LayoutFormer: Hierarchical Text Detection Towards Scene Text Understanding

Min Liang, Jia-Wei Ma, Xiaobin Zhu*, Jingyan Qin, Xu-Cheng Yin
University of Science and Technology Beijing
{lm20200126, mjw20151001}@hotmail.com, {zhuxiaobin, xuchengyin}@ustb.edu.cn,
qinjingyanking@foxmail.com

Abstract

Existing scene text detectors generally focus on accurately detecting single-level (i.e., word-level, line-level, or paragraph-level) text entities without exploring the relationships among different levels of text entities. To comprehensively understand scene texts, detecting multi-level texts while exploring their contextual information is critical. To this end, we propose a unified framework (dubbed LayoutFormer) for hierarchical text detection, which simultaneously conducts multi-level text detection and predicts the geometric layouts for promoting scene text understanding. In LayoutFormer, WordDecoder, LineDecoder, and ParaDecoder are proposed to be responsible for word-level text prediction, line-level text prediction, and paragraph-level text prediction, respectively. Meanwhile, WordDecoder and ParaDecoder adaptively learn word-line and line-paragraph relationships, respectively. In addition, we propose a Prior Location Sampler to be used on multi-scale features to adaptively select a few representative foreground features for updating text queries. It can improve hierarchical detection performance while significantly reducing the computational cost. Comprehensive experiments verify that our method achieves state-of-the-art performance on single-level and hierarchical text detection.

1. Introduction

Reading and understanding texts in scene images and digital documents is important in various real-world applications, including visual recognition [40], scene understanding [4], and text-based VQA [1, 30]. Compared to words, phrases and sentences express more semantic messages, and paragraphs convey richer contextual semantic messages. For better reading and understanding texts in scenes, it is necessary to simultaneously detect multi-level texts (i.e., word-level for words, line-level for phrases and sentences, and paragraph-level for paragraphs) and predict their geometric layouts [16, 44], i.e., hierarchical text detection [25]. Hierarchical texts can deliver complete and meaningful text messages, which is beneficial for various downstream tasks.

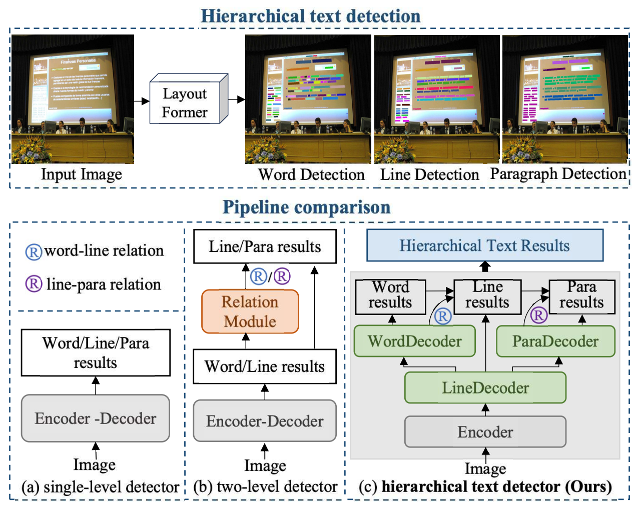

Figure 1. **Top**: LayoutFormer is introduced for hierarchical text detection, which simultaneously detects multi-level texts (i.e., word-level, line-level, and paragraph-level) and predicts their geometric layouts. **Bottom**: Pipeline comparisons of (a) single-level detector, (b) two-level detector, and (c) hierarchical text detector.

Although existing scene text detectors have achieved promising performance, they [32, 47] often concentrate on detecting single-level text, which decodes the encoded features of an input image into their representation of independent text entities, e.g., words, lines, or paragraphs. As shown in Fig. 1 (a), single-level detectors only predict one level of text results and do not explore the contextual relationship among different levels of texts. This makes them unable to deliver more informative messages for facilitating scene text understanding. Recently, some methods have investigated two-level text detection. A schematic of the framework is shown in Fig. 1 (b). CUTE [43] proposes a

*Corresponding Author. This work was supported by National Key Research and Development Program of China (2020AAA0109701), National Science Fund for Distinguished Young Scholars (62125601), and National Natural Science Foundation of China (62076024, 62172035).

two-stage network, which first adopts a Transformer structure like DETR [3] to perform word/line bounding boxes and then directly learns the sequential relationships between detected text boxes based on visual features after ROI operation and positional information. Unified Detector [24] proposes a unified scene text detection and layout analysis network, which first uses a Transformer-based structure [35] to generate word/line mask maps and then models their affiliate relationships. These two methods are all supervised by an affinity matrix or index label, directly modelling relationships on the word-level/line-level detection results. They only rely on the former detection results to model contextual relationships, making the model's performance highly limited by the text detection results at the first stage. Moreover, this paradigm cannot be easily extended to the task of text detection at more than two levels.

Existing scene text detectors generally adopt multi-scale feature maps that contain rich information for accurate prediction. However, methods based on the Transformer encoder-decoder structure often suffer from huge computational and memory costs on high-resolution feature maps. This is because the attention weight computation in the Transformer decoder is of linear computation w.r.t. pixel numbers. There have been several work to tackle the issues. Deformable DETR [51] introduces the multi-scale deformable attention module to select a small set of key locations. AdaMixer [10] adaptively samples features over the space and scales of an object. However, these methods are based on estimated offsets of instance coordinates, not applicable to segmentation methods. In image segmentation, Mask2Former [6] claims that local features can well update query features and proposes to use image features of foreground regions. TSP [31] adopts a RoIAlign operation to extract features of RoIs. However, these methods still suffer from complex operations. We think that it is enough to update query features with only foreground features of a few representative pixels.

In this paper, we propose a hierarchical text detector named LayoutFormer, which can simultaneously produce hierarchical text detection boxes and their layout relationships. Unlike using inter-level relationships as supervision, we directly adopt multi-level text boxes as supervision, which can significantly reduce the dependence on the former text detection results and implicitly exploit multi-level textual information. In LayoutFormer, we propose three Transformer-based modules, i.e., WordDecoder, LineDecoder, and ParaDecoder, for detecting word-level texts, line-level texts, and paragraph-level texts, respectively. We first adopt text line as the detection unit, and LineDecoder updates and forms line query features. Because a line can be split into several words, WordDecoder learns the deviation of each word within a line based on the line query features and forms final word query features. ParaDecoder aggregates line query features by learning line-paragraph relationships and forms final paragraph query features. In addition, we propose a Prior Location Sampler, which adaptively selects a few representative features that are highly relevant to scene texts. The sampled features are used for cross-attention operation to update query features. Prior Location Sampler can improve the model performance while reducing the training cost. Extensive experiments verify the superior performance on hierarchical text detection.

In summary, our contributions are as follows:

- We propose an innovative hierarchical text detector that simultaneously detects multi-level text instances and explores their contextual information for predicting the geometric layouts, finally performing hierarchical outputs.
- We propose WordDecoder, LineDecoder, and ParaDecoder in LayoutFormer to decode word-level text, line-level text, and paragraph-level text, respectively. WordDecoder and ParaDecoder adaptively learn word-line relationships and line-paragraph relationships, respectively.
- We propose a Prior Location Sampler to adaptively select a few representative foreground features for updating queries, which can significantly reduce the computational cost while improving the detection performance.
- Extensive experiments on multiple publicly available datasets verify the state-of-the-art performance of our LayoutFormer.

2. Related Work

2.1. Scene Text Detection

Deep learning-based scene text detection methods have been widely investigated and roughly divided into bottom-up and top-down methods. The bottom-up methods generally detect local components [23, 29, 33] or text pixels [19, 37] for grouping text instances. TextSnake [23] describes a text instance as a sequence of ordered, overlapping disks. The pixel-based methods use some auxiliary information (e.g., similarity vectors in PAN [38], and threshold map in DB [19]) or post-processings (e.g., a progressive scale expansion algorithm in PSENet [37]) to generate text instances. The top-down methods [18, 50] directly predict bounding boxes of scene text instances. To detect arbitrary-shaped texts, some methods [39, 49, 52] localize the key points on the contours of scene texts. Recently, DETR [3] presents a Transformer-based architecture for object detection and achieved great success. Inspired by it, many researchers [24, 32, 47] apply the Transformer structure to scene text detection. Overall, existing scene text detection methods mainly address single-level text detection, ignoring the text detection of multiple levels and further predicting the hierarchical relationships.

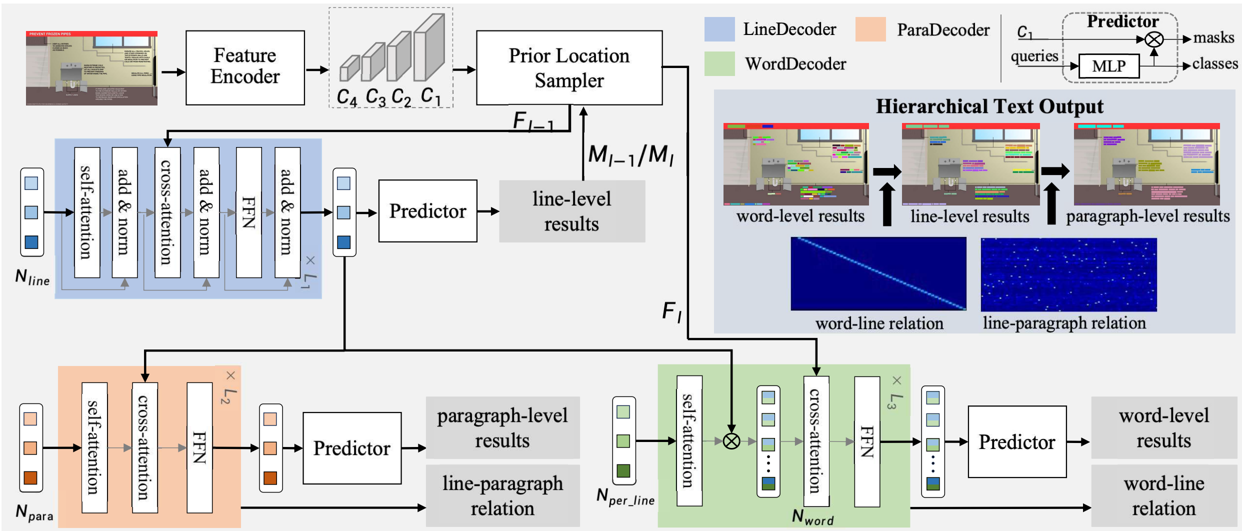

Figure 2. Illustration of the proposed LayoutFormer, which mainly consists of a feature encoder, a Prior Location Sampler, three Transformer decoders (WordDecoder, LineDecoder, and ParaDecoder) and three predictors. M_l denotes predicted mask maps of text lines by l-th Transformer decoder, and F_l denotes the corresponding image features generated by M_l. We frist predict text lines, and then predict words and paragraphs. In testing, word-level results produce line-level results based on word-line relationships, and then paragraph-level results are produced based on line-paragraph relationships, finally obtaining hierarchical text detection results.

Detection Unit	Query	Memory/GB	PQ
word	256	12.1	53.09
line	256	10.6	57.48
paragraph	256	8.5	40.34

Table 1. Comparisons of word, line, and paragraph as text detection unit on the HierText validation set, respectively. "Query" denotes the number of text queries. "Memory" denotes GPU Memory consumed in training.

2.2. Layout Analysis

Document Layout analysis has made great progress, which can be mainly divided into object detection-based methods [17, 27, 34] and segmentation-based methods [2, 16, 41]. Inspired by object detection [15, 28] and semantic segmentation [5, 12, 22], these methods treat semantically coherent text blocks as a special kind of object. However, they fail to produce word or line-level detections and can only be used in companies with standalone text detectors, increasing the complexity of the pipeline. Some work [36] takes a hierarchical view and applies graph-based models on the finest granularity, i.e., individual words, to analyze the layout.

Recently, some work has investigated layout analysis in scene text detection. CUTE [43] detects contextual text blocks consisting of one or multiple ordered integral text units. Their model is a two-stage structure, which first detects integral text units and then models the relationship of integral text units by a designed relation module. Unified Detector [24] introduces a novel task of unified scene text detection and layout analysis, which models the two closely related tasks with a unified model. It produces mask outputs for the text detection task and an affinity matrix for the layout analysis task. The above methods are highly dependent on text detection results at the first stage. Differently, we use multi-level text boxes as supervision to learn the relationship among different levels for layout analysis.

3. Methodology

3.1. Preliminary

How to effectively model hierarchical text detection tasks? Is the top-down or bottom-up approach better? As we know, the text layouts in real scene images is highly complex, bringing significant challenges to hierarchical text detection. According to experiments and analyses, we found that paragraph-level text boxes are considerably irregular and always contain much background, which makes existing scene text detection approaches difficult to model text instances. For example, as shown in the second row of Fig. 3, a paragraph text box annotates the entire image. As listed in Tab. 1, the detection performance of paragraphs is much lower than that of words and lines. Thus, the top-down modeling approach is considered inappropriate. Both word and line are generally used as text detection units. However, text lines always contain more complete semantic information than words. The aspect ratios and sizes of words are considerably more variable than text lines, making detection more challenging. Meanwhile, if the word is used

Figure 3. Some examples of word-level (first column), line-level (second column), and paragraph-level (third column) text boxes.

as the detection unit, the detection error will accumulate bigger after two aggregation processes. Hence, we finally choose line-level as our detection unit, generate word-level texts by splitting them downward, and get paragraph-level texts by aggregating them upward.

3.2. Network Architecture

The architecture of our LayoutFormer is illustrated in Fig. 2. It mainly consists of a feature encoder, a Prior Location Sampler, three Transformer decoders and three predictors, respectively, corresponding to word-level text detection, line-level text detection and paragraph-level text detection. Specifically, given an input image $I \in R^{H\times W\times 3}$, the feature encoder with a ResNet-50 [14] backbone and a Transformer encoder [6] extracts and enhances the features and then generates multi-scale feature maps, i.e., $C_4 \in R^{\frac{H}{32}\times\frac{W}{32}\times C}$, $C_3 \in R^{\frac{H}{16}\times\frac{W}{16}\times C}$, $C_2 \in R^{\frac{H}{8}\times\frac{W}{8}\times C}$, and $C_1 \in R^{\frac{H}{4}\times\frac{W}{4}\times C}$. The typical value we use is $C = 256$. Then, Prior Location Sampler adaptively selects image features for the Transformer decoder. Afterwards, the LineDecoder, WordDecoder, and ParaDecoder update line query features $X_{line} \in R^{N_{line}\times C}$, word query features $X_{word} \in R^{N_{word}\times C}$, and paragraph query features $X_{para} \in R^{N_{para}\times C}$, respectively. The three predictors convert X_{line}, X_{word}, and X_{para} to line-level prediction (P_{line}, M_{line}), word-level prediction (P_{word}, M_{word}), and paragraph-level prediction (P_{para}, M_{para}), where $P_* \in R^{N_*\times 2}$ is text/non-text probability values of N_* instances and $M_* \in R^{N_*\times\frac{H}{4}\times\frac{W}{4}}$ is mask maps of N_* instances.

3.3. Prior Location Sampler

Mask2Former [6] proposes masked attention to attend within the foreground region of the predicted mask for each query, which can be computed as

$$X_l = softmax(B_{l-1} + Q_l K_l^T)V_l + X_{l-1}, \quad (1)$$

where l is the layer index, B_{l-1} is the binary mask prediction of $l-1$-th layer, X_l is query features at the l-th layer, Q_l is query features X_{l-1} under linear transformation, K_l and V_l are the input image features under linear transformation. X_0 denotes input query features to the Transformer decoder. B_0 is the binary mask prediction obtained from X_0. The input image features are the entire multi-scale feature maps from a feature encoder. Their computational complexity of Eq. (1) is $O(NH'W'C)$, where H' and W' are the height and width of a specific feature map respectively, and N is the number of queries. The cross-attention operation suffers from a linear complexity growth with the spatial size of feature maps. Moreover, the main accompanying problem is the high demand for computing resources.

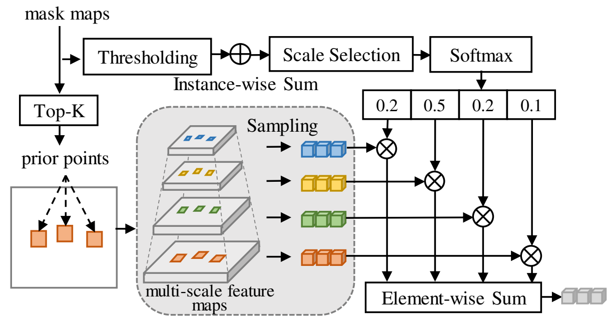

Figure 4. Illustration of the Prior Location Sampler, which selects representative foreground features with Top-K scores in predicted mask maps and then aggregates with weights.

Thus, we propose the Prior Location Sampler to select representative foreground features with Top-K scores in mask maps and obtain the image features $F_l \in R^{K\times N\times C}$ for l-th decoder layer. Our computational complexity of Eq. (1) is $O(NKC)$. In our implementation, the value $K \ll H'W'$, and thus the complexity of our cross-attention operation could be significantly reduced. The computational resource consumption due to an increase in the number of feature maps or an increase in the scale of feature maps is negligible.

The architecture of the Prior Location Sampler is illustrated in Fig. 4. Specifically, M_l denotes the predicted mask maps of l-th decoder layer, and

$$M_l = \{M_{ln} \in R^{\frac{H}{4}\times\frac{W}{4}} | n = 1, 2, ..., N\}. \quad (2)$$

For each M_{ln}, we use 0.5 for thresholding. We sort scores in M_{ln} and select features with Top-K scores [13] in multi-scale feature maps $\{C_i | i = 1, 2, 3, 4\}$ respectively, where $f_K(\cdot)$ is sampling operation. Then, we assign [20, 28] text ROIs of different scales to the pyramid levels for calculating weights [10] as Eq. (4), where S_{ln} is the cumulative sum of the binarized M_{ln}, i.e., the area of a candidate text, and 56 [20] is a scaling factor. The selected features for M_{ln} are gathered into $F_{ln} \in R^{K\times C}$:

$$F_{ln} = \sum_{i=1}^{4} w_{li} f_K(C_i, M_{ln}), \quad (3)$$

$$t_{ln} = \lfloor \log_2 \frac{S_{ln}}{56} \rfloor,$$
$$w_l = softmax[-\frac{(t_{ln} - i + 1)^2}{2}]. \quad (4)$$

The final features $F_l \in R^{K \times N \times C}$ are denoted as:

$$F_l = \{F_{ln} | n = 1, 2, ..., N\}. \quad (5)$$

3.4. Transformer Decoder

LineDecoder. We feed the selected image features $F_{l-1} \in R^{K \times N_{line} \times C}$ and N_{line} learnable line queries into LineDecoder, where $K = 32$. LineDecoder uses the standard Transformer decoder structure, consisting of self-attention, cross-attention, and feedforward network. We also add the positional embeddings to queries and keys at every self-attention and cross-attention layer. We refine line query features layer-by-layer by setting $L_1 = 4$. The updated line query features are adopted to predict l-th mask maps, which are fed into Prior Location Sampler to generate image features for the next decoder layer.

WordDecoder. We frist hypothesize that there are N_{per_line} words in each line and initialize N_{per_line} learnable word queries. Thus, the number of words in an image can be formulated as the product of the number of lines in an image and words per line, as

$$N_{word} = N_{line} \times N_{per\_line}. \quad (6)$$

So, we can decouple the self-attention between words into self-attention between lines and self-attention between words per line. Based on line queries from LineDecoder, we only need to compute self-attention between N_{per_line}, greatly reducing the computational cost. Then we obtain word query features $X_{word} \in R^{N_{word} \times C}$ by

$$X_{word} = X_{line} X_{per\_line}^T, \quad (7)$$

where $X_{line} \in R^{N_{line} \times 1 \times C}$ is line query features and $X_{per\_line} \in R^{N_{per\_line} \times 1 \times C}$ is learnable word query features per line. Then cross-attention followed by feedforward network adopts image features $F_l \in R^{K \times N_{line} \times C}$ with $K = 64$ to update X_{word}. The image features are obtained by Prior Location Sampler, which feeds the predicted mask maps of the last decoder layer in LineDecoder. In our experiments, we set $L_3 = 1$. With our word decoupling mechanism, every N_{per_line} queries predicts words within a text line for capturing word-line relationships.

ParaDecoder. We adaptively learn line-paragraph relationships and aggregate line query features for paragraph-level texts to get paragraph query features. We initialize N_{para} learnable paragraph queries. ParaDecoder also consists of self-attention, cross-attention, and feedforward networks. It uses X_{line} line query features as inputs of cross attention to update paragraph queries. Positional embeddings are added to queries and keys at every self-attention and cross-attention layer. In our experiments, we adopt X_{line} of last decoder layer in LineDecoder and set $L_2 = 1$. In this process, the learned attention weights ($R^{N_{para} \times N_{line}}$) in cross-attention of ParaDecoder are line-paragraph relationships.

3.5. Optimization

Training. Instead of relation matrices, we use multi-level text boxes as supervision. The total loss is formulated as:

$$L = L_{line} + L_{word} + L_{para}, \quad (8)$$

where L_{line}, L_{word}, and L_{para} respectively denote losses of line-level, word-level, and paragraph-level, which can be formulated as:

$$L_* = \lambda_{cls} L_{cls}(P_*, P'_*) + \lambda_{mask} L_{mask}(M_*, M'_*),$$
$$* \in \{word, line, para\}, \quad (9)$$

where P_* and P'_* are the class predictions and their corresponding ground truth, M_* and M'_* are the mask predictions and their corresponding ground truth, L_{cls} denotes binary cross-entropy loss, L_{mask} is the sum of binary cross-entropy loss and dice loss. We set $\lambda_{cls} = 2.0$ for predictions matched with ground truth and 0.1 for unmatched predictions. λ_{mask} is set to 5.0.

Inference. We present a simple inference procedure that converts class predictions and mask predictions to text boxes, which mainly consists of four steps: (1) the text queries whose class scores are over 0.3 will be selected; (2) the mask predictions of selected queries are binarized by 0.5 to get the binary maps; (3) the connected regions are obtained from the binary maps; (4) the text queries whose averaged foreground mask probability is below 0.9 will be filtered. For hierarchical text detection task, we frist generate the word-level detection boxes by the above inference procedure. Then, we use word-line relationships from WordDecoder to form a word-line geometric layouts and further use line-paragraph relationships from ParaDecoder to form a line-paragraph geometric layouts, finally performing hierarchical text detection outputs.

4. Experiments

4.1. Datasets

HierText [24]. It is a hierarchical text detection dataset consisting of 8,281 training images, 1,724 validation images, and 1,634 testing images. It annotates images hierarchically, which first annotates word locations with polygons, then clusters words into lines and lines into paragraphs.

MSRA-TD500 [45]. It is a line-level annotated arbitrary-oriented long text dataset. It consists of 300 training images and 200 testing images collected from natural scenes.

Method		Total-Text				HierText-Line					
		R	P	F	Memory/GB	R	P	F	T	PQ	Memory/GB
$C_2 - C_4$	masked attention	84.68	87.79	86.21	15.3	65.56	82.73	73.15	**77.57**	56.74	18.1
$C_2 - C_4$	Top-K&concat	84.66	87.80	86.20	15.1	65.34	82.56	72.95	77.37	56.44	18.3
$C_1 - C_4$	Top-K&concat	84.46	88.15	86.27	16.2	65.88	82.96	73.44	77.53	56.94	20.3
$C_1 - C_4$	Top-K&weight	**85.07**	**89.26**	**87.12**	14.2	**66.65**	**84.10**	**74.37**	77.29	**57.48**	17.0

Table 2. Ablation comparison of Prior Location Sampler on the Total-Text and line-level of the HierText validation set. "Memory" denotes GPU Memory consumed in training. "$C_1 - C_4$" denotes the multi-scale feature maps used in the Transformer decoder. The results show that "Top-K&weight" with "$C_1 - C_4$" achieves a considerable improvement and consumes fewer training resources than other methods.

CTW1500 [21]. It is an arbitrary-shaped scene text dataset that consists of 1,000 training images and 500 testing images. In this dataset, the annotations of text instances are line-level and labelled by a polygon with 14 key points.

Total-Text [7]. It is an arbitrary-shaped scene text dataset that contains 1,255 training and 300 testing images, which are annotated by the word-level polygon with 14 key points.

4.2. Implementation Details

In our experiments, for HierText, we only use the training images of HierText to train the models by $60k$ steps for single-level text detection and $80k$ steps for multi-level text detection. In testing, we keep the aspect ratio of testing images and resize them to 1,120 height. For MSRA-TD500, CTW1500, and Total-Text, we adopt SynthText [11] to pretrain the models by $150k$ steps. Then, we finetune the models on the corresponding real-world datasets by $40k$ steps. In testing, we keep the aspect ratio of testing images and resize them to 800 height on three datasets.

In training, we use AdamW [26] optimizer and the step learning rate schedule with an initial learning rate of 0.0001 and a weight decay of 0.05. A learning rate multiplier of 0.1 is applied to the ResNet-50 backbone. We decay the learning rate at 0.9 and 0.95 fractions of the total number of training steps by a factor of 10. We train our models with a batch size of 8. Data augmentation includes: (1) Randomly Flipping; (2) Randomly rotate them in range ($-10°$ to $10°$); (3) Randomly resize them in range (0.5 to 3.0); (4) Randomly cropping. Finally, we resize the images to 640×640.

4.3. Evaluation Metrics

For MSRA-TD500, CTW1500, and CTW1500 datasets, we follow the standard evaluation protocol Recall (R), Precision (P), and F-measure (F). For the HierText dataset, we follow [24] and adopt Panoptic Quality (PQ) as the main evaluation metric for the hierarchical text detection task.

4.4. Ablation Studies

K value in Prior Location Sampler. We compare the different Top-K values in Prior Location Sampler. Results are listed in Tab. 3. Selecting too few foreground points may fail to capture and learn the features of an entire text. Selecting too many foreground points may introduce noise, which is detrimental to the performance of the model. For Total-Text, we set K to 32, and the same for MSRA-TD500 and CTW1500. While text lines of HierText may have larger aspect ratios and scales, the performance of $K = 64$ is better than that of $K = 32$. However, we finally set K to 32 for a trade-off between performance and efficiency.

K	Total-Text			HierText-Line			
	R	P	F	R	P	F	PQ
16	84.34	88.01	86.14	65.34	81.12	72.38	55.54
32	**85.07**	**89.26**	**87.12**	66.65	84.10	74.37	57.48
64	84.75	88.63	86.64	**68.67**	**84.44**	**75.75**	**58.86**

Table 3. Ablation comparison of Top-K value on Total-Text and line-level of the HierText validation set.

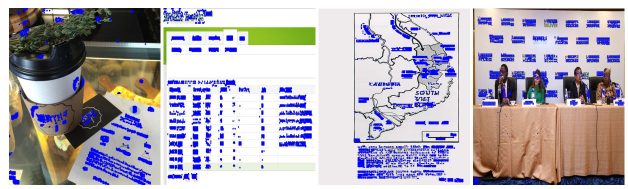

Figure 5. Visualization of Top-K sampling points in Prior Location Sampler.

Implementation of Prior Location Sampler. We evaluate the effectiveness of the Prior Location Sampler on the Total-Text and HierText validation set. Results are listed in Tab. 2. "$C_1 - C_4$" denotes that multi-scale feature maps $\{C_i|i = 1, 2, 3, 4\}$ are used in Transformer decoder. "masked attention" is followed from Mask2Former [6], which calculates the attention matrix by additionally adding a binary mask prediction. "Top-K&concat" applies Top-K strategy to multi-scale feature maps and then performs concatenation. "Top-K&weight" applies Top-K strategy to multi-scale feature maps and then element-wise sums in weights. Inspired by Mask2Former, the first row of Tab. 2 feeds the entire feature map into the Transformer decoder. The experimental results show that "Top-K&weight" apparently achieves a significant improvement in harmonic accuracy and memory. Meanwhile, due to using a small number of foreground points, our method consumes fewer training resources (GPU Memory) than other methods.

Method	Word					Line					Paragraph				
	R	P	F	T	PQ	R	P	F	T	PQ	R	P	F	T	PQ
LayoutFormer	57.04	75.25	64.89	75.88	49.24	50.39	76.32	60.70	76.12	46.21	45.89	73.14	56.40	76.19	42.97

Table 4. Experimental results of word-line-paragraph three-level text detection on the HierText testing set.

Method	N_{per_line}	N_{line}	Word PQ	Line PQ
In the below experiments, we always set L_3 to 1.				
X_{line}	10	120	44.02	36.35
F_l	10	120	50.29	48.60
In the below experiments, we use F_l as image features.				
$L_3 = 1$	10	120	50.29	48.60
$L_3 = 4$	10	120	51.07	44.05
$L_3 = 1$	6	120	46.21	39.60
$L_3 = 1$	6	200	50.12	46.00

Table 5. Ablation comparison of WordDecoder on the HierText validation set. N_{per_line} and N_{line} denote the number of word queries in each line and text line queries, respectively.

Method	N_{line}	N_{para}	Line PQ	Paragraph PQ
In the below experiments, we always set L_2 to 1.				
X_{line}	128	128	58.31	51.98
F_l	128	128	57.27	46.15

Table 6. Ablation comparison of ParaDecoder on the HierText validation set. N_{line} and N_{para} denote the number of text line queries and paragraph queries, respectively.

Implementation of WordDecoder. We conduct ablation experiments by comparing different settings of image features, L_3, N_{per_line} and N_{line} in WordDecoder. Results are listed in Tab. 5. Using F_l as image features for updating word queries is better than X_{line}. This is because to identify different text lines, X_{line} extracts consistent features of each text line, whereas X_{per_line} needs to learn differentiated features within a text line to identify different words within a line. In addition, we find that L_3 being set to 1 performs better than being set to 4. Thus, we adopt $L_3 = 1$ by default. We also conduct comparative experiments with different N_{per_line} and N_{line}. We can conclude that the larger the query number, the higher the performance.

Implementation of ParaDecoder. We compare different image features for ParaDecoder. Results are listed in Tab. 6. Adopting X_{line} as image features is better than F_l. We think that after the learning of LineDecoder, X_{line} is able to characterize text lines well.

4.5. Comparisons with State-of-the-art Methods

Hierarchical Text Detection. In this section, we evaluate the performance on the HierText testing set for hierarchical text detection. To our knowledge, few work has conducted research on hierarchical text detection. Hence, we mainly compare with [24]. Results are listed in Tab. 7, Tab. 9, and Tab. 4. Meanwhile, we demonstrate hierarchical detection results on images in Fig. 6. Both qualitative and quantitative verify the advances of our method. And we can observe that our method can precisely detect hierarchical texts.

Method	N_{word}	N_{line}	Word PQ	Line PQ
LayoutFormer	1200	120	**50.35**	**48.62**

Table 7. Experimental results of word-line two-level text detection on the HierText testing set. N_{word} and N_{line} denote the number of word queries and text line queries, respectively.

Figure 6. The hierarchical text detection examples on the HierText. The visualization images in each row are in turn the input image, word-level detection results, line-level detection results, and paragraph-level detection results.

For word-line detection, our results are listed in Tab. 7. Because there is no any prior research on this, we provide a baseline to facilitate comparisons with other methods. Benefiting from our decoupling operation on words, our N_{word} can reach 1,200, much higher than Unified Detector [24].

For line-paragraph detection, our LayoutFormer achieves the best text detection and layout analysis performance. Results are listed in Tab. 9. "GCP API", "GCN-PP", "Mask-RCNN-Cluster", "Max-DeepLab-Cluster" are two-stage approaches, while "Unified Detector" is an end-to-end unified approach. Our LayoutFormer with

Method	Venue	Line Detection						Word Detection		
		MSRA-TD500			CTW1500			Total-Text		
		R	P	F	R	P	F	R	P	F
TextSnake [23]	ECCV18	73.9	83.2	78.3	67.9	85.3	75.6	74.5	82.7	78.4
MSR [42]	IJCAI19	76.7	87.4	81.7	77.8	83.8	80.7	73.0	85.2	78.6
LOMO [48]	CVPR19	-	-	-	76.5	85.7	80.8	79.3	87.6	83.3
PAN [38]	ICCV19	83.8	84.4	84.1	81.2	86.4	83.7	81.0	89.3	85.0
ContourNet [39]	CVPR20	-	-	-	84.1	83.7	83.9	83.9	86.9	85.4
DB [19]	AAAI20	79.2	91.5	84.9	80.2	86.9	83.4	82.5	87.1	84.7
TextFuseNet [46]	IJCAI20	-	-	-	85.0	85.8	85.4	83.2	87.5	85.3
FCENet [53]	CVPR21	-	-	-	83.4	87.6	85.5	82.5	89.3	85.8
BPNet [49]	ICCV21	80.68	85.40	82.97	81.45	87.81	84.51	84.65	90.27	87.37
PCR [8]	CVPR21	83.5	90.8	87.0	82.3	87.2	84.7	82.0	88.5	85.2
I3CL [9]	IJCV22	-	-	-	84.5	87.4	85.9	83.7	89.2	86.3
FSG [32]	CVPR22	84.8	91.6	88.1	82.4	88.1	85.2	85.7	90.7	88.1
Unified Detector [24]	CVPR23	87.44	88.04	87.70	**87.44**	84.56	85.97	**91.06**	84.96	87.90
DPText-DETR [47]	AAAI23	-	-	-	86.2	**91.7**	**88.8**	86.4	**91.8**	**89.0**
LayoutFormer	-	**88.32**	**91.95**	**90.10**	84.26	88.16	86.17	85.07	89.26	87.12

Table 8. Experimental results of single-level text detection. We set N to 100 on three datasets. On MSRA-TD500 and CTW1500, our LayoutFormer achieves the state-of-the-art performance. On Total-Text, our LayoutFormer achieves competitive results.

Method	N_{line}	N_{para}	Line PQ	Paragraph PQ
GCP API [24]	-	-	56.17	46.33
GCN-PP [24]	384	-	62.23	50.10
Mask-RCNN-Cluster [24]	384	-	62.23	51.67
Max-DeepLab-Cluster [24]	384	-	62.23	52.52
Unified Detector [24]	128	-	58.76	51.48
	256	-	-	52.50
	384	-	62.23	53.60
LayoutFormer	384	200	**62.37**	**53.76**

Table 9. Experimental results of line-paragraph two-level text detection on the HierText testing set. N_{line} denotes the number of text line queries.

N_{line} = 384 outperforms Unified Detector [24] with $N_{line} = 384$ by 0.14% and 0.16% in terms of PQ of text detection and layout analysis, respectively.

For word-line-paragraph detection, there is no any prior research on three-level text detection and layout analysis. We only list our performance on Tab. 4, which can be used by subsequent methods for comparison. Our LayoutFormer achieves 49.24%, 46.21%, and 42.97% in terms of PQ of word-level, line-level, and paragraph-level, respectively. The inference speed of our method is 3.0 FPS.

Single-level Text Detection. In this section, we evaluate the performance of our model on the most widely used benchmarks for single-level scene text detection, i.e., MSRA-TD500 and CTW1500 for line-level, Total-Text for word-level. Results are listed in Tab. 8. The quantitative results demonstrate the advances of our model.

For line detection, we achieve the state-of-the-art results on MSRA-TD500. Notably, our LayoutFormer achieves 88.32%, 91.95%, and 90.10% in terms of Recall, Precision, and F-measure, respectively, which significantly outperforms other methods with a great margin. For example, LayoutFormer outperforms PCR [8], FSG [32], and Unified Detector [24] by 3.10%, 2.00%, 2.40% in terms of F-measure, respectively. On CTW1500, LayoutFormer achieves 84.26%, 88.16%, and 86.17% in terms of Recall, Precision, and F-measure, respectively, achieving the promising performance.

For word detection, we achieve competitive results on Total-Text. The performance of our LayoutFormer is slightly lower than FSG [32] and Unified Detector [24], in which the former utilizes more training epochs and the latter utilizes larger training datasets and more training epochs. DPText-DETR [47] pre-trains their models on more datasets and these datasets are specifically for improving the performance of arbitrary-shape texts.

5. Conclusion

In this work, we propose a hierarchical text detector for promoting scene text understanding, which simultaneously detects multi-level texts and predicts their geometric layouts. In LayoutFormer, we propose WordDecoder, LineDecoder, and ParaDecoder to detect word-level text, line-level text, and paragraph-level text, respectively. WordDecoder and ParaDecoder adaptively learn word-line relationships and line-paragraph relationships, respectively. Through this research, we hope to provide a baseline for hierarchical text detection and inspire other hierarchical tasks. In addition, we will continue to promote and improve the hierarchical architecture for scene text understanding.

References

[1] Ali Furkan Biten, Rubèn Tito, Andrés Mafla, Lluís Gómez i Bigorda, Marçal Rusiñol, C. V. Jawahar, Ernest Valveny, and Dimosthenis Karatzas. Scene text visual question answering. In *ICCV*, pages 4290–4300, 2019. 1

[2] Samuele Capobianco, Leonardo Scommegna, and Simone Marinai. Historical handwritten document segmentation by using a weighted loss. In *ANNPR*, pages 395–406, 2018. 3

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020. 2

[4] Praneeth Chakravarthula, Jim Aldon D'Souza, Ethan Tseng, Joe Bartusek, and Felix Heide. Seeing with sound: Long-range acoustic beamforming for multimodal scene understanding. In *CVPR*, pages 982–991, 2023. 1

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):834–848, 2018. 3

[6] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, pages 1280–1289, 2022. 2, 4, 6

[7] Chee-Kheng Ch'ng, Chee Seng Chan, and Cheng-Lin Liu. Total-text: toward orientation robustness in scene text detection. *IJDAR*, 23(1):31–52, 2020. 6

[8] Pengwen Dai, Sanyi Zhang, Hua Zhang, and Xiaochun Cao. Progressive contour regression for arbitrary-shape scene text detection. In *CVPR*, pages 7393–7402, 2021. 8

[9] Bo Du, Jian Ye, Jing Zhang, Juhua Liu, and Dacheng Tao. I3CL: intra- and inter-instance collaborative learning for arbitrary-shaped scene text detection. *IJCV*, 130(8):1961–1977, 2022. 8

[10] Ziteng Gao, Limin Wang, Bing Han, and Sheng Guo. Adamixer: A fast-converging query-based object detector. In *CVPR*, pages 5354–5363, 2022. 2, 4

[11] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *CVPR*, pages 2315–2324, 2016. 6

[12] Fei He, Haoyang Zhang, Naiyu Gao, Jian Jia, Yanhu Shan, Xin Zhao, and Kaiqi Huang. Inspro: Propagating instance query and proposal for online video instance segmentation. 2022. 3

[13] Junjie He, Pengyu Li, Yifeng Geng, and Xuansong Xie. Fastinst: A simple query-based model for real-time instance segmentation. In *CVPR*, pages 23663–23672, 2023. 4

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 4

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *ICCV*, pages 2980–2988, 2017. 3

[16] Joonho Lee, Hideaki Hayashi, Wataru Ohyama, and Seiichi Uchida. Page segmentation using a convolutional neural network with trainable co-occurrence features. In *ICDAR*, pages 1023–1028, 2019. 1, 3

[17] Kai Li, Curtis Wigington, Chris Tensmeyer, Handong Zhao, Nikolaos Barmpalios, Vlad I. Morariu, Varun Manjunatha, Tong Sun, and Yun Fu. Cross-domain document object detection: Benchmark suite and method. In *CVPR*, pages 12912–12921, 2020. 3

[18] Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, and Wenyu Liu. Textboxes: A fast text detector with a single deep neural network. In *AAAI*, pages 4161–4167, 2017. 2

[19] Minghui Liao, Zhaoyi Wan, Cong Yao, Kai Chen, and Xiang Bai. Real-time scene text detection with differentiable binarization. In *AAAI*, pages 11474–11481, 2020. 2, 8

[20] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017. 4

[21] Yuliang Liu, Lianwen Jin, Shuaitao Zhang, Canjie Luo, and Sheng Zhang. Curved scene text detection via transverse and longitudinal sequence connection. *PR*, 90:337–345, 2019. 6

[22] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. 3

[23] Shangbang Long, Jiaqiang Ruan, Wenjie Zhang, Xin He, Wenhao Wu, and Cong Yao. Textsnake: A flexible representation for detecting text of arbitrary shapes. In *ECCV*, pages 19–35, 2018. 2, 8

[24] 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 *CVPR*, pages 1039–1049, 2022. 2, 3, 5, 6, 7, 8

[25] Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. ICDAR 2023 competition on hierarchical text detection and recognition. In *ICDAR*, pages 483–497, 2023. 1

[26] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 6

[27] Devashish Prasad, Ayan Gadpal, Kshitij Kapadni, Manish Visave, and Kavita Sultanpure. Cascadetabnet: An approach for end to end table detection and structure recognition from image-based documents. In *CVPR*, pages 2439–2447, 2020. 3

[28] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. pages 91–99, 2015. 3, 4

[29] Baoguang Shi, Xiang Bai, and Serge J. Belongie. Detecting oriented text in natural images by linking segments. In *CVPR*, pages 3482–3490, 2017. 2

[30] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *CVPR*, pages 8317–8326, 2019. 1

[31] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris Kitani. Rethinking transformer-based set prediction for object detection. In *ICCV*, pages 3591–3600, 2021. 2

[32] Jingqun Tang, Wenqing Zhang, Hongye Liu, Mingkun Yang, Bo Jiang, Guanglong Hu, and Xiang Bai. Few could be bet-

ter than all: Feature sampling and grouping for scene text detection. In *CVPR*, pages 4553–4562, 2022. 1, 2, 8

[33] Zhi Tian, Weilin Huang, Tong He, Pan He, and Yu Qiao. Detecting text in natural image with connectionist text proposal network. In *ECCV*, pages 56–72, 2016. 2

[34] Matheus Palhares Viana and Dário Augusto Borges Oliveira. Fast cnn-based document layout analysis. In *ICCV*, pages 1173–1180, 2017. 3

[35] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan L. Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, pages 5463–5474, 2021. 2

[36] Renshen Wang, Yasuhisa Fujii, and Ashok C. Popat. Post-ocr paragraph recognition by graph convolutional networks. In *WACV*, pages 2533–2542, 2022. 3

[37] Wenhai Wang, Enze Xie, Xiang Li, Wenbo Hou, Tong Lu, Gang Yu, and Shuai Shao. Shape robust text detection with progressive scale expansion network. In *CVPR*, pages 9336–9345, 2019. 2

[38] Wenhai Wang, Enze Xie, Xiaoge Song, Yuhang Zang, Wenjia Wang, Tong Lu, Gang Yu, and Chunhua Shen. Efficient and accurate arbitrary-shaped text detection with pixel aggregation network. In *ICCV*, pages 8439–8448, 2019. 2, 8

[39] Yuxin Wang, Hongtao Xie, Zheng-Jun Zha, Mengting Xing, Zilong Fu, and Yongdong Zhang. Contournet: Taking a further step toward accurate arbitrary-shaped scene text detection. In *CVPR*, pages 11750–11759, 2020. 2, 8

[40] Guanghui Xu, Shuaicheng Niu, Mingkui Tan, Yucheng Luo, Qing Du, and Qi Wu. Towards accurate text-based image captioning with content diversity exploration. In *CVPR*, pages 12637–12646, 2021. 1

[41] Yue Xu, Fei Yin, Zhaoxiang Zhang, and Cheng-Lin Liu. Multi-task layout analysis for historical handwritten documents using fully convolutional networks. In *IJCAI*, pages 1057–1063, 2018. 3

[42] Chuhui Xue, Shijian Lu, and Wei Zhang. MSR: multi-scale shape regression for scene text detection. In *IJCAI*, pages 989–995, 2019. 8

[43] Chuhui Xue, Jiaxing Huang, Wenqing Zhang, Shijian Lu, Changhu Wang, and Song Bai. Contextual text block detection towards scene text understanding. In *ECCV*, pages 374–391, 2022. 1, 3

[44] 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 *CVPR*, pages 4342–4351, 2017. 1

[45] Cong Yao, Xiang Bai, Wenyu Liu, Yi Ma, and Zhuowen Tu. Detecting texts of arbitrary orientations in natural images. In *CVPR*, pages 1083–1090, 2012. 5

[46] Jian Ye, Zhe Chen, Juhua Liu, and Bo Du. Textfusenet: Scene text detection with richer fused features. In *IJCAI*, pages 516–522, 2020. 8

[47] 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. In *AAAI*, 2023. 1, 2, 8

[48] Chengquan Zhang, Borong Liang, Zuming Huang, Mengyi En, Junyu Han, Errui Ding, and Xinghao Ding. Look more than once: An accurate detector for text of arbitrary shapes. In *CVPR*, pages 10552–10561, 2019. 8

[49] Shi-Xue Zhang, Xiaobin Zhu, Chun Yang, Hongfa Wang, and Xu-Cheng Yin. Adaptive boundary proposal network for arbitrary shape text detection. In *ICCV*, pages 1285–1294, 2021. 2, 8

[50] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. EAST: an efficient and accurate scene text detector. In *CVPR*, pages 2642–2651, 2017. 2

[51] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2

[52] Yiqin Zhu, Jianyong Chen, Lingyu Liang, Zhanghui Kuang, Lianwen Jin, and Wayne Zhang. Fourier contour embedding for arbitrary-shaped text detection. In *CVPR*, pages 3123–3131, 2021. 2

[53] Yiqin Zhu, Jianyong Chen, Lingyu Liang, Zhanghui Kuang, Lianwen Jin, and Wayne Zhang. Fourier contour embedding for arbitrary-shaped text detection. In *CVPR*, pages 3123–3131, 2021. 8