scale and location of preceding results. Leveraging this stabilized regression pipeline, even employing just a single feature vector to guide polygon instance regression yields promising detection results. Simultaneously, the leverage of instance-level feature proposal substantially enhances memory efficiency ($> 50\%$ less vs. the SOTA method DPText-DETR) and reduces inference speed ($> 40\%$ less vs. DPText-DETR) with comparable performance on benchmarks. The code is available at https://github.com/Albertchen98/Box2Poly.git.
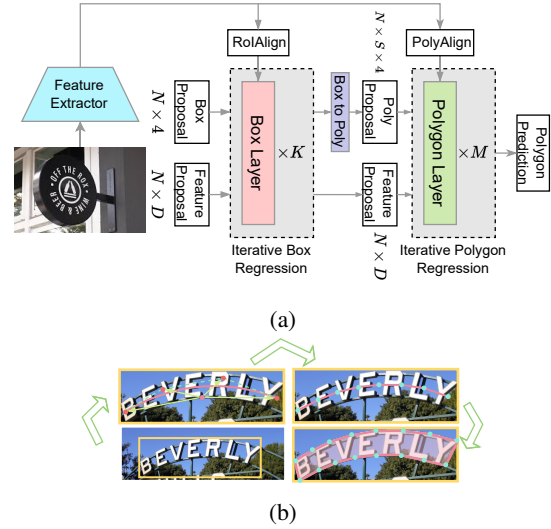
Figure 1: (a) The overview structure of Box2Poly. (b) Transformation from Box to Poly.

Introduction

In an increasingly digitized world, the ability to automatically detect and extract textual information from images is an indispensable component for many applications of machine vision, including autonomous driving (Zhu et al. 2017; Sun and Liu 2022), SLAM (Li et al. 2020), Visual Place Recognition (Hong et al. 2019), and Assisted Navigation (Rong et al. 2016). A reliable text detector is essential for localizing or parsing written text within a scene. Recent text detection methods have diversified the regression targets to adapt to the diverse appearance of text instances. This variability typically arises in terms of text orientation and curvature level.

Text orientation encompasses the possibilities of horizontal alignment or rotation, while text curvature spans the spectrum from straight to intricately bent. When dealing with text detection in cases where the layout is primarily straight

and orientation is horizontal, a viable approximation of the effects of perspective projection caused by pinhole cameras is achievable through rotating and scaling bounding boxes (Zhang et al. 2016; Ma et al. 2018; He et al. 2021) or utilizing quadrilaterals (Liu and Jin 2017; Bi and Hu 2021).

Besides, for text instances with arbitrary shapes, where rotation and curvature can be present in a variety of configurations, the use of mask-based representations and contour-based methods has gained prominence due to their heightened adaptability.

Mask-based approaches (Baek et al. 2019; Wang et al. 2019; Xu et al. 2019; Long et al. 2018) initially generate predictions at the pixel level, subsequently assembling individual pixels to construct segmentation masks at the instance level. Because masks possess nature at encapsulating deformable objects, these techniques offer a versatile solution for detecting text with irregular shapes. Nonetheless, their practical implementation is restricted by their substantial memory requirements and computationally intensive post-

---

processing steps. Alternative approaches (Lyu et al. 2018; Huang et al. 2022; Zhang et al. 2019) approach the detection task by framing it as instance segmentation. They incorporate a mask prediction head to generate segmentation masks from RoI(Region of Interest) proposals, which helps mitigate the computational workload to a certain degree, but the high-dimensional masks are hard to be effectively learned.

On the other side, contour-based methods directly regress predictions towards the ground truth contours. Their effectiveness is notably impacted by the initial quality of the polygon proposals acquired. Therefore, several approaches (Dai et al. 2021a; Zhang et al. 2023) further explore mask-based techniques through initializing contour proposals from binary segmentation masks, followed by contour refinement. In a different vein, several methods (Ye et al. 2023a; Zhang et al. 2022) demonstrates a novel perspective by utilizing bounding boxes as the initial estimation of contours, either implicitly or explicitly. However, achieving a seamless transformation amidst the shape disparity between the box and the polygon contour remains an open challenge. Simultaneously, the memory efficiency of point query embedding is a notable concern.

In this paper, we delve deeper into the *box-to-polygon* pipeline, focusing on addressing these issues.

As aforementioned, shape inconsistency arises during the box-to-polygon transformation process. This is especially evident when polygon vertices are formed by sampling points on boundaries (Ye et al. 2023a). As a result, curved text instances that rotate more than 45 degrees incur an additional learning cost due to the misaligned initialization. A distinctive proposition introduced by DeepSolo (Ye et al. 2023b) involves utilizing top-K Bézier curves (Liu et al. 2020) as initial proposals for the regression of the center polyline in text instances. While this approach enhances the suitability of the proposals for arbitrary shapes, it overlooks scale priors when generating Bézier curves across different feature map levels. Simultaneously, rather than refining the boundary directly, the center polyline undergoes recursive refinement during coordinate decoding. This makes the predictions more inclined to failure cases, such as self-intersection and misalignment to ground truth. We address these concerns by generating polygon proposal from bounding box using Bézier centerline as intermediary. As illustrated in Fig. 1(b), given a text instance enclosed by a bounding box, a Bézier curve is generated within and expanded to match the same scale as the box. Subsequently, a polyline is derived by sampling a fixed number of vertices from this intermediate representation. Finally, we expand the polyline in its orthogonal direction to create a polygon, which participates into subsequent iterative regression. In this approach, the scale of polygon is inherited from the bounding box, while the Bézier centerline guarantees the flexibility of the orientation and shape.

The memory footprint overhead can be mitigated in two perspectives. First, unlike two-stage DETR paradigm (Ye et al. 2023a; Zhang et al. 2022; Ye et al. 2023b), which generates Top-K proposals on the output of transformer encoder, we opt for a sparse proposal initialization (Sun et al. 2021) to alleviate computational demands and reduce memory us-

age. This sparse proposal initialization involves introducing a fixed set of instances at the outset of coordinate decoding. As aforementioned, directly initializing text polygons leads to instability and complications, we utilize learnable bounding boxes as their prior estimates. We initiate the process by regressing the box proposals before proceeding to learn the polygon representations (refer to Fig. 1). Furthermore, we employ instance-level feature embedding[1] for each instance proposal. In contrast, TESTR (Zhang et al. 2022) and DPText-DETR (Ye et al. 2023a) utilize point-level feature embedding for individual vertices, encoding their coordinates and interrelationships within each polygon instance.

Undoubtedly, instance-level feature embedding lacks the representative capacity when contrasted with point-level one. In light of this, our approach adopts the iterative regression strategy (Cai and Vasconcelos 2018; Sun et al. 2021) to ensure the stable regression of polygon coordinates. Diverging from refining with instance-independent absolute offsets (Ye et al. 2023a), the predicted refinement offset in our method is made to be scale- and translation-invariant.

Orientation discontinuity exists when the predicted polygon possesses reversed orientation w.r.t. ground truth, this hurdles the learning process since the network needs extra steps to correct the "wrong" orientation. To accelerate the convergence, we augment the ground truth by extending the orientation-including annotation to be orientation-equivalent.

Sparse R-CNN (Sun et al. 2021) is a representative work depending on iterative regression that achieves accurate results and high data-efficiency in object detection. At each decoding layer, RoIAlign (He et al. 2017) is leveraged to extract image RoI feature for each box proposal. To conduct the task of polygon prediction, we propose a RoIAlign-like operation PolyAlign to assist the regression of polygons.

In summary, the objectives of this paper are as follows:

- Propose an **memory efficient** and **computationally efficient** text polygon prediction method with **comparable performance to SOTA**.

- Thoughtfully devise the **smooth transformation from box to polygon**.

- Introduce RoIAlign-like operation to extract **precise RoI features for polygon regression**.

- Stabilize the training process by **recursively refining polygon proposals**.

Related Works

Iterative Regression Methods in Object Detection

The iterative regression technique has garnered significant attention within the realm of bounding box object detection. Cascade R-CNN(Cai and Vasconcelos 2018) extends the architecture of Faster R-CNN (Ren et al. 2015) by introducing multi-stage detection sub-networks. This breaks down the regression task into a series of cascade layers, where each layer's regressor operates directly on the prediction results provided by the previous layer. This sequential refinement

---

[1]corresponding to the term *object query* in DETR (Carion et al. 2020) and *proposal feature* in Sparse RCNN (Sun et al. 2021)
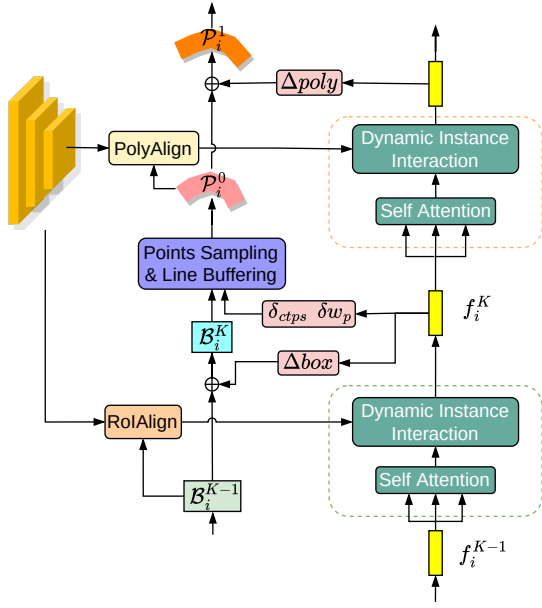
Figure 2: Illustration of the last bounding box regression iteration and the first polygon regression iteration.

process contributes to an improvement in detection quality. Similarly, within the framework of DETR methods, such as Deformable DETR (Zhu et al. 2020) and DAB DETR (Dai et al. 2021b), a comparable strategy is employed. These methods also embrace iterative bounding refinement, where the regression head of each decoder layer predicts bounding boxes as relative offsets w.r.t the bounding boxes of the preceding layer. This iterative approach contributes to the gradual enhancement of localization accuracy. In a different vein, Sparse R-CNN (Sun et al. 2021) introduces an innovative paradigm by circumventing the computationally intensive proposal generator in the Fast R-CNN paradigm. Instead, they initialize learnable bounding boxes with thoroughly sparse setting and subsequently update them using a methodology akin to that of Cascade R-CNN. Despite its streamlined pipeline, this approach achieves remarkable performance and training efficiency.

Text Detection With Contour Iterative Regression

Recognizing the complexity of accomplishing the regression in a single pass, researchers have explored the application of iterative regression to predict concise contour boundaries for text instances, An example is TextBPN++ (Zhang et al. 2023), where the boundary originates from a pixel-level score map and is subsequently refined through recursive optimization facilitated by a boundary transformer. However, this transformation from a score map to a boundary is heavily reliant on predefined rules and the pixel-level prediction restricts its application to higher input resolution. Leveraging on the impressive performance of DETR methods, TESTR (Zhang et al. 2022) takes a novel approach rooted in Deformable DETR (Zhu et al. 2020). It harnesses bounding boxes predicted by the transformer encoder's output as

proposals and then transforms these boxes into polygons by sampling points along the upper and lower sides. Subsequently, the generated polygon vertex is embedded into point query through sine positional encoding and participates into coordinate regression at each layer of the decoder. A further advancement, DPText-DETR (Ye et al. 2023a), refines the TESTR framework by introducing iterative refinement of polygon vertices during decoding. It also introduces the utilization of vertex coordinates as reference points for conducting deformable cross-attention (Zhu et al. 2020), in response to each point query. DPText-DETR surpasses the performance of TESTR and emerges as the new state-of-the-art method across several text detection benchmarks. However, DPText-DETR shares a common challenge with TESTR, namely, a substantial memory footprint. This is attributed to the requirement of encoding coordinate information for each polygon vertex with an individual query.

Method

Overview

The overall structure of Box2poly Network is illustrated in Fig. 1(a), which is built upon Sparse R-CNN (Sun et al. 2021). The image containing text instance is fed into an FPN backbone to extract multiple layer features. Then the decoding part on the right can leverage these features to complete the detection task. Like in Sparse R-CNN, a fixed, learnable set of N bounding boxes serve as region proposals, and each of those proposals is corresponding to a feature proposal of dimension D. The network performs structured regression with two heads, one for boxes and one for polygons. Each of them contains K and M layers respectively. The box proposals serve as priors for the subsequent polygon prediction. To transition from box prediction to polygon representation, a transitional layer *Box to Poly* is inserted between the last layer of box head and the first layer of polygon head. PolyAlign is leveraged to extract precise RoI features for polygon regression. Besides that, orientation-equivalent annotation for text polygon is introduced to eliminate the learning discontinuity when the predicted polygon shows reversed orientation w.r.t. the spatially proximal ground truth one.

Learnable Bounding Boxes as Priors

We leverage learnable bounding boxes to estimate the location and scale of text instances before polygon regression. Following Sparse RCNN (Sun et al. 2021), a set of learnable bounding boxes are initialized with the image size and refined recursively for K times. As illustrated in Fig. 2, given a box proposal $\mathcal{B}_i^{K-1}, i \in N$ from previous layer, RoIAlign (He et al. 2017) is leveraged to extract RoI feature within its scope. This extracted RoI feature is merged with its corresponding proposal feature f_i^{K-1} using Dynamic Instance Interaction (Sun et al. 2021), yielding an updated proposal feature f_i^K. To further refine the bounding box proposal $\mathcal{B}_i^{K-1}$, a coordinate offset Δbox is predicted based on f_i^K to achieve $\mathcal{B}_i^K$.
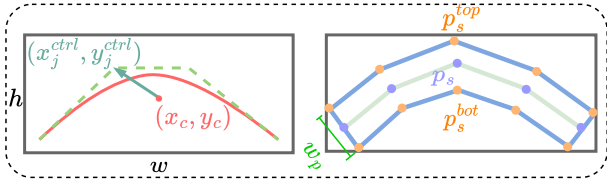
Figure 3: Ilustration of transforming from box to polygon with Bézier curve as intermediate representation.

Box to Poly

To conduct polygon regression, we first transform the bounding box prior $\mathcal{B}_i^K$ to polygon representation using Bézier curve (Liu et al. 2020) as intermediary. As depicted in Fig. 2 and Fig. 3, at the final layer $(K-1)$ of iterative bounding box regression, a Bézier curve with four control points is predicted upon proposal feature f_i^K as relative offsets $[\delta p_j = (\delta x_j, \delta y_j)]_{j=0}^3$ w.r.t. the center of the reference box. We denote reference bounding box as (x_c, y_c, w, h), where (x_c, y_c) represents the box center and (w, h) refers to the box scale. The control points $\left(x_j^{ctrl}, y_j^{ctrl}\right)$ of the cubic Bézier curve are expressed as

$$\left(x_j^{ctrl}, y_j^{ctrl}\right) = (x_c, y_c) + (\delta x_j \cdot w, \delta y_j \cdot h).$$

By multiplying $(\delta x_j, \delta y_j)$ with (w, h), the generated Bézier curve inherits the scale from the reference bounding box. Next, we uniformly sample points on this Bézier curve to obtain vertices of a polyline that fits the center-curve of text instance. This process is performed using the following equations:

$$p_{s,x} = \sum_{j=0}^3 x_j^{ctrl} B_j(t_s), \;\; p_{s,y} = \sum_{j=0}^3 y_j^{ctrl} B_j(t_s),$$

$$B_j(t_s) = \binom{3}{j} t_s^j (1-t_s)^{3-j}, \; t_s = \frac{s}{S-1},$$

where $s \in [0, \ldots, S-1]$ indexes each sample point, and $(p_{s,x}, p_{s,y})$ represents their coordinates. $B_j(t_s)$ calculates the value of each Berstein basis polynomial (Lorentz 2012) at every single step t_s. The polyline is complemented with a scalar width to form a boundary around a text instance. To that end an additional value δw_p is predicted, which describes the scaling factor between the bounding box size and the initial value of the text polygon's width:

$$w_p = \sqrt{wh} \cdot \exp(\delta w_p).$$

To obtain polygon vertices, we expand the polyline with distance $w_p/2$ towards outside using f_{buffer}:

$$p_s^{top}, p_s^{bot} = f_{buffer}(p_s, \frac{w_p}{2}).$$

The complete procedure is depicted in Figure 3 to enhance comprehension. To ensure a distinctive representation, the *top* vertices p_s^{top} are designated to reside on the left side while traversing the polyline in vertex order.

Iterative Regression of Proposal Polygons

A set of polygon proposals $N \times S \times 4$ are generated from last step after transforming N proposal boxes. Similar to the proposal boxes, their coordinates undergo updates across M layers of iterative regression. For instance, as depicted in Figure 2, during the initial polygon regression layer, given a proposal $\mathcal{P}_i^{m-1} = \left[\left(p_{s,y}^{top}, p_{s,y}^{bot}\right)^{m-1}\right]_{s=0}^{S-1}$ from the preceding layer, we first apply the following transformation:

$$(p_{s,x}, p_{s,y}, p_{s,dx}, p_{s,dy})^{m-1} =$$
$$\left(\frac{p_{s,x}^{top} + p_{s,x}^{bot}}{2}, \frac{p_{s,y}^{top} + p_{s,y}^{bot}}{2}, p_{s,x}^{top} - p_{s,x}^{bot}, p_{s,y}^{top} - p_{s,y}^{bot}\right)^{m-1}$$

This transformation converts the representation to center point coordinates and the coordinate differences between the top and bottom vertices. To ensure stability in the iterative regression process, the polygon regression head yields an output denoted as a scale- and location-invariant distance vector $\Delta_{poly} = [(\delta p_{s,x}, \delta p_{s,y}, \delta p_{s,dx}, \delta p_{s,dy})]_{s=0}^{S-1}$. Subsequently, the mth decoder refines the polygon as follows:

$$p_{s,x}^m = p_{s,x}^{m-1} + \delta p_{s,x} |p_{s,dx}^{m-1}|,$$
$$p_{s,y}^m = p_{s,y}^{m-1} + \delta p_{s,y} |p_{s,dy}^{m-1}|,$$
$$p_{s,dx}^m = p_{s,dx}^{m-1} \exp\left(\delta p_{s,dx}\right),$$
$$p_{s,dy}^m = p_{s,dy}^{m-1} \exp\left(\delta p_{s,dy}\right).$$

Gradients are soly back-propagated through distance vector Δ_{poly} to make the training stable, as in (Zhu et al. 2020; Sun et al. 2021). Afterwards, $(p_{s,x}, p_{s,y}, p_{s,dx}, p_{s,dy})^m$ are transformed back to the representation $\mathcal{P}_i^m = \left[\left(p_{s,y}^{top}, p_{s,y}^{bot}\right)^m\right]_{s=0}^{S-1}$ to align with the ground truth and compute the coordinate loss.

As $\exp\left(\delta p_{s,dx}\right)$ and $\exp\left(\delta p_{s,dy}\right)$ are invariably greater than 0, the sign of $(p_{s,dx}, p_{s,dy})^m$ remains consistent with $(p_{s,dx}, p_{s,dy})^{m-1}$. In simpler terms, this ensures that the updated polygon maintains a coherent alignment of its top and bottom boundaries across various regression layers.

PolyAlign

In pursuit of more representative Region of Interest (RoI) features for each polygon proposal, we introduce an innovative step called PolyAlign. In contrast to the conventional bounding box RoIAlign (He et al. 2017), which employs bilinear interpolation at the center of each RoI grid, PolyAlign directly operates on the paired top and bottom vertices of polygons and their corresponding centerline vertices. This approach is underpinned by the assumption that point-level coordinate regression is especially sensitive to deviations in the foothold of RoI feature extraction.

Upon extracting Poly RoI features $[\boldsymbol{z}^1, \boldsymbol{z}^2.., \boldsymbol{z}^L]$ from multi-layered feature maps, we aggregate them using sum aggregation $\boldsymbol{z}^{ms} = \sum_{l=1}^L \boldsymbol{z}^l$. Here, L refers to the number of multi-layer feature maps and $\boldsymbol{z}^{ms}$ represents the aggregated feature. The resultant Polygon RoI feature assumes spatial dimensions of $S \times 3 \times 256$, where S is the number of vertices on each boundary.
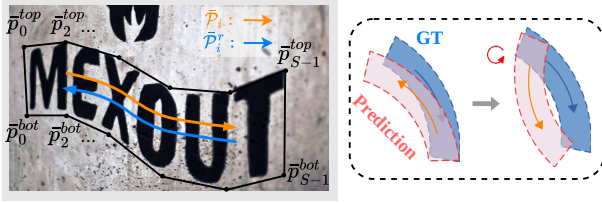
Figure 4: (left) The extended polygon annotations. $\mathcal{P}_i$ represents the original order, while $\mathcal{P}_i^r$ is the reversed one. (right) Learning orientation reversal under disparity.

Orientation-Equivalent Annotation

As the polygon representation, with pair-wise top and bottom vertices, always carries the attribute of orientation, an issue similar to the boundary discontinuity (Yang et al. 2021b,a, 2022) exists in its regression. As illustrated in Fig. 4 (right), if the polygon is initialized with reverse orientation w.r.t. the ground truth, the network needs extra effort to learn the reversal meanwhile correcting the vertex coordinates, which greatly complicates the learning process in the presence of diverse rotations. Inspired by (Liao et al. 2022), we remedy this issue by augmenting the ground truth with its orientation-reversed version. Given the ground truth as $\bar{\mathcal{P}}_i = \left[\left(\bar{p}_0^{top}, \bar{p}_0^{bot} \right), ..., \left(\bar{p}_{S-1}^{top}, \bar{p}_{S-1}^{bot} \right) \right]$ in Fig. 4 (left), the reversed version is then $\bar{\mathcal{P}}_i^r = \left[\left(\bar{p}_{S-1}^{bot}, \bar{p}_{S-1}^{top} \right), ..., \left(\bar{p}_0^{bot}, \bar{p}_0^{top} \right) \right]$. Given both orientations, a candidate polygon can, during training, be regressed to either $\bar{\mathcal{P}}_i$ or $\bar{\mathcal{P}}_i^r$.

Optimization

Polygon Bipartite Matching Like other detectors (Carion et al. 2020; Sun et al. 2021; Dai et al. 2021b; Zhu et al. 2020) that perform set prediction, we employ Bipartite Matching to assign predictions to ground truth. This cost consists of two components: correctness of class prediction and distance between coordinates. Given prediction set $Y = \{y_i\}_{i=0}^{N-1}$ and padded ground truth set $\bar{Y} = \{\bar{y}_i\}_{i=0}^{N-1}$. The matching cost is defined as

$$\mathcal{L}_{match}^{poly} \left(y_{\pi(i)}, \bar{y}_i \right) = \mathcal{L}_{focal} \left(\rho_{\pi(i)}, \bar{c}_i \right) + \mathcal{L}_{poly} \left(\mathcal{P}_{\pi(i)}, \bar{\mathcal{P}}_i^E \right).$$

Here, $\pi_{(i)} \in \Pi_N$ represents a permutation of N elements. $\mathcal{L}_{focal} \left(\rho_{\pi(i)}, \bar{c}_i \right)$ evaluates the class matching cost with Focal loss (Lin et al. 2017) between the predicted class score $\rho_{\pi(i)}$ and ground truth label $\bar{c}_i$, while $\mathcal{L}_{poly} \left(\mathcal{P}_{\pi(i)}, \bar{\mathcal{P}}_i^E \right)$ calculates the coordinate matching cost for polygon vertex coordinates by $L1$ distance between prediction $\mathcal{P}_{\pi(i)}$ and extended ground truth $\bar{\mathcal{P}}_i^E$. $\mathcal{L}_{poly} \left(\mathcal{P}_{\pi(i)}, \bar{\mathcal{P}}_i^r \right)$ is made to be orientation-invariant by taking the minimum one between two prediction-target pairs : $\mathcal{P}_{\pi(i)} \sim \bar{\mathcal{P}}_i$ and $\mathcal{P}_{\pi(i)} \sim \bar{\mathcal{P}}_i^r$. This is expressed as

$$\mathcal{L}_{poly} \left(\mathcal{P}_{\pi(i)}, \bar{\mathcal{P}}_i^E \right) = \mathcal{L}_{poly} \left(\mathcal{P}_{\pi(i)}, \tilde{\mathcal{P}}_i \right),$$
$$\tilde{\mathcal{P}}_i = \underset{\left[\bar{\mathcal{P}}_i, \bar{\mathcal{P}}_i^r \right]}{argmin} \left(\| \mathcal{P}_{\pi(i)} - \bar{\mathcal{P}}_i \|_1, \| \mathcal{P}_{\pi(i)} - \bar{\mathcal{P}}_i^r \|_1 \right).$$

Set Prediction Loss The training loss is identical to the matching cost, but summed only over the matched pairs. To encourage consistency, the ground truth for polygon coordinates directly inherits the orientation $\tilde{\mathcal{P}}_i$ from the matching cost calculation.

Experiments

Datasets

The datasets involved in the experiment are **SynthText 150K**, **TotalText**, **CTW1500**, **ICDAR19 MLT** and **InverseText**. As implied by the name, **SynthText 150K** (Liu et al. 2020) collects 150k synthesized scene text images, consisting of 94,723 images with multi-oriented straight texts and 54,327 images with curved ones. **TotalText** (Ch'ng and Chan 2017) contains 1,255 training images and 300 test images with highly diversified orientations and curvatures. In this dataset, the text instance is labeled in word-level. **CTW1500** is another dataset with curved texts including 1,000 training images and 500 test images. Its annotation is in text-line level. **ICDAR19 MLT** is a multi-lingual scene text detection dataset (Nayef et al. 2019) providing 10k images for training. **InverseText** (Ye et al. 2023a) is a test set that consists of 500 test images with text instances concurrently possessing rotation and curvature.

Implementation Details

ResNet50 (He et al. 2016) is adopted for all experiments and initialized with ImageNet pretrained weights. The batch size is set 16 and all models are trained with 8 pieces NVIDIA RTX 3090 GPUs. The final results on TotalText and CTW1500 are reported with the training strategy similar to (Zhang et al. 2022; Ye et al. 2023a): First, the network is pretrained on a combined dataset for 180k iterations with a learning rate 2.5×10^{-5} that drops at 144k and 162k step. The learning rate drop factor is 10. The combined dataset is composed of TotalText, ICDAR19 MLT and SynthText 150K. Following (Ye et al. 2023a), data augmentation includes instance aware random cropping, random blur, color jittering and multi-scale resizing, with the shorter side restricted to the range $[480, \ldots, 832]$ pixels and the longer side capped at a maximum of 1600 pixels.

Before training, the datasets are polished following the protocol of (Ye et al. 2023a) – annotation of polygon vertices is rearranged to eliminate the influence of the implicit reading order, and the training set is expanded by rotating training images with a set of predefined rotation angles. Our chosen proposal number, denoted as N, is set at 300. Meanwhile, for Bezier curves, the number of sampling points S has been established as 8, yielding 16 vertices for each polygon proposal. The box head employs a multi-stage approach with K layers set at 3, while the polygon head similarly employs a multi-stage structure with M layers also set at 3.

Comparisons With Other Methods

On Arbitrary-Shaped Text Benchmarks We evaluate our network on two widely-used text detection benchmarks, namely TotalText and CTW1500. Despite using just one feature embedding per polygon instance, our approach

Method	Feature Extractor	TotalText			CTW1500		
		P	R	F	P	R	F
DB(Liao et al. 2020)	ResNet50+DCN	87.1	82.5	84.7	86.9	80.2	83.4
I3CL(Du et al. 2022)	ResNet50+FPN	89.2	83.7	86.3	87.4	84.5	85.9
ABCNet-v2(Liu et al. 2021)	ResNet50+FPN	90.2	84.1	87.0	85.6	83.8	84.7
Boundary(end-to-end)(Wang et al. 2020)	ResNet50+FPN	88.9	85.0	87.0	-	-	-
TESTR-Polygon(Zhang et al. 2022)	ResNet50+Deform. Encoder	93.4	81.4	86.9	92.0	82.6	87.1
SwinTextSpotter(Huang et al. 2022)	Swin-Transformer+FPN	-	-	88.0	-	-	88.0
DPText-DETR(Ye et al. 2023a)	ResNet50+Deform. Encoder	91.8	86.4	**89**	91.7	86.2	**88.8**
Box2Poly(ours)	ResNet50+FPN	90.22	86.57	<u>88.35</u>	88.84	87.46	<u>88.13</u>

Table 1: Quantitative text detection results on arbitrarily shaped datasets, measured using Precision (P), Recall (R), and F-score (F). Our method's results are presented as *mean* achieved through five rounds of finetuning, the spread of F-score is conveyed as 3σ *empirical standard deviation*, which are correspondingly ±0.26 and ±0.35 for TotalText and CTW1500.

Figure 5: Qualitative results on images containing text instances rotated and randomly curved.

achieves comparable performance to the current state-of-the-art method, DPText-DETR. The latter utilizes point-level feature embedding on each polygon vertex (16 vertices per polygon). In comparison, our method's F-Score is only 0.7% lower on both dataset: 88.35 vs. 89 on TotalText and 88.13 vs. 88.8 on CTW1500 (Tab. 1). Moreover, it is noteworthy that DPText-DETR employs a stronger feature extracting method compared to our network, and it takes advantage of proposal generator. Box2Poly leads TESTR-Polygon by 1.7% and 1.2% on these two benchmarks (Tab. 1). Additionally, when compared to SwinTextSpotter, which employs a stronger backbone, our detection performance overhead on TotalText and CTW1500 is 0.35 and 0.13 regarding F-score, respectively (Tab. 1).

Stability Against Rotation Together With Arbitrary Shape To showcase the robustness of our network in simultaneously managing text rotation and arbitrary curvature, we conduct a comparative analysis of detection results with other polygon-based methods on InverseText. As delineated in Tab. 3, our network achieves marginally improved performance over DPText-DETR under the setting of no pretraining. With the inclusion of pretraining, Box2Poly achieves an F-score of 88.7, surpassing DPText-DETR by 1.6%. Further substantiating our network's capabilities, we

visually present a selection of detection results in Fig. 5. Notably, the network consistently generates compact polygons for images featuring text instances with varying degrees of rotation and curvature.

Memory Efficiency and Data Efficiency As depicted in Tab. 4, the incorporation of the *single feature embedding per instance* approach notably reduces the GPU memory footprint of our network compared to DPText-DETR, despite our model leveraging three times as many instance proposals (300 vs. 100). And a faster inference speed is observed in our method (FPS 19.6 vs. 13.6). Remarkably, after solely 10k training iterations on the TotalText dataset, Box2Poly achieves performance on par with no-pretraining DPText-DETR. This result underscores Box2Poly's commendable data efficiency(Tab. 4).

Ablation Studies

In this section, we undertake an analysis of the distinct contributions made by various components introduced in Sec. . Our experimental approach involves systematically deactivating or substituting different components of Box2Poly during each experiment. The model is trained for 50k iterations on the TotalText dataset, with subsequent evaluation conducted on InverseText. The initial learning rate is set at 2.5×10^{-5} and is reduced to 2.5×10^{-6} at 80% of the iterations. The confidence score threshold is established at 0.3. Comprehensive results are detailed in Tab. 2.

Transformation From Box to Polygon In Experiment 1, as outlined in Tab. 2, we employ a strategy akin to DPText-DETR (Ye et al. 2023a) and TESTR (Zhang et al. 2022) for the conversion from bounding box to polygon. This involves sampling a consistent number of points along the upper and lower edges of the bounding box, ensuring equidistant distribution. It's performance is significantly lower compared to our *Box to Poly* method discussed in Sec. . This verifies the effectiveness of our design conducting transformation.

PolyAlign In order to validate the assumption introduced in Sec. , we conduct an evaluation of the impact of different PolyAlign implementations: Vertex, Grid and Bézier. Among these, PolyAlign-Vertex is the implementation integrated into our proposed network. This variant projects ver-

Experiments	Trans. Box to Poly		PolyAlign			Orientation-Equivalent Annotation	Polygon IoU Loss	F-score	
	Sampling	Box+bezier	Vertex	Grid	Bezier			50K	30K
1.	✓	-	✓	-	-	✓	-	53.61	-
2.	-	✓	-	✓	-	✓	-	86.07	-
3.	-	✓	-	-	✓	✓	-	86.26	-
4.	-	✓	✓	-	-	-	-	**87.49**	86.94
5.	-	✓	✓	-	-	✓	✓	86.29	-
6.(final)	-	✓	✓	-	-	✓	-	87.41	**87.34**

Table 2: Ablation studies on InverseText. 50K and 30K corresponds to the number of training iterations. "final" corresponds to the version of model we leverage to derive the final result. ✓ marks the integrated component or technique.

Method	P	R	F
TESTR-Polygon	91.9	84.4	86.8
DPText-DETR	90.7	84.2	87.3
Box2Poly(ours)	91.4	86.1	**88.7**
DPText-DETR†	-	-	86.8
Box2Poly†(ours)	90.4	85.7	**87.41**

Table 3: Evaluation of methods on the InverseText dataset. †means pretraining is not included. The results of the other two methods are fetched from DPText-DETR.

Method	Total-Text			FPS	VRAM
	P	R	F		
DPText-DETR	-	-	86.79	13.6	17409MB
Box2Poly(ours)	89.58	83.83	86.60	**19.6**	**8072MB**

Table 4: Qualitative results on TotalText without pretrainig. VRAM occupation is measured with 2 images per GPU.

tex coordinates back onto the feature maps and performs RoI feature extraction from there. On the other hand, PolyAlign-Grid aligns its feature extraction foothold with the center of sampling grids within the polygon, following the design principles of RoI Align (He et al. 2017). The Bézier implementation, PolyAlign-Bézier, is based on BezierAlign from ABCNet (Liu et al. 2020), albeit with several modifications made to ensure uniformly distributed sampling grids. To cope with this feature extractor, the regression is conducted on Bézier-Polygon, where top and bottom boundary is represented with Bézier curves. As indicated in Tab. 2, it is observed that PolyAlign-Vertex achieves the best performance. This result lends support to our assumption that precise feature extraction at specific locations is advantageous for vertex regression. This finding is congruent with the notion that DPText-DETR outperforms TESTR-Polygon due to the former using vertex coordinates as reference points to conduct deformable cross-attention (Zhu et al. 2020), rather than the center of proposal bounding box.

Orientation-Equivalent Annotation As demonstrated in the results of $50K$ training iterations presented in Tab. 2, the incorporation of *OEA* (*Orientation-Equivalent Annotation*) does not yield a performance improvement for the network. However, upon integrating *OEA*, a better convergence behavior is observed after re-running Exp. 4 and 7 with identical configuration for less iterations($30K$) (Tab. 2). As a result, the disruption in learning caused by orientation discrepancy has been notably mitigated. The relatively modest impact of *OEA* on performance can be attributed to the fact that, within the polished datasets, text orientation issue has been mitigated to a certain extent.

Polygon IoU Loss Since its inception, the Intersection over Union (IoU) loss (Yu et al. 2016) has emerged as an indispensable technique in the realm of bounding box object detection. Therefore, we are also inclined to devise a similar loss to aid the task of polygon prediction. Recognizing that formulating the intersection between two non-convex polygons in a differentiable manner is a challenging endeavor, we address this issue by decomposing the polygon into a set of quadrilaterals. The IoU loss is then calculated at this level. Notably, this polygon IoU loss does not yield any improvement, and even leads to a performance drop (as evidenced by Exp. 5 in Tab. 2). We assume the reason behind this is, unlike bounding box, IoU loss is redundant when vertex-level $L1$ loss is introduced, and this estimated IoU cannot reflect the overlapping quality between non-convex polygons well.

Conclusion

In this paper, we present a novel approach for text detection by performing iterative polygon regression. Our method modifies the box iterative regression pipeline to make it suit the polygon regression task. It begins with a learnable bounding box and incorporates a transformative process, allowing for seamless transition from box to polygon representation. Then, the polygon coordinates are iteratively refined using polygon RoI(Region of Interest) features. Through this process, our network demonstrates remarkable memory efficiency and stability in detecting text of varying and intricate shapes. Despite a slight reduction in performance compared to the current state-of-the-art (SOTA), our approach still maintains commendable results. It's worth noting that our detector may face challenges in scenarios where small texts dominate the scene, difficult to guarantee high recall. To address this, the incorporation of a reliable proposal generator could be beneficial. Additionally, any resulting increase in memory occupation due to this augmentation can be probably counterbalanced by reducing the number of box and polygon regression layers.

References

Baek, Y.; Lee, B.; Han, D.; Yun, S.; and Lee, H. 2019. Character region awareness for text detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9365–9374.

Bi, Y.; and Hu, Z. 2021. Disentangled contour learning for quadrilateral text detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 909–918.

Cai, Z.; and Vasconcelos, N. 2018. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6154–6162.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, 213–229. Springer.

Ch'ng, C. K.; and Chan, C. S. 2017. Total-text: A comprehensive dataset for scene text detection and recognition. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, 935–942. IEEE.

Dai, P.; Zhang, S.; Zhang, H.; and Cao, X. 2021a. Progressive contour regression for arbitrary-shape scene text detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 7393–7402.

Dai, X.; Chen, Y.; Yang, J.; Zhang, P.; Yuan, L.; and Zhang, L. 2021b. Dynamic detr: End-to-end object detection with dynamic attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2988–2997.

Du, B.; Ye, J.; Zhang, J.; Liu, J.; and Tao, D. 2022. I3cl: Intra-and inter-instance collaborative learning for arbitrary-shaped scene text detection. *International Journal of Computer Vision*, 130(8): 1961–1977.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

He, M.; Liao, M.; Yang, Z.; Zhong, H.; Tang, J.; Cheng, W.; Yao, C.; Wang, Y.; and Bai, X. 2021. MOST: A multi-oriented scene text detector with localization refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8813–8822.

Hong, Z.; Petillot, Y.; Lane, D.; Miao, Y.; and Wang, S. 2019. TextPlace: Visual place recognition and topological localization through reading scene texts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2861–2870.

Huang, M.; Liu, Y.; Peng, Z.; Liu, C.; Lin, D.; Zhu, S.; Yuan, N.; Ding, K.; and Jin, L. 2022. Swintextspotter: Scene text spotting via better synergy between text detection and text recognition. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4593–4603.

Li, B.; Zou, D.; Sartori, D.; Pei, L.; and Yu, W. 2020. Textslam: Visual slam with planar text features. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2102–2108. IEEE.

Liao, B.; Chen, S.; Wang, X.; Cheng, T.; Zhang, Q.; Liu, W.; and Huang, C. 2022. MapTR: Structured Modeling and Learning for Online Vectorized HD Map Construction. *arXiv preprint arXiv:2208.14437*.

Liao, M.; Wan, Z.; Yao, C.; Chen, K.; and Bai, X. 2020. Real-time scene text detection with differentiable binarization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 11474–11481.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2980–2988.

Liu, Y.; Chen, H.; Shen, C.; He, T.; Jin, L.; and Wang, L. 2020. Abcnet: Real-time scene text spotting with adaptive bezier-curve network. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9809–9818.

Liu, Y.; and Jin, L. 2017. Deep matching prior network: Toward tighter multi-oriented text detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1962–1969.

Liu, Y.; Shen, C.; Jin, L.; He, T.; Chen, P.; Liu, C.; and Chen, H. 2021. Abcnet v2: Adaptive bezier-curve network for real-time end-to-end text spotting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11): 8048–8064.

Long, S.; Ruan, J.; Zhang, W.; He, X.; Wu, W.; and Yao, C. 2018. Textsnake: A flexible representation for detecting text of arbitrary shapes. In *Proceedings of the European conference on computer vision (ECCV)*, 20–36.

Lorentz, G. G. 2012. *Bernstein polynomials*. American Mathematical Soc.

Lyu, P.; Liao, M.; Yao, C.; Wu, W.; and Bai, X. 2018. Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. In *Proceedings of the European conference on computer vision (ECCV)*, 67–83.

Ma, J.; Shao, W.; Ye, H.; Wang, L.; Wang, H.; Zheng, Y.; and Xue, X. 2018. Arbitrary-oriented scene text detection via rotation proposals. *IEEE transactions on multimedia*, 20(11): 3111–3122.

Nayef, N.; Patel, Y.; Busta, M.; Chowdhury, P. N.; Karatzas, D.; Khlif, W.; Matas, J.; Pal, U.; Burie, J.-C.; Liu, C.-l.; et al. 2019. ICDAR2019 robust reading challenge on multilingual scene text detection and recognition—RRC-MLT-2019. In *2019 International conference on document analysis and recognition (ICDAR)*, 1582–1587. IEEE.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Rong, X.; Li, B.; Munoz, J. P.; Xiao, J.; Arditi, A.; and Tian, Y. 2016. Guided text spotting for assistive blind navigation in unfamiliar indoor environments. In *Advances in Visual*

Computing: 12th International Symposium, ISVC 2016, Las Vegas, NV, USA, December 12-14, 2016, Proceedings, Part II 12, 11–22. Springer.

Sun, L.; and Liu, K. 2022. Center TextSpotter: A novel text spotter for autonomous unmanned vehicles. *IEEE Transactions on Intelligent Transportation Systems*.

Sun, P.; Zhang, R.; Jiang, Y.; Kong, T.; Xu, C.; Zhan, W.; Tomizuka, M.; Li, L.; Yuan, Z.; Wang, C.; et al. 2021. Sparse r-cnn: End-to-end object detection with learnable proposals. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 14454–14463.

Wang, H.; Lu, P.; Zhang, H.; Yang, M.; Bai, X.; Xu, Y.; He, M.; Wang, Y.; and Liu, W. 2020. All you need is boundary: Toward arbitrary-shaped text spotting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 12160–12167.

Wang, W.; Xie, E.; Song, X.; Zang, Y.; Wang, W.; Lu, T.; Yu, G.; and Shen, C. 2019. Efficient and accurate arbitrary-shaped text detection with pixel aggregation network. In *Proceedings of the IEEE/CVF international conference on computer vision*, 8440–8449.

Xu, Y.; Wang, Y.; Zhou, W.; Wang, Y.; Yang, Z.; and Bai, X. 2019. Textfield: Learning a deep direction field for irregular scene text detection. *IEEE Transactions on Image Processing*, 28(11): 5566–5579.

Yang, X.; Yan, J.; Ming, Q.; Wang, W.; Zhang, X.; and Tian, Q. 2021a. Rethinking rotated object detection with gaussian wasserstein distance loss. In *International conference on machine learning*, 11830–11841. PMLR.

Yang, X.; Yang, X.; Yang, J.; Ming, Q.; Wang, W.; Tian, Q.; and Yan, J. 2021b. Learning high-precision bounding box for rotated object detection via kullback-leibler divergence. *Advances in Neural Information Processing Systems*, 34: 18381–18394.

Yang, X.; Zhou, Y.; Zhang, G.; Yang, J.; Wang, W.; Yan, J.; Zhang, X.; and Tian, Q. 2022. The KFIoU loss for rotated object detection. *arXiv preprint arXiv:2201.12558*.

Ye, M.; Zhang, J.; Zhao, S.; Liu, J.; Du, B.; and Tao, D. 2023a. Dptext-detr: Towards better scene text detection with dynamic points in transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 3241–3249.

Ye, M.; Zhang, J.; Zhao, S.; Liu, J.; Liu, T.; Du, B.; and Tao, D. 2023b. Deepsolo: Let transformer decoder with explicit points solo for text spotting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19348–19357.

Yu, J.; Jiang, Y.; Wang, Z.; Cao, Z.; and Huang, T. 2016. Unitbox: An advanced object detection network. In *Proceedings of the 24th ACM international conference on Multimedia*, 516–520.

Zhang, C.; Liang, B.; Huang, Z.; En, M.; Han, J.; Ding, E.; and Ding, X. 2019. Look more than once: An accurate detector for text of arbitrary shapes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10552–10561.

Zhang, S.-X.; Yang, C.; Zhu, X.; and Yin, X.-C. 2023. Arbitrary shape text detection via boundary transformer. *IEEE Transactions on Multimedia*.

Zhang, X.; Su, Y.; Tripathi, S.; and Tu, Z. 2022. Text spotting transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9519–9528.

Zhang, Z.; Zhang, C.; Shen, W.; Yao, C.; Liu, W.; and Bai, X. 2016. Multi-oriented text detection with fully convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4159–4167.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2020. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*.

Zhu, Y.; Liao, M.; Yang, M.; and Liu, W. 2017. Cascaded segmentation-detection networks for text-based traffic sign detection. *IEEE transactions on intelligent transportation systems*, 19(1): 209–219.