Hierarchical Text Spotter for Joint Text Spotting and Layout Analysis

Shangbang Long, Siyang Qin, Yasuhisa Fujii, Alessandro Bissacco, Michalis Raptis
Google Research
{longshangbang,qinb,yasuhisaf,bissacco,mraptis}@google.com

Abstract

*We propose **Hierarchical Text Spotter (HTS)**, a novel method for the joint task of word-level text spotting and geometric layout analysis. HTS can recognize text in an image and identify its 4-level hierarchical structure: characters, words, lines, and paragraphs. The proposed HTS is characterized by two novel components: (1) a **Unified-Detector-Polygon (UDP)** that produces Bezier Curve polygons of text lines and an affinity matrix for paragraph grouping between detected lines; (2) a **Line-to-Character-to-Word (L2C2W)** recognizer that splits lines into characters and further merges them back into words. HTS achieves state-of-the-art results on multiple word-level text spotting benchmark datasets as well as geometric layout analysis tasks.*

1. Introduction

The extraction and comprehension of text in images play a critical role in many computer vision applications. Text spotting algorithms have progressed significantly in recent years [33, 42, 45, 49, 67], specifically within the task of detecting [2, 28, 36, 63] and recognizing [5, 12, 40, 41, 59] individual text instances in images. Previously, defining the geometric layout [7, 9, 24, 62] of extracted textual content occurred independent of text spotting and remained focused on document images. In this paper, we aim to further the argument [34] that consolidating these separately treated tasks is complementary and mutually enhancing. We postulate a joint approach for text spotting and geometric layout analysis could provide useful signals for downstream tasks such as semantic parsing and reasoning of text in images such as text-based VQA [6, 53] and document understanding [16, 23, 25].

Existing text spotting methods [45, 49, 67] most commonly extract text at the word level, where 'word' is defined as a sequence of characters delimited by space without taking into account the text context. Recently, the Unified Detector [34], which is built upon detection transformer [58], detects text 'lines' with instance segmentation mask and produces an affinity matrix for paragraph grouping in an
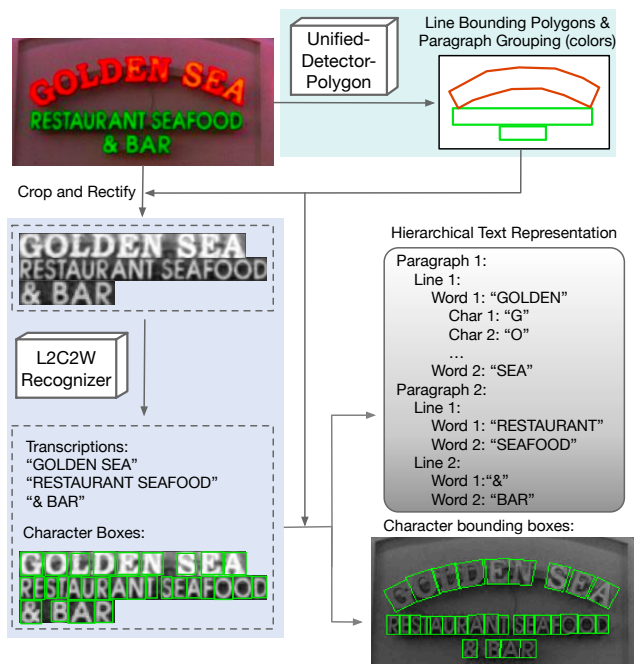

Figure 1. Illustration of our **Hierarchical Text Spotter (HTS)**. HTS consists of two main components: (1) **Unified-Detector-Polygon (UDP)** that detects text lines with bounding polygons and clusters them into paragraph groups. In this figure *(upper right)*, paragraph groups are illustrated by different colors. The bounding polygons are used to crop and rectify text lines into canonical forms that are easy to recognize. (2) **Line-to-Character-to-Word (L2C2W)** Recognizer that jointly predicts character classes and bounding boxes. Spaces are used to split lines into words. The output of HTS is a **Hierarchical Text Representation (HTR)** that encodes the layout of all text entities in an image. In this figure, we use indents to represent the hierarchy of text entities *(middle right)*, and visualize the character bounding boxes *(bottom right)*.

end-to-end way. This method is limited to the detection task and it can not produce character or word-level outputs.

In this paper, we propose a novel method, termed *Hierarchical Text Spotter (HTS)*, that simultaneously localize, recognize and recovers the geometric relationship of the text on an image. The framework of HTS is illustrated in Fig. 1. It is designed to extract a hierarchical text representa-

tion (HTR) of text entities in images. HTR has four levels of hierarchy[1], including *character*, *word*, *text line*, and *paragraph*, from bottom to top. The HTR representation encodes the structure of text in images. To the best of our knowledge, HTS is the first unified method for text spotting and geometric layout analysis.

The proposed HTS consists of two main components: (1) A *Unified-Detector-Polygon (UDP)* model that jointly predicts Bezier Curve polygons [30] for text lines and an affinity matrix supporting the grouping of lines to paragraphs. Notably, we find that the conventional way of training Bezier Curve polygon prediction head, i.e. applying $L1$ losses on control points directly [30, 47, 56], fails to capture text shapes accurately on highly diverse dataset such as HierText [34]. Hence, we propose a novel *Location and Shape Decoupling Module (LSDM)* which decouples the representation learning of location and shape. UDP equipped with LSDM can accurately detect text lines of arbitrary shapes, sizes and locations across multiple datasets of different domains. (2) A *Line-to-Character-to-Word (L2C2W)* text line recognizer based on Transformer encoder-decoder [57] that jointly predicts character bounding boxes and character classes. L2C2W is trained to produce the special space character to delimit text lines into words. Also, unlike other recognizers or text spotters that are based on character detection [3, 27, 31, 61], L2C2W only needs a small fraction of training data to have bounding box annotations.

The proposed HTS method achieves state-of-the-art text spotting results on multiple datasets across different domains, including ICDAR 2015 [19], Total-Text [10], and HierText [34]. It also surpasses Unified Detector [34] on the geometric layout analysis benchmark of HierText, achieving new state-of-the-art result. Importantly, these results are obtained with a single model, without fine-tuning on target datasets; ensuring that the proposed method can support generic text extraction applications. In ablation studies, we also examine our key design choices.

Our core contributions can be summarized as follows:

- A novel Hierarchical Text Spotter for the joint task of word-level text spotting and geometric layout analysis.
- Location and Shape Decoupling Module which enables accurate polygon prediction of text lines on diverse datasets.
- L2C2W that reformulates the role of recognizer in text spotter algorithms by performing part of layout analysis and text entities localization.
- State-of-the-art results on both text spotting and geometric layout analysis benchmarks without fine-tuning to each particular test dataset.

---

[1]Here, we follow the definitions of these levels in [34].

2. Related Works

Text Spotting Two-stage text spotters consist of a text detection stage and a text recognition stage. Text detection stage produces bounding polygons or rotated bounding boxes for text instances at one granularity, usually words. Text instances are cropped from input image pixels [4], encoded backbone features [26, 45], or both [49]. The text recognition stage decodes the text transcription. End-to-end text spotters use feature maps for the cropping process. In this case, the text recognition stage reuses those features, improving the computational efficiency [29]. However, end-to-end text spotters suffer from asynchronous convergence between the detection and the recognition branch [22]. Due to this challenge, our proposed HTS crops from input image pixels with bounding polygons. The aforementioned text spotter framework connects detection and recognition explicitly with detection boxes. Another branch of two-stage text spotter performs implicit feature feeding via object queries [21] as in detection transformer [8] or deformable multi-head attention [67]. More recently, single stage text spotters [20, 42] are proposed under a sequence-to-sequence framework. These works do not perform layout analysis and are thus orthogonal to this paper.

Text Detection Top-down text detection methods view text instances as objects. These methods produce detection boxes [30, 56] or instance segmentation masks [34] for each text instance. Bottom-up methods first detect sub-parts of text instances and then connect these parts to construct whole-text bounding boxes [52] or masks [36]. Top-down methods tend to have simpler pipelines, while bottom-up techniques excel at detecting text of arbitrary shapes and aspect ratios. Neither top-down nor bottom-up mask prediction methods are proficient for spotting curved text, because a mask can only locate text but cannot rectify it. Additionally, the performance of such models on curved text datasets is commonly reported by fine-tuning those models on the specific data. Therefore, it is unknown whether polygon prediction methods can adapt to text of arbitrary shapes and aspect ratios on diverse datasets.

Text Recognition An important branch of text recognizers [12, 32, 40] formulates the task as a sequence-to-sequence task [55], where the only output target is a sequence of characters. Another branch formulates the task as character detection [27, 31], where it produces character classes and locations simultaneously. However, it requires bounding box annotations on all training data, which are rare for real-image data. Our recognition method falls into the sequence-to-sequence learning paradigm, with the additional ability to produce each character bounding box. Importantly, our model's training requires only partially annotated data, i.e. only a fraction of the data needs to include character level bounding box annotations.

Layout analysis Geometric layout analysis [18, 43, 60, 68]

aims to detect visually and geometrically coherent text blocks as objects. Recent works formulate this task as object detection [51], semantic segmentation [24, 34], or learning on the graphical structure of OCR tokens via GCN [60]. Almost all entries in the HierText competition at IDCAR 2023 [35] adopt the segmentation formulation. Unified Detector [34] consolidates the task of text line detection and geometric layout analysis. However, it can not produce word-level entities and does not provide a recognition output. Another line of layout analysis research focuses on semantic parsing of documents [16, 23, 25] to identify key-value pairs. These methods build language models [13, 46] on top of OCR results. Recently, StruturalLM [25] and LayoutLMv3 [16] show that the grouping of words into *segments* using heuristics, which is equivalent to text line formation, improves parsing results. We believe our work of jointly text spotting and geometric layout analysis can benefit semantic parsing and layout analysis.

3. Methodology

3.1. Hierarchical Text Spotter

As illustrated in Fig. 1 our HTS method mainly comprises 2 stages: (1) *Unified Detection Stage*: we propose an end-to-end trainable model termed Unified-Detector-Polygon (UDP) that detects text lines in the form on Bezier Curve Polygons [30], and simultaneously clusters them into paragraphs. UDP contains the Location and Shape Decoupling Module (LSDM), a key component in accurate text line detection across diverse datasets. Text line images are cropped from the input image with BezierAlign [30] and then converted to grayscale image patches. (2) *Line Recognition Stage*: We propose an autoregressive text line recognizer based on Transformer encoder-decoder [57] that jointly predicts character bounding boxes and character classes. We train our recognizer to identify printable characters and a special non-printable *space* delimiter. We use the *space* character to split text lines into word-level granularity. The word-level bounding boxes are formed from the predicted character-level bounding boxes. Character and word bounding boxes are estimated in the coordinate space of text line image patches. During the post-processing step, they are projected back to the input image coordinate space. Putting these together, we obtain a hierarchical text representation of *character*, *word*, *line*, and *paragraph*.

3.2. Unified Detection of Text Line and Paragraph

Preliminaries Based on MaX-DeepLab [58], Unified-Detector [34] detects text lines by producing instance segmentation masks from the inner product of object queries and pixel features. Further, an affinity matrix that represents the paragraph grouping is produced by computing the inner product of layout features which are extracted by extra
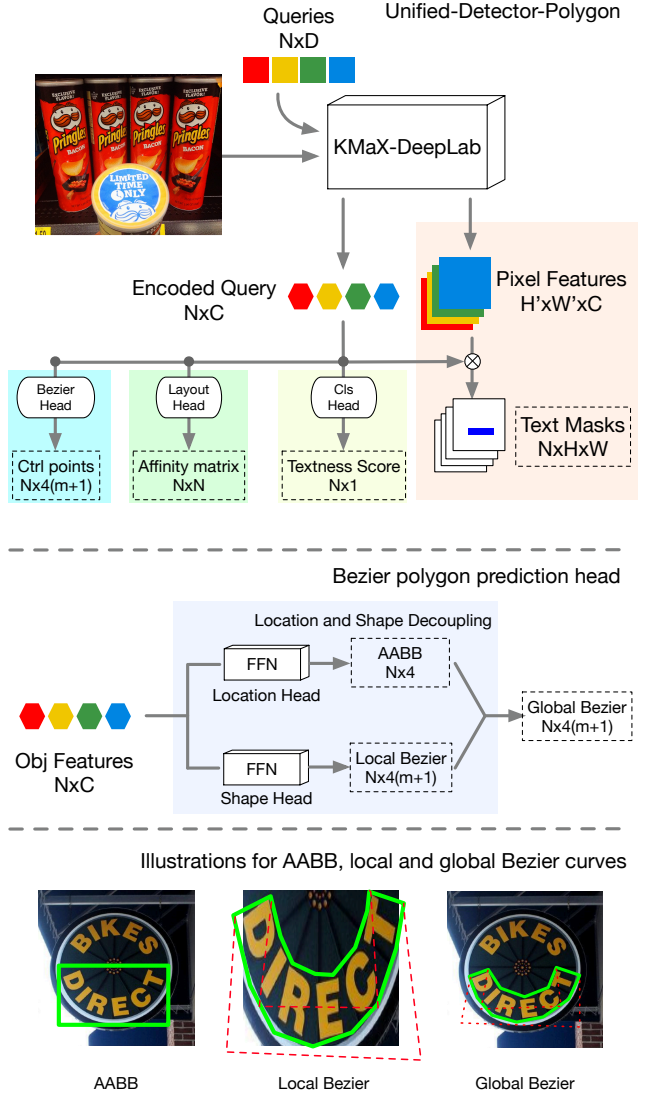

Figure 2. Illustration of our **Unified-Detector-Polygon (UDP)**. **Top**: Architecture of UDP, where each color tint represents one prediction branch. N is the number of queries. m is the order of the Bezier Curves. C is the model width. D is the query dimension. **Middle**: Architecture of our Bezier polygon prediction head with a dual-head Location and Shape Decoupling Module. **Bottom**: Illustrations for axis-aligned bounding box (AABB), local and global Bezier curve representation.

transformer layers applied on object queries.

Unified-Detector-Polygon (UDP) While Unified Detector [34] achieves state-of-the-art text detection performance, it uses only masks to localize text instances. The estimated masks can not be directly used to rectify curved text lines. Thus, complex post-processing heuristics are required to build an effective text-spotting system. Therefore, we extend the model with an additional Bezier polygon prediction

head applied on the encoded object queries, as illustrated in the top of Fig. 2. The Bezier polygon prediction head produces a polygon representation [30] based on Bezier Curve[2]. In this representation, each text line is parametrized as two Bezier Curves of order m, one for the top and one for the bottom polyline of the text boundary. Each Bezier Curve has $m + 1$ control points. The model is trained to predict these $2(m + 1)$ control points i.e. $4(m + 1)$ coordinates. During inference, the text boundaries are reconstructed from the predicted control points. In addition, we also replace MaX-DeepLab in the original Unified Detector with KMaX-DeepLab [65] as the backbone, which is faster and more accurate.

Location and Shape Decoupling Module Previous works [47, 56] use a single feed-forward neural network (FFN) to predict the control points in image space and train the network by applying L1 loss on the control points. However, as shown in Sec. 4.4, such approach results to sub-optimal detection accuracy for text line datasets such as HierText [34] due to its diverse locations, aspect ratios, and shapes. To mitigate this issue, we propose a novel *Location and Shape Decoupling Module (LSDM)*. As shown in the middle of Fig. 2, it consists of two parallel FFNs, one for location prediction and the other for shape prediction. The Location Head predicts Axis-Aligned Bounding Boxes (AABB) whose coordinates are normalized in the image space. For the $i-$th text instance, we denote its predicted AABB as:

$$AABB_i = [x_{\text{center},i}, y_{\text{center},i}, w_i, h_i] \in R^4 \quad (1)$$

representing its center, width, and height. The Shape Head predicts Local Bezier Curve control points whose coordinates are normalized in the space of the AABB:

$$bezier_{local,i} = \{(\tilde{x}_{i,j}, \tilde{y}_{i,j})\}_{j=1}^{2(m+1)} \quad (2)$$

Finally, the Global, i.e. image space, Bezier curve control point coordinates are obtained by scaling and translating Local Bezier coordinates by AABB:

$$bezier_{global,i} = \{(x_{i,j}, y_{i,j})\}_{j=1}^{2(m+1)} \quad (3)$$

$$\text{where } x_{i,j} = \tilde{x}_{i,j} * w_i + x_{\text{center},i} \quad (4)$$

$$y_{i,j} = \tilde{y}_{i,j} * h_i + y_{\text{center},i} \quad (5)$$

The concepts of AABB, Local Bezier coordinates, and Global Bezier coordinates are further illustrated in the bottom of Fig. 2. During training, we generate appropriately ground-truth data for both heads and apply supervision on both of them. Specifically, given ground-truth Global Bezier control points, we first compute ground-truth AABB as the minimum area AABB enclosing the ground-truth polygons, and then use the reverse of Eq. (3) (4) (5) to compute ground-truth Local Bezier control points. The

final training loss is the weighted sum of all Unified Detector [34] loss, GIoU loss on AABB [48], L1 loss on AABB, and L1 loss on local control points:

$$L_{det} = L_{\text{unified detector}}$$
$$+ \lambda_1 L_{\text{AABB,GIoU}} + \lambda_2 L_{\text{AABB},L_1} + \lambda_3 L_{\text{Local},L_1} \quad (6)$$

where $\lambda_1, \lambda_2, \lambda_3$ are the weights for loss balancing.

3.3. Line-to-Character-to-Word Recognition

We propose a novel hierarchical text recognition framework, termed *Line-to-Character-to-Word (L2C2W)*. Fig. 3 illustrated our framework. Text line images are cropped and rectified from the input image with BezierAlign [30]. We use the grayscale cropped image as input for the recognizer. The model predicts character-level outputs. To correctly group characters into words, our recognition model learns to predict both printable characters and the *space* character. During inference, we use the space as the delimiter to segment a text line string into words. The model also produces character-level bounding boxes. These character bounding boxes are grouped based on each word's boundaries and produce the words' bounding boxes.

Text Line Recognition Model Our transformer-based recognizer consists of three stages. First, a MobileNetV2 [50] convolutional backbone encodes the image pixels, and reduces the height dimension to 1 using strided convolutions. Then, a sinusoidal positional encoding [57] is added, and transformer encoder layers are applied on the encoded features. Lastly, a transformer decoder produces the predicted output autoregressively [55].

Character Localization We use axis-aligned bounding boxes to represent the location of characters in cropped text lines. Vanilla transformer decoder [57] has only one prediction head to produce a probability distribution over the next token. To predict character bounding boxes, we add a 2-layer FFN prediction head on the output feature from decoder, in parallel to the classification head. The character location head produces a 4d vector representing the top-left and bottom-right coordinates of the character bounding boxes. These character coordinates are normalized by each text line's height.

Training The total loss for training is the weighted sum of character classification loss and character localization loss. We use cross-entropy for character classification and L_1 loss for character localization. It is important to note that ground-truth annotated character bounding boxes are rare in real-image datasets but are available in most synthetic text data [14, 37, 64]. During training, we mix real-image and synthetic data and apply character localization loss only when ground-truth labels are available. The training target for one text line can be formulated as:

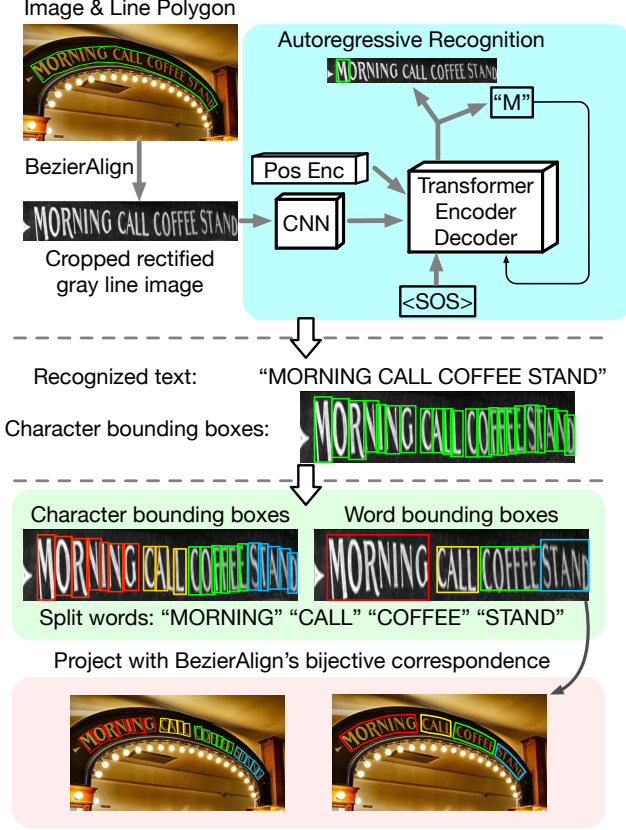[2]https://en.wikipedia.org/wiki/Bezier_curve

Figure 3. Illustration of our **Line-to-Character-to-Word (L2C2W)** recognition method. **Top**: Text line images are cropped and rectified from the input image using BezierAlign [30]. Our L2C2W recognition model uses an autoregressive transformer encoder-decoder model [57] to predict character class and box simultaneously. **Middle**: Output sample. **Bottom**: Text line recognition results are split into words, and character bounding boxes are clustered in accordance with words and form word bounding boxes. Bounding boxes are projected back to image space.

$$L_{rec} = \frac{1}{T}\sum_{t=1}^{T} L_{CE}(y_t, \hat{y}_t) + \frac{\lambda_4 \sum_{t=1}^{T} \alpha_t L_{L_1}(\mathrm{box}_t, \widehat{\mathrm{box}_t})}{\sum_{t=1}^{T} \alpha_t + \epsilon}$$

(7)

where T is the number of characters, λ_4 is the weight for localization loss, α_t is an indicator for whether it has ground-truth character bounding box, and ϵ is a small positive number to avoid zero denominator. In practice, we do the summation and average on a batch level, to balance the loss between long and short text.

Post-processing We partition text lines into words using the predicted *space* character. We obtain word-level bounding boxes by finding the minimum-area axis-aligned bounding boxes of each word's characters. Finally, we project these word and character bounding boxes back to the image space. When we perform BezierAlign [30] in line cropping,

we build a bijection from coordinates in text line crops to coordinates in the input image. We re-use this bijection to compute this projection operation (detailed in *supplementary material Sect. A*).

4. Experiments

In this section, we evaluate the proposed method on a number of benchmarks. We first introduce the experimental settings, including the training and test datasets, the hyper-parameters of models, and the evaluation practices. We compare our method to the current state-of-the-art on end-to-end text spotting, text detection, and geometric layout analysis [34]. Finally, we conduct comprehensive ablation studies and analyze our design choices.

4.1. Experiment Setting

Unified-Detector-Polygon We base our UDP implementation on the official repository[3] of Unified Detector [34]. The input resolution is 1600×1600. Model dimensions are $N = 384$, $D = 256$, $C = 128$ respectively, using the same settings as Unified Detector [34]. As for the Bezier polygon prediction head, we use a 2-layer MLPs for both branches, with a hidden state dimension of 256. ReLU and LayerNorm [1] are applied in between the two layers. The AABB and Local Bezier prediction head outputs are activated by a sigmoid and a linear function respectively. We use $m = 3$, i.e. cubic order Bezier Curves. For the loss balancing weights, we set $\lambda_1 = 1.0$, $\lambda_2 = 2.5$, and $\lambda_3 = 0.5$. The ratio of λ_1 and λ_2 are set after DETR [8]. UDP is trained on 128 TPUv3 devices for $100K$ iterations with a batch size of 256, AdamW [39] optimizer, cosine learning rate [38] of 0.001, and weight decay of 0.05. We train UDP on a combination of the training sets of HierText [34] and CTW1500 [66], which both provide line-level text annotations. During training, images are randomly rotated, cropped, padded, and resized to the input resolution. A random scheme of color distortion [11] is also applied.

L2C2W Recognizer We use the TensorFlow Model Garden library [15] to implement our model. Input text lines are resized to height of 40 pixels and padded to width of 1024 pixels to accommodate the variable aspect ratios of lines. The CNN backbone is a MobileNetV2 [50] model with 7 identical blocks each with a filter dimension of 64 and an expansion ratio of 8. The following strided convolution has 128 filters. The transformer encoder stack consists of 8 encoder layers, with hidden size of 256 and 4 heads for each layer. The inner layers of FFNs in transformer encoders have a hidden size of 512. We use a single layer transformer decoder with hidden size 256 and only 1 attention head. The character classification head is trained to recognize case-sensitive Latin characters, digits and printable punctuation

---

[3]https://github.com/tensorflow/models/tree/master/official/projects/unified_detector

symbols (see *supplementary material Sect. B*). We set $\lambda_4 = 0.05$ for the bounding box loss. L2C2W is trained on 16 TPUv3 cores for $200K$ iterations with a batch size of 1024 and the same optimizer setup as UDP. The training data consists of SynthText [14], Synth90K [17], HierText [34], ICDAR 2015 [19], Total-Text [10], CTW1500 [66], and an internal dataset of $1M$ synthetic text lines, with a sampling ratio of $[0.25, 0.20, 0.25, 0.0005, 0.001, 0.001, 0.25]$. From the full-image datasets [10, 14, 19, 34, 66], we use the ground-truth text polygon to crop and rectify text. SynthText and HierText provide word and line-level annotations. The internal synthetic dataset generation process utilizes a similar method to Synth90K [17] but mainly contains text lines instead of single words[4]. SynthText and our internal synthetic dataset provide annotations for character-level bounding boxes.

Evaluation Practices Unless specified, e.g. in ablation studies, we use the same model and weights in all experiments. We do not perform fine-tuning on individual datasets. During inference, we filter the model's output with a confidence threshold of 0.5 for the detector and 0.8 for the recognizer. We determine these thresholds on the HierText validation set and apply them to all experiments.

4.2. Results on End-to-End Text Spotting

4.2.1 Comparison with State-of-the-Art Results

We evaluate the proposed HTS method on ICDAR 2015 Incidental [19] and Total-Text [10], the most popular benchmarks for straight and curved text respectively, and compare our results with current state-of-the-art. The evaluation of ICDAR 2015 is case-insensitive and includes several heuristics with regard to punctuation symbols and text length. In the *End-to-End* mode, if a ground-truth text starts with or ends with punctuation, it is considered a true positive match whether or not the prediction includes those punctuation symbols. In the *Word-Spotting* mode, both ground-truth and predictions are normalized by: (1) removing the *'s* and *'S* suffix; (2) removing dash ('-') prefix and suffix; (3) remove other punctuation symbols; (4) only keep normalized words that are at least 3 letters long. The Total-Text dataset does not provide an evaluation script for text spotting, and previous works evaluate their results using a script[5] adapted from ICDAR 2015's. This script inherits the aforementioned heuristics but computes IoU based on polygons as opposed to rotated bounding boxes. Additionally, both datasets are evaluated with the help of lexicon lists of different levels of perplexity[6].

To adapt to these heuristics, we transform the output of HTS by: (1) convert all characters to lower cases; (2) re-

move all non-alphanumeric symbols; (3) remove a detection if it consists of only punctuation symbols; (4) use edit distance to pick the best-match when lexicons are used.

Table 1 summarizes our evaluation results. We mark methods with different labels based on: a) the ability to recognize case-sensitive or case-insensitive characters and b) whether the underlying models are fine-tuned on the target dataset. On ICDAR 2015, our proposed HTS surpasses the recent state-of-the-art UNITS [20] considerably and beats previous ones significantly without fine-tuning. In the *Word-Spotting* mode, HTS has a large margin of **+1.45** / **+0.82** / **+0.53** on S/W/G lexicons and almost matches UNITS on non-lexicon. In the *End-to-End* mode, HTS also achieves considerable margins on all lexicon settings. Note that, the strongest competitor UNITS uses additional training data from TextOCR [54] while we don't, demonstraing the advantage of our method.

On Total-Text, we surpass all current state-of-the-art in both settings. Some of these prior arts [21, 49, 67] fine-tune their models on Total-Text which boosts the performance on this target dataset at the cost of dropping performance on others. Also note that, some prior arts [21, 22, 26, 44] limit recognition to case-insensitive letters and no punctuation symbols, while ours operate in a case-sensitive mode, a more difficult but more important one. This is not reflected in the scores due to the text normalization rules in the evaluation protocol.

4.2.2 Comparison based on HierText's Eval

We also compare the proposed HTS method with others under the evaluation protocol[7] of HierText [34]. The HierText protocol directly compares predictions against ground-truth, without normalizing letter cases, punctuation symbols, or filtering based on text lengths. It does not have lexicon modes either. Compared with the ICDAR 2015 protocol, it provides a more strict and comprehensive comparison since letter cases, punctuation symbols, and text of different lengths are all important in real-scenario applications.

For a more fair comparison, we **re-train** several previous state-of-the-art methods that have opensourced code by the time of this work, including MTSv3 [26], TESTR [67] and GLASS [49], using their open-source codes. We use the same combination of HierText [34], Total-Text [10], CTW1500 [66], SynthText [14], and ICDAR 2015 [19] as training data and evaluate on HierText [34], Total-Text [10], and ICDAR 2015 [19]. We obtain results on HierText test set using the online platform[8] since the test set annotation is not released. We are also the first to report results on the HierText test set [34]. Results are summarized in Table 2.

---

[4]See Supp. Sect. C. The dataset will be made publicly available.
[5]https://github.com/MhLiao/MaskTextSpotterV3/tree/master/evaluation/totaltext/e2e
[6]https://rrc.cvc.uab.es/?ch=4&com=tasks

[7]https://github.com/google-research-datasets/hiertext/blob/main/eval.py
[8]https://rrc.cvc.uab.es/?ch=18

| Method | ICDAR 2015 Incidental | | | | | | | | Total-Text | | | | | |
| | Word-Spotting | | | | End-to-End | | | | N | | | Full | | |
	S	W	G	N	S	W	G	N	P	R	F1	P	R	F1
MTSv3★ [26]	83.1	79.1	75.1	-	83.3	78.1	74.2	-	-	-	71.2	-	-	78.4
MANGO★ [44]	85.2	81.1	74.6	-	85.4	80.1	73.9	-	-	-	68.9	-	-	78.9
YAMTS★ [22]	86.8	82.4	76.7	-	85.3	79.8	74	-	-	-	71.1	-	-	78.4
CharNet [61]	-	-	-	-	83.10	79.15	69.14	65.73	-	-	69.2	-	-	-
TESTR♯ [67]	-	-	-	-	85.16	79.36	73.57	65.27	-	-	73.3	-	-	83.9
Qin et al.♯ [45]	-	-	-	-	85.51	81.91	-	69.94	-	-	70.7	-	-	-
TTS★ [21]	85.0	81.5	77.3	-	85.2	81.7	77.4	-	-	-	75.6	-	-	84.4
GLASS♯ [49]	86.8	82.5	78.8	71.69*	84.7	80.1	76.3	70.15*	-	-	76.6	-	-	83
UNITS★ [20]	88.1	84.9	80.7	**78.7**	88.4	83.9	79.7	78.5	-	-	77.3	-	-	85.0
HTS★♯ (ours)	**89.55**	**85.72**	**81.23**	78.62	**89.38**	**84.61**	**80.69**	**78.81**	80.41	75.92	**78.10**	90.12	80.74	**85.17**

Table 1. **Results for ICDAR 2015 and Total-Text.** 'S', 'W', 'G' and 'N' refer to *strong*, *weak*, *generic* and *no* lexicons. 'Full' for Total-Text means all test set words, and is equivalent to the *weak* setting in ICDAR 2015. '-' means scores are not reported by the papers. '*' means scores are obtained from the open-source code and weights. ★ means models are not fine-tuned on individual datasets. ♯ means models recognize all symbol classes, including case-sensitive characters and punctuation symbols.

| Method | ICDAR 2015 | | | Total-Text | | | HierText test | | |
	P	R	F1	P	R	F1	P	R	F1
TESTR	65.52	68.08	66.78	59.40	68.33	63.55	65.05	44.89	53.12
MTSv3	63.89	58.88	61.28	64.13	62.85	63.48	66.61	41.29	50.98
GLASS	74.11	63.08	68.15	68.54	60.12	64.05	73.84	57.20	64.47
HTS (ours)	**81.87**	**68.41**	**74.53**	**75.65**	**69.43**	**72.40**	**86.71**	**68.48**	**76.52**

Table 2. Results under the evaluation protocol of HierText.

| Method | Line Grouping | | | | | Paragraph Grouping | | | | |
	P	R	F1	T	PQ	P	R	F1	T	PQ
Unified Detector [34]	79.64	80.19	79.91	77.87	62.23	**76.04**	62.45	68.58	78.17	53.60
HTS (ours)	**82.71**	**82.03**	**82.37**	**80.51**	**66.31**	75.26	**75.98**	**75.62**	**79.67**	**60.25**

Table 3. **Results of geometric layout analysis on HierText test set.** *Panoptic Quality*, equals to the product of *F1* and *Tightness*.

Our HTS achieves significant advantage over these baselines by a large margin on all datasets, proving the effectiveness of our method across straight and curved text, and sparse and dense text. For ICDAR 2015 and Total-Text, the performance gap with Tab. 1 highlights the impact of text normalization and the use of lexicon lists, and that with such heuristics in evaluation we tend to overestimate the progress of text spotting method's accuracy. HierText is a new dataset that is characterized by its high word density of more than 100 words per image, a variety of image domains, a diversity in text sizes and locations and an abundance in text lines that have plenty of punctuation symbols. The recall rate is lower than 45% for TESTR and MTSv3, and lower than 60% for GLASS, while our proposed HTS can recall more than 68% of words. That indicates that the word-centric design of most existing text spotting models is not optimal for natural images with high text density.

4.3. Results on Geometric Layout Analysis

The proposed HTS is able to estimate the text's layout structure in images, as shown in Fig. 4. We further evaluate our model on HierText on the geometric layout analysis
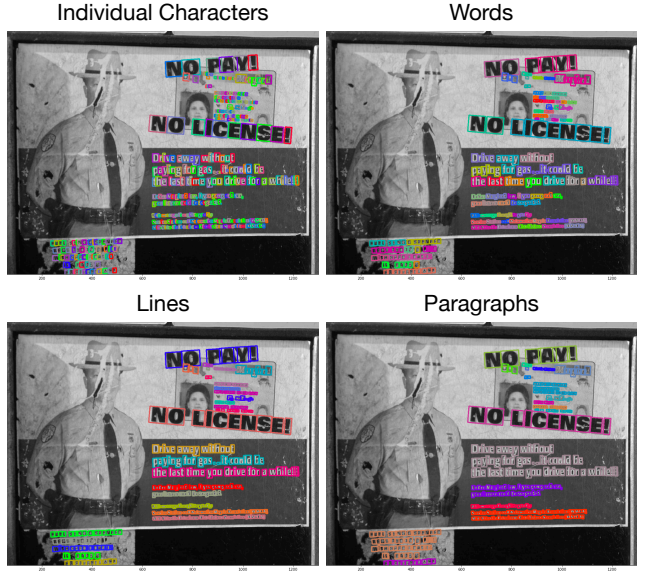

Figure 4. **Qualitative results for layout analysis.** We draw character bounding boxes with different colors to indicate layout at different levels.

task, and summarize the results in Table 3. HTS achieves betters scores in the PQ metric on both line (**+4.08**) and paragraph grouping (**+6.65**) compared to Unified Detector. Most notably, line and paragraph predictions are formed as union masks of underlying character boxes. This indicates that our character localization, as well as the word box estimation based on it, are accurate.

4.4. Ablation Studies

To better understand the effectiveness of our design choices, we conduct ablation studies and summarize the results in Tab. 4. Different from the previous sections, here we use 1024×1024 as input image resolution.

Single FFN Prediction Head

Location and Shape Decoupling Prediction Head

Figure 5. **Qualitative comparison** between Single FFN prediction head (Top) and the proposed *Location and Shape Decoupling Module (LSDM)* prediction head (Bottom) for Bezier curve polygons. The original images are turned to gray for clearer views.

Abaltion	Total-Text			HierText Val		
	P	R	F1	P	R	F1
Full	**72.33**	**62.95**	**67.31**	86.54	**67.03**	**75.55**
w/o LSDM	68.79	56.12	61.81	**86.75**	60.97	71.61
Mask-based detection	64.52	54.39	59.02	84.87	63.52	72.66
Word-level detection	71.54	58.61	64.43	80.71	51.94	63.21
MaX-DeepLab	70.16	59.32	64.28	87.66	65.07	74.69

Table 4. Ablation results as evaluated on the text spotting task.

LSDM We replace our LSDM prediction head with a single FFN branch prediction head to produce the Global Bezier directly, and remove the 2 loss items on AABB. We use $\lambda_5 = 3$ for the weight of this loss, which is larger than λ_{1-3} to compensate for the difference of loss scales. The model is trained on the same combination of HierText and CTW1500. As shown in Tab. 4, removal of LSDM results in a sharp drop of text spotting performances on both Total-Text (-5.50) and HierText validation set (-3.94). Fig. 5 further demonstrates that LSDM is important for the learning of text shapes. Without LSDM, predicted location is only roughly correct but its shape is inaccurate. This shows that training Bezier prediction head with L1 loss on diverse datasets could be dominated by the location learning and thus shape prediction fails. The proposed LSDM, on the other hand, can solve this issue by separating and balancing the learning of location and shape.

Mask v.s. Polygon One main difference between our UDP with Unified Detector [34] is that UDP produces polygons as output as opposed to masks. In this ablation study, we use the mask outputs as detections and find minimum-area rotated bounding boxes instead of using the predicted Bezier polygons. This results in a significant drop in Total-Text (-8.29) and a less severe drop in HierText (-2.89). Mask

representation is unsuitable for curved text spotting since it is non-trivial to crop and rectify with masks. Note that HierText consists mostly of straight text and is thus less affected.

Word Based v.s. Line Based OCR We train HTS on HierText and Total-Text for the word spotting task, as opposed to line-level. The line-based model is better than the word based model on both Total-Text (+2.88), a sparse text dataset, and HierText (+12.34), a dense text dataset (Tab. 4). The recall rate on HierText drops by (-15.09) if the model detects words instead of lines. It is consistent with Tab. 2, where word-based current arts have much lower scores on the dense HierText dataset.

Choice of Backbone We train two versions of HTS, one with MaX-DeepLab as the backbone and the other with KMaX-DeepLab as backbone. HTS with KMaX-DeepLab achieves **+3.03** / **+0.86** better F1 scores on Total-Text and HierText respectively, demonstrating the advantage of KMaX-DeepLab, a follow-up model of MaX-DeepLab. In addition to improved accuracy, the KMaX-DeepLab version of HTS can run at a much higher speed, with a 5.6 FPS on average on HierText, while the MaX-DeepLab version is 1.2 FPS, when measured on A100. Adopting KMaX-DeepLab benefits both accuracy and latency.

4.5. Limitations

Latency On a A100 GPU, our method runs at 7.8 FPS on ICDAR 2015 and Total-Text, and 5.6 FPS on HierText which is an order of magnitude more dense, while TESTR [67] runs at 10.2 FPS on HierText. We believe a faster backbone for UDP can help. Sharing features for UDP and L2C2W i.e. making it end-to-end trainable will also save computations.

Line labels The training of UDP requires line level annotations which are limited in few public datasets. However, it is relatively low-cost to annotate line grouping on top of existing word-level polygons. Line grouping of ground-truth words can also be accurately estimated using heuristics based on word size, location, and orientation.

Character Localization Benchmark datasets used in this work do not provide character-level labels so we are unable to evaluate the accuracy of our character localization. We can only use word level text spotting and layout analysis results as a proxy as it highly depends on character localization quality.

5. Conclusion

In this paper, we propose the first Hierarchical Text Spotter (HTS) for the joint task of text spotting and layout analysis. HTS achieves new state-of-the-art performance on multiple word-level text spotting benchmarks as well as a geometric layout analysis task.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 5

[2] Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. Character region awareness for text detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9365–9374, 2019. 1

[3] Youngmin Baek, Seung Shin, Jeonghun Baek, Sungrae Park, Junyeop Lee, Daehyun Nam, and Hwalsuk Lee. Character region attention for text spotting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pages 504–521. Springer, 2020. 2

[4] Christian Bartz, Haojin Yang, and Christoph Meinel. See: towards semi-supervised end-to-end scene text recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018. 2

[5] Darwin Bautista and Rowel Atienza. Scene text recognition with permuted autoregressive sequence models. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 178–196. Springer, 2022. 1

[6] Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marçal Rusinol, Ernest Valveny, CV Jawahar, and Dimosthenis Karatzas. Scene text visual question answering. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4291–4301, 2019. 1

[7] Thomas M Breuel. Two geometric algorithms for layout analysis. In *International workshop on document analysis systems*, pages 188–199. Springer, 2002. 1

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 2, 5

[9] Roldano Cattoni, Tarcisio Coianiz, Stefano Messelodi, and Carla Maria Modena. Geometric layout analysis techniques for document image understanding: a review. *ITC-irst Technical Report*, 9703(09), 1998. 1

[10] Chee Kheng Ch'ng and Chee Seng Chan. 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, pages 935–942. IEEE, 2017. 2, 6

[11] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020. 5

[12] Cheng Da, Peng Wang, and Cong Yao. Levenshtein ocr. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 322–338. Springer, 2022. 1, 2

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 3

[14] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2315–2324, 2016. 4, 6

[15] Xianzhi Du Yeqing Li Abdullah Rashwan Le Hou Pengchong Jin Fan Yang Frederick Liu Jaeyoun Kim Hongkun Yu, Chen Chen and Jing Li. TensorFlow Model Garden. https://github.com/tensorflow/models, 2020. 5

[16] Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. Layoutlmv3: Pre-training for document ai with unified text and image masking. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4083–4091, 2022. 1, 3

[17] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014. 6

[18] Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 1–6. IEEE, 2019. 2

[19] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. Icdar 2015 competition on robust reading. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1156–1160. IEEE, 2015. 2, 6

[20] Taeho Kil, Seonghyeon Kim, Sukmin Seo, Yoonsik Kim, and Daehee Kim. Towards unified scene text spotting based on sequence generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15223–15232, 2023. 2, 6, 7

[21] Yair Kittenplon, Inbal Lavi, Sharon Fogel, Yarin Bar, R Manmatha, and Pietro Perona. Towards weakly-supervised text spotting using a multi-task transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4604–4613, 2022. 2, 6, 7

[22] Ilya Krylov, Sergei Nosov, and Vladislav Sovrasov. Open images v5 text annotation and yet another mask text spotter. In *Asian Conference on Machine Learning*, pages 379–389. PMLR, 2021. 2, 6, 7

[23] Chen-Yu Lee, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister. Formnet: Structural encoding beyond sequential modeling in form document information extraction. *arXiv preprint arXiv:2203.08411*, 2022. 1, 3

[24] Joonho Lee, Hideaki Hayashi, Wataru Ohyama, and Seiichi Uchida. Page segmentation using a convolutional neural network with trainable co-occurrence features. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1023–1028. IEEE, 2019. 1, 3

[25] Chenliang Li, Bin Bi, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si. Structurallm: Structural pre-training for form understanding. *arXiv preprint arXiv:2105.11210*, 2021. 1, 3

[26] Minghui Liao, Guan Pang, Jing Huang, Tal Hassner, and Xiang Bai. Mask textspotter v3: Segmentation proposal network for robust scene text spotting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 706–722. Springer, 2020. 2, 6, 7

[27] Minghui Liao, Jian Zhang, Zhaoyi Wan, Fengming Xie, Jiajun Liang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Scene text recognition from two-dimensional perspective. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 8714–8721, 2019. 2

[28] Minghui Liao, Zhisheng Zou, Zhaoyi Wan, Cong Yao, and Xiang Bai. Real-time scene text detection with differentiable binarization and adaptive scale fusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):919–931, 2022. 1

[29] Xuebo Liu, Ding Liang, Shi Yan, Dagui Chen, Yu Qiao, and Junjie Yan. Fots: Fast oriented text spotting with a unified network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5676–5685, 2018. 2

[30] Yuliang Liu, Chunhua Shen, Lianwen Jin, Tong He, Peng Chen, Chongyu Liu, and Hao Chen. 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, 2021. 2, 3, 4, 5

[31] Shangbang Long, Yushuo Guan, Kaigui Bian, and Cong Yao. A new perspective for flexible feature gathering in scene text recognition via character anchor pooling. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2458–2462. IEEE, 2020. 2

[32] Shangbang Long, Yushuo Guan, Bingxuan Wang, Kaigui Bian, and Cong Yao. Rethinking irregular scene text recognition. *arXiv preprint arXiv:1908.11834*, 2019. 2

[33] Shangbang Long, Xin He, and Cong Yao. Scene text detection and recognition: The deep learning era. *International Journal of Computer Vision*, 129(1):161–184, 2021. 1

[34] Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. Towards end-to-end unified scene text detection and layout analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1049–1059, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[35] Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. Icdar 2023 competition on hierarchical text detection and recognition. In Gernot A. Fink, Rajiv Jain, Koichi Kise, and Richard Zanibbi, editors, *Document Analysis and Recognition - ICDAR 2023*, pages 483–497, Cham, 2023. Springer Nature Switzerland. 3

[36] Shangbang Long, Jiaqiang Ruan, Wenjie Zhang, Xin He, Wenhao Wu, and Cong Yao. Textsnake: A flexible representation for detecting text of arbitrary shapes. In *Proceedings of the European conference on computer vision (ECCV)*, pages 20–36, 2018. 1, 2

[37] Shangbang Long and Cong Yao. Unrealtext: Synthesizing realistic scene text images from the unreal world. *arXiv preprint arXiv:2003.10608*, 2020. 4

[38] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 5

[39] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 5

[40] Byeonghu Na, Yoonsik Kim, and Sungrae Park. Multimodal text recognition networks: Interactive enhancements between visual and semantic features. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 446–463. Springer, 2022. 1, 2

[41] Oren Nuriel, Sharon Fogel, and Ron Litman. Textadain: Paying attention to shortcut learning in text recognizers. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 427–445. Springer, 2022. 1

[42] Dezhi Peng, Xinyu Wang, Yuliang Liu, Jiaxin Zhang, Mingxin Huang, Songxuan Lai, Jing Li, Shenggao Zhu, Dahua Lin, Chunhua Shen, et al. Spts: single-point text spotting. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4272–4281, 2022. 1, 2

[43] Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S Nassar, and Peter Staar. Doclaynet: A large human-annotated dataset for document-layout segmentation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3743–3751, 2022. 2

[44] Liang Qiao, Ying Chen, Zhanzhan Cheng, Yunlu Xu, Yi Niu, Shiliang Pu, and Fei Wu. Mango: A mask attention guided one-stage scene text spotter. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2467–2476, 2021. 6, 7

[45] Siyang Qin, Alessandro Bissacco, Michalis Raptis, Yasuhisa Fujii, and Ying Xiao. Towards unconstrained end-to-end text spotting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4704–4714, 2019. 1, 2, 7

[46] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018. 3

[47] Zobeir Raisi, Georges Younes, and John Zelek. Arbitrary shape text detection using transformers. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 3238–3245. IEEE, 2022. 2, 4

[48] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019. 4

[49] Roi Ronen, Shahar Tsiper, Oron Anschel, Inbal Lavi, Amir Markovitz, and R Manmatha. Glass: Global to local attention for scene-text spotting. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 249–266. Springer, 2022. 1, 2, 6, 7

[50] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 4, 5

[51] Sebastian Schreiber, Stefan Agne, Ivo Wolf, Andreas Dengel, and Sheraz Ahmed. Deepdesrt: Deep learning for detection and structure recognition of tables in document images. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 1162–1167. IEEE, 2017. 3

[52] Baoguang Shi, Xiang Bai, and Serge Belongie. Detecting oriented text in natural images by linking segments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2550–2558, 2017. 2

[53] Amanpreet Singh, Vivek Natarjan, Meet Shah, Yu Jiang, Xinlei Chen, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8317–8326, 2019. 1

[54] Amanpreet Singh, Guan Pang, Mandy Toh, Jing Huang, Wojciech Galuba, and Tal Hassner. Textocr: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8802–8812, June 2021. 6

[55] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014. 2, 4

[56] Jingqun Tang, Wenqing Zhang, Hongye Liu, MingKun Yang, Bo Jiang, Guanglong Hu, and Xiang Bai. Few could be better than all: Feature sampling and grouping for scene text detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4563–4572, 2022. 2, 4

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 2, 3, 4, 5

[58] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5463–5474, 2021. 1, 3

[59] Peng Wang, Cheng Da, and Cong Yao. Multi-granularity prediction for scene text recognition. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 339–355. Springer, 2022. 1

[60] Renshen Wang, Yasuhisa Fujii, and Ashok C Popat. Post-ocr paragraph recognition by graph convolutional networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 493–502, 2022. 2, 3

[61] Linjie Xing, Zhi Tian, Weilin Huang, and Matthew R Scott. Convolutional character networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9126–9136, 2019. 2, 7

[62] Xiao Yang, Ersin Yumer, Paul Asente, Mike Kraley, Daniel Kifer, and C Lee Giles. Learning to extract semantic structure from documents using multimodal fully convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5315–5324, 2017. 1

[63] Maoyuan Ye, Jing Zhang, Shanshan Zhao, Juhua Liu, Bo Du, and Dacheng Tao. Dptext-detr: Towards better scene text detection with dynamic points in transformer. *arXiv preprint arXiv:2207.04491*, 2022. 1

[64] Moonbin Yim, Yoonsik Kim, Han-Cheol Cho, and Sungrae Park. Synthtiger: synthetic text image generator towards better text recognition models. In *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part IV 16*, pages 109–124. Springer, 2021. 4

[65] Qihang Yu, Huiyu Wang, Siyuan Qiao, Maxwell Collins, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. k-means mask transformer. In *European Conference on Computer Vision*, pages 288–307. Springer, 2022. 4

[66] Liu Yuliang, Jin Lianwen, Zhang Shuaitao, and Zhang Sheng. Detecting curve text in the wild: New dataset and new solution. *arXiv preprint arXiv:1712.02170*, 2017. 5, 6

[67] Xiang Zhang, Yongwen Su, Subarna Tripathi, and Zhuowen Tu. Text spotting transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9519–9528, 2022. 1, 2, 6, 7, 8

[68] Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. Publaynet: largest dataset ever for document layout analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1015–1022. IEEE, 2019. 2