CONTEXTUAL TEXT DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Most existing scene text detectors focus on the detection of characters or words which capture partial textual messages only in most cases due to the missing of contextual information. For a better understanding of text in scenes, it is more desired to detect contextual text blocks which consist of one or multiple integral text units (e.g., characters, words, or phrases) in a specific order, delivering certain complete textual messages. This paper presents Contextual Text Detection, a new setup that detects contextual text blocks for better understanding of texts in scenes. We formulate the new setup by a dual detection task that first detects integral text units and then groups them into a contextual text block. Specifically, we design a novel scene text grouping technique which treats each integral text unit as a token and groups multiple integral tokens belonging to the same contextual text block into an ordered token sequence. To facilitate the future research, we create two new datasets SCUT-CTW-Context and ReCTS-Context where each contextual text block is well annotated by an ordered sequence of integral text units. In addition, we introduce three evaluation metrics that measure contextual text detection in local accuracy, continuity, and global accuracy, respectively. Extensive experiments show that the proposed method detects contextual text blocks effectively. This development including codes, datasets and annotation tools will be published at `http://xxxxxxx`.

1 INTRODUCTION

Scene texts often convey precise and rich semantic information that is very useful to visual recognition and scene understanding. To facilitate reading and understanding by humans, they are usually designed and placed in the form of *contextual text block* which consists of one or multiple integral text units (e.g., a character, word, or phrase). These contextual text blocks deliver complete and meaningful textual messages with not only each of their integral text units but also the specific reading order of these units (i.e., text contexts including spatial adjacency and spatial orderliness).

Most existing scene text detectors (Long et al., 2021; Liao et al., 2020a; Zhang et al., 2021; Liao et al., 2020b) focus on the detection of characters or words as shown in Fig. 1 (the detected text units under *Scene Text Detection*). Due to the missing of contextual information, these detected text units (e.g., a digit in the telephone number in the first sample image or a word in the phrase in the second sample image) usually convey partial textual messages only as compared with contextual text blocks as shown under *Contextual Text Detection* in Fig. 1. Without considering the reading order of the detected text units and grouping them into contextual text blocks, the value of the existing scene text detection is compromised greatly especially while considering the ensuing tasks in scene understanding and natural language processing.

We propose a new scene text detection setup, namely *contextual text detection*, where the objective is to detect *contextual text blocks* (consisting of one or multiple ordered *integral text units*) from images instead of individual integral text units. This detection setup has two challenges. First, it needs to detect and group the integral text units into a contextual text block that delivers a complete text message. Many existing studies (Tian et al., 2015; 2016) adopt a bottom-up approach by first detecting characters (or words) and then grouping them to a word (or a text line). However, their detected texts usually deliver partial textual messages which can not be considered as contextual text blocks, e.g., the text line within a contextual text block in which integral text units lie on multiple lines as illustrated in Fig. 1. Second, it needs to order the detected integral text units (belonging to a contextual text block) according to reading orders. Though some work (Li et al., 2020) studies text

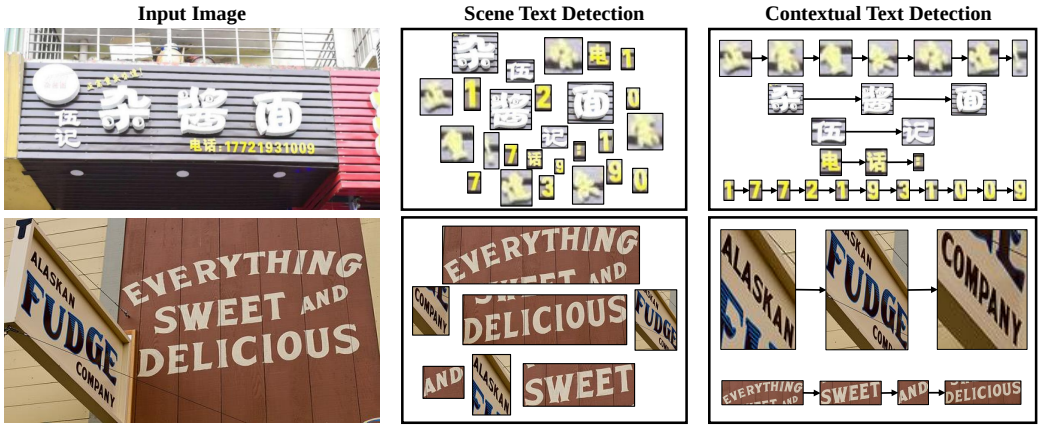

Figure 1: Illustration of Scene Text Detection and Contextual Text Detection problems: Given input images in the first column, scene text detection approaches usually detect integral text units only which contain incomplete textual information as shown in the second column. Differently, the contextual text detection problem aims to contextual text blocks which consist of multiple integral texts in proper reading order.

sequencing in document images, it assumes a single block of text in document images and cannot handle scene images which often contain multiple contextual text blocks with different semantics.

We design a Contextual Text Detector (CUTE) to tackle the contextual text detection problem. CUTE models the grouping and ordering of integral text units from a NLP perspective. Specifically, it extracts visual features and encodes contextual features (representing spatial adjacency and spatial orderliness of integral texts) of all detected text units within a scene text image. These features are transformed into feature embeddings to produce integral text tokens which finally predicts contextual text blocks. In addition, we create two datasets ReCTS-Context (with characters/digits as integral texts) and SCUT-CTW-Context (with words as integral texts) where each contextual text block is well annotated as illustrated in Fig. 1. For evaluation of contextual text detection, we also introduce three evaluation metrics that measure local accuracy, continuity, and global accuracy, respectively.

The contributions of this work are three-fold. First, we propose contextual text detection, a new scene text detection setup that first detects integral text units and then groups them into a contextual text block that conveys a complete text message. To the best of our knowledge, this is the first work that studies the contextual text detection problem. Second, we design CUTE, a contextual text detector that can detect contextual text blocks effectively. Third, we create two well-annotated datasets on contextual text detection and introduce three evaluation metrics that can measure contextual text detection performance comprehensively from multiple aspects.

2 Related Works

2.1 Scene Text Detection

Recent scene text detection approaches can be broadly classified into two categories. The first category takes a bottom-up approach which first detects text fundamental elements and then groups them into words or text lines. SegLink (Shi et al., 2017; Tang et al., 2019) proposes to detect small segments of text instance and link them together to form text bounding boxes. CRAFT (Baek et al., 2019) instead detects characters and uses an affinity score map to aggregate detected characters.

The second category treats words as one specific type of objects and detects them directly by adapting various generic object detection techniques. These text detectors including EAST (Zhou et al., 2017), TextBoxes++ (Liao et al., 2018a), RRD (Liao et al., 2018b) and PSENet (Wang et al., 2019) direct detect text bounding boxes by using generic object detection or segmentation approaches. Recent works further improve the direct detection or segmentation approaches by using border or counter awareness (Xue et al., 2018; Wang et al., 2020; Zhu et al., 2021; Dai et al., 2021), local

refinement (Zhang et al., 2019a; He et al., 2021), deformation convolution (Wang et al., 2018; Xiao et al., 2020), Bezier curve (Liu et al., 2020b) and so on.

These approaches achieve remarkable performances on scene text detection task. However, they are designed to detect individual text units like characters or words while the contextual information is missing. Differently, we propose a new problem setting and a novel method that aims to detect integral text units and group them into contextual text blocks delivering complete text messages.

2.2 SEQUENCE MODELING

Sequence modeling has been widely studied in the field of NLP. Seq2Seq (Sutskever et al., 2014) proposes an encoder-decoder structure for sequential natural language processing by using Recurrent Neural Network (RNN) (Rumelhart et al., 1985). Attention mechanisms (Bahdanau et al., 2014; Luong et al., 2015) is further introduced to relate different positions of a single sequence in order to compute a representation of the sequence. More recently, the advanced Transformer (Vaswani et al., 2017) is proposed which relies entirely on self-attention to compute representations of the input and output without using sequence-aligned RNNs or convolution.

On the other hand, sequence modeling is also applied in the field of computer vision. RNNs (Shi et al., 2016; Su & Lu, 2014) or Transformers (Yu et al., 2020; Xue et al., 2021a) have been widely used in recent scene text recognition approaches because most of the texts are placed in sequential way in images. Besides, some works study the visual permutation for Jigsaw puzzle (Santa Cruz et al., 2017; Noroozi & Favaro, 2016). With the recent advances in Transformers, some works try to model different computer vision task sequentially such as image recognition (Dosovitskiy et al., 2020), object detection (Carion et al., 2020), etc.. More recently, Li et al. (2020) and Wang et al. (2021) propose to learn the text sequence in document analysis by using Graph Convolution Network (GCN) (Kipf & Welling, 2017).

We propose a contextual text detector which detects integral texts and groups them into contextual text blocks by attention mechanism. Different from existing work, the proposed CUTE can detect multiple contextual text blocks that convey different textual messages in one image.

3 PROBLEM DEFINITION

In this section, we formalise the definition of terminologies in contextual text detection problem.

Integral Text Unit: We define the basic detection units as integral text units which are usually integral components of a contextual text block. These units vary from characters, words to text lines, depending on different real-world scenarios and applications. In contextual text detection problem, each integral text unit in image $\boldsymbol{I} \in \mathbb{R}^{3\times H\times W}$ is localized using a bounding box $\boldsymbol{t}$ by:

$$\boldsymbol{t} = (\boldsymbol{p}_0, \boldsymbol{p}_1, ..., \boldsymbol{p}_{k-1}), \quad \boldsymbol{p}_i = (x_i, y_i), \quad x_i \in [0, W-1], y_i \in [0, H-1], \tag{1}$$

where k is the number of vertices in bounding boxes. k may vary depending on different shapes of bounding boxes.

Contextual Text Block: The contextual text block is defined as a set of integral text units arranged in the reading order. They deliver complete textual messages which can be words, phrases, sentences, or paragraphs, depending on real scenarios. The units in one contextual text block may lie on different lines.

Each contextual text block $\boldsymbol{c}$ is defined by:

$$\boldsymbol{c} = (\boldsymbol{t}_0, \boldsymbol{t}_1, ..., \boldsymbol{t}_{m-1}), \tag{2}$$

where m is the number of integral text units in C.

Contextual Text Detection: Given an input image $\boldsymbol{I} \in \mathbb{R}^{3\times H\times W}$, the contextual text detection setup considers a model f that is trained to predict a set of contextual text blocks by :

$$\boldsymbol{C} = f(\boldsymbol{I}), \quad \boldsymbol{C} = \{\boldsymbol{c}_0, \boldsymbol{c}_1, ..., \boldsymbol{c}_{n-1}\}, \tag{3}$$

where n refers to the number of contextual text block in image $\boldsymbol{I}$.

4 METHOD

We propose a network CUTE for contextual text detection which consists an *Integral Text Detector*, an *Integral Embedding Extractor* and a *Contextual Text Block Generator* as illustrated in Fig. 2. The *Integral Text Detector* first localizes a set of integral text units from input images. The *Integral Embedding Extractor* hence learns visual and contextual feature embeddings for each detected integral text unit. In final, the *Contextual Text Block Generator* groups and arranges the detected integral texts in reading order to produce contextual text blocks.

4.1 INTEGRAL TEXT DETECTOR

We adopt Transformer-based generic object detector (Carion et al., 2020) as the integral text detector in our CUTE which is built upon CNN and Transformer architecture. Given an input image $\boldsymbol{I} \in \mathbb{R}^{3\times H\times W}$, the DETR first extracts image features $\boldsymbol{x} \in \mathbb{R}^{3\times H_0\times W_0}$ by using a CNN backbone (e.g., ResNet (He et al., 2016)). A Transformer hence predicts bounding boxes $\boldsymbol{t}$ (in Equation 1) of integral text units from the extracted features $\boldsymbol{x}$. More details about the integral text detector are available in Appendix.

4.2 INTEGRAL EMBEDDING EXTRACTOR

Both visual and contextual features of integral text units are indispensable to accurate detection of contextual text blocks. We therefore design an Integral Embedding Extractor to extract three types of embeddings for each integral text unit including: (1) feature embeddings that are learnt from visual features of integral text units; (2) indexing embeddings that are encoded for integral ordering; (3) spatial embeddings that are predicted from spatial features of integral text units.

Feature Embeddings: We first extract visual features of integral text units and predict a set of feature embeddings. Given the image features x that are extracted from backbone network, the feature embeddings of the integral text units $\boldsymbol{v}_{fe} \in \mathbb{R}^{r\times d}$ are defined by:

$$\boldsymbol{v}_{fe} = (\boldsymbol{v}^0_{fe}, \boldsymbol{v}^1_{fe}, ..., \boldsymbol{v}^{r-1}_{fe}), \quad \boldsymbol{v}^i_{fe} = \boldsymbol{x}^i_c W + b, \quad \boldsymbol{x}^i_c = \text{flatten}(\text{crop}(\boldsymbol{x}, \boldsymbol{t}_i)). \tag{4}$$

Specifically, we first crop the visual features $\boldsymbol{x}_c$ for each of detected integral text units from the image features $\boldsymbol{x}$ by using the detected integral text boxes $\boldsymbol{t}$ from integral text detector. These features $\boldsymbol{x}_c$ are hence flattened and linearly projected to dimension d to produce feature embeddings $\boldsymbol{v}_{fe}$, where r is the number of detected integral text units in image. More details about dimension d are available in Appendix.

Indexing Embeddings: We also introduce indexing embeddings for integral text ordering. Given a set of detected integral text units, we assign each integral text unit with an index number i, where $i \in [0, r-1]$ refers to the i-th integral text unit. Next, we adopt sinusoidal positional encoding (Vaswani et al., 2017) on these indices to produce indexing embeddings $\boldsymbol{v}_{ie} \in \mathbb{R}^{r\times d}$ by:

$$\boldsymbol{v}_{ie} = (\boldsymbol{v}^0_{ie}, \boldsymbol{v}^1_{ie}, ..., \boldsymbol{v}^{r-1}_{ie}), \quad \boldsymbol{v}^i_{ie} = \begin{cases} sin(i/10000^{2d_k/d}), & \text{if } d_k = 2n, \\ cos(i/10000^{2d_k/d}), & \text{if } d_k = 2n+1. \end{cases} \tag{5}$$

Spatial Embeddings: The spatial information of each detected integral text unit (i.e., size and position of integral texts in images) are lost because integral text features are extracted by cropping and resizing. For accurate contextual text block detection, we introduce spatial embeddings that encodes the spatial information to each integral text unit. Specifically, we use a vector $\boldsymbol{v}^i_s$ to represent the spatial information of i-th integral text unit which is defined by:

$$\boldsymbol{v}^i_s = (w, h, x_1, y_1, x_2, y_2, w \times h), \tag{6}$$

where w, h, (x_1, y_1) and (x_2, y_2) refer to the width, height, top-left vertex coordinate, and bottom-right vertex coordinate of integral text bounding box t^i. The spatial embeddings $\boldsymbol{v}_{se} \in \mathbb{R}^{r\times d}$ are hence obtained by two linear transformations:

$$\boldsymbol{v}_{se} = (\boldsymbol{v}^0_{se}, \boldsymbol{v}^1_{se}, ..., \boldsymbol{v}^{r-1}_{se}), \quad \boldsymbol{v}^i_{se} = max(0, max(0, \boldsymbol{v}^i_s W_1 + b_1)W_2 + b_2). \tag{7}$$

The text tokens are hence obtained by:

$$\boldsymbol{v}_{token} = \text{Concat}(\boldsymbol{v}_{fe}, \boldsymbol{v}_{ie}, \boldsymbol{v}_{se}). \tag{8}$$

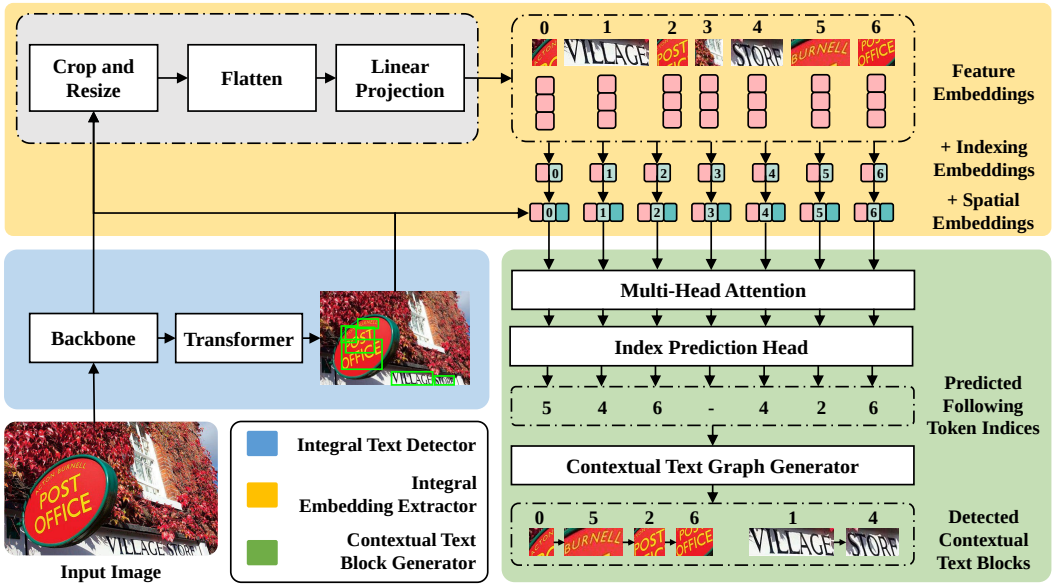

Figure 2: The framework of the proposed contextual text detector (CUTE): Given a scene text image as input, CUTE first detects integral text units with an *Integral Text Detector*. For each detected integral text unit, it then learns textual *Feature Embeddings*, *Indexing Embeddings* and *Spatial Embeddings* that capture visual text features, text order features, and text spatial adjacency features, respectively. In final, it models the relationship of integral text units by learning from the three types of embeddings with a *Contextual Text Block Generator* and produces contextual text blocks that convey complete text messages.

4.3 Contextual Text Block Generator

Taking the integral tokens $\boldsymbol{v}_{token}$ as input, the Contextual Text Block Generator groups and arranges these integral tokens in reading order. As illustrated in Fig. 2, it learns the relationship between each pairs of integral tokens $\boldsymbol{v}_{token}$ by a multi-head attention layer and produces contextual text blocks by an index prediction head and a contextual text graph generator.

Multi-Head Attention: We use multi-head self-attention mechanism to model the relationships between each pair of integral text units. Six stacked attention modules are adopted and each of them contains a multi-head self-attention layer following by a linear transformation layer. Layer normalization (Ba et al., 2016) is applied to the input of each layer. The text tokens $\boldsymbol{v}_{token}$ serve as values, keys, and queries of the attention function.

Index Prediction Head: We model the contextual information learning as an index classification problem by an index prediction head. Specifically, a linear projection layer is adopted to predict a set of indices $i = (i^0, i^1, ..., i^{r-1})$, where $\boldsymbol{v}^{i^k}_{token}$ follows $\boldsymbol{v}^{k}_{token}$ in reading order.

For the i-th indexed query token $\boldsymbol{v}^{i}_{token}$, three cases are considered including: (1) if $\boldsymbol{v}^{i}_{token}$ is not the last integral text in a contextual block, the index prediction head outputs index class j if $\boldsymbol{v}^{j}_{token}$ follows $\boldsymbol{v}^{i}_{token}$; (2) if $\boldsymbol{v}^{i}_{token}$ is the last integral unit in a contextual block, the class i will be predicted; (3) if v^{i}_{token} is not a text (i.e., false alarms from Integral Text Detector), it will be classified to 'not a text' class. In this way, a $(N+1)$-way classification problem is defined for index prediction where 'N' refers the number of index categories and '$+1$' is the 'not a text' category. 'N' is a fixed number that is significantly larger than the possible number of integral text units in an image.

Contextual Text Graph Generator: A directed contextual text graph $G = (V, E)$ is constructed by a Contextual Text Graph Generator which considers the detected integral text units as vertices V. The E refers to the edges of the graph G that is obtained from the Index Prediction Head (IPH) by $E = \{(V_i, V_j)|\text{IPH}(\boldsymbol{v}^{i}_{token}) = j, i \in |V|, j \in |V|\}$. A set of weakly connected components $G' = \{G'_0, G'_1, ...G'_n\}$ are produced from graph G where n refers to the number of contextual text

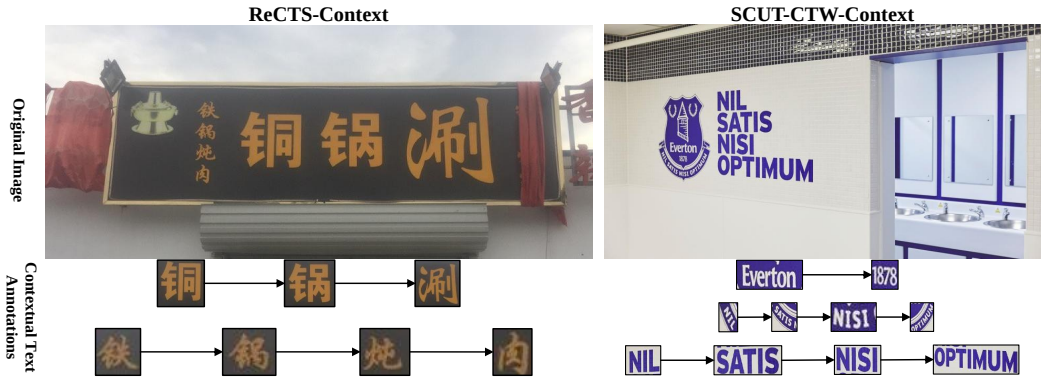

Figure 3: Illustration of contextual text block annotation: We annotate each contextual text block by an ordered sequence of integral text units (characters or words) which together convey a complete text message. The two sample images are picked from datasets ReCTS and SCUT-CTW.

blocks in the image. Each $G'_i = (V'_i, E'_i)$ represents a contextual text block in image where V'_i refers its integral text units and E'_i produces their reading order.

5 Datasets and Evaluation Metrics

5.1 Datasets

We create two contextual-text-block datasets ReCTS-Context and SCUT-CTW-Context as shown in Table 1. Fig. 3 shows two sample images where integral text units belonging to the same contextual text block are grouped in proper order.

ReCTS-Context (ReCTS): We annotate contextual text blocks for images in dataset ICDAR2019-ReCTS (Zhang et al., 2019b), which are split into a training set and a test set with 15,000 and 5,000 images, respectively. It contains largely Chinese texts with characters as integral text units.

SCUT-CTW-Context (SCUT-CTW): We annotate contextual text blocks for dataset SCUT-CTW-1500 dataset (Yuliang et al., 2017) which consists of 940 training images and 498 test images. Most integral text units in this dataset are words which have rich contextual information as captured in various scenes. More details about the two created datasets are available in Appendix.

Table 1: The statistics of the ReCTS-Context and SCUT-CTW-Context datasets: 'integral': Integral Text Units; 'block': Contextual Text Blocks; '#': Number.

Dataset	Integral Text	# integral	# block	# image	# integral per block	# integral per image	# block per image
ReCTS	Character	440,027	107,754	20,000	4.08	22.00	5.39
SCUT-CTW	Word	25,208	4,512	1,438	5.56	17.65	3.17

5.2 Evaluation Metrics

We propose three evaluation metrics for the evaluation of contextual text detection:

Local Accuracy (LA): We introduce LA to evaluate the accuracy of order prediction for neighbouring integral text units. Considering two correctly detected integral text units a and b (with b following a as ground-truth), a true positive is counted if the detection box of b is predicted as directly following that of a. We compute LA by $LA = TP/N$ where TP denotes the number of true positives and N is the total number of connected pairs in ground-truth.

Local Continuity (LC): We introduce LC to evaluate the continuity of integral text units by computing a modified n-gram precision score as inspired by BLEU (Papineni et al., 2002). Specifically, we

Table 2: Quantitative comparison of CUTE with state-of-the-art methods on ReCTS-Context. LA: Local Accuracy; LC: Local Continuity; GA: Global Accuracy.

Model	IoU=0.5			IoU=0.75			IoU=0.5:0.05:0.95		
	LA	LC	GA	LA	LC	GA	LA	LC	GA
CLUSTERING (Cheng, 1995)	32.22	19.06	10.59	26.06	17.01	9.66	25.60	16.13	9.02
CRAFT-R50 (Baek et al., 2019)	63.66	53.26	45.96	51.22	48.39	36.76	50.06	45.46	35.60
LINK-R50 (Xue et al., 2021b)	68.15	57.50	48.39	53.83	50.19	38.36	52.95	47.69	37.33
CUTE-R50	**70.43**	**64.74**	**51.55**	**54.39**	**56.63**	**39.52**	**53.92**	**53.56**	**38.92**
CRAFT-R101 (Baek et al., 2019)	65.21	54.59	47.02	52.01	48.65	37.21	51.56	46.10	36.33
LINK-R101 (Xue et al., 2021b)	70.78	59.10	49.92	54.53	51.02	38.98	53.42	48.26	37.94
CUTE-R101	**72.36**	**67.33**	**53.76**	**55.14**	**57.03**	**40.21**	**54.56**	**53.94**	**39.42**

Table 3: Quantitative comparison of CUTE with state-of-the-art methods on SCUT-CTW-Context. LA: Local Accuracy; LC: Local Continuity; GA: Global Accuracy.

Model	IoU=0.5			IoU=0.75			IoU=0.5:0.05:0.95		
	LA	LC	GA	LA	LC	GA	LA	LC	GA
CLUSTERING (Cheng, 1995)	18.36	7.93	6.78	14.11	5.88	4.72	13.54	5.71	4.88
LINK-R50 (Xue et al., 2021b)	25.47	3.33	18.88	20.25	3.15	14.70	19.31	2.93	14.26
CUTE-R50	**54.01**	**39.19**	**30.65**	**41.62**	**31.19**	**23.71**	**39.44**	**29.03**	**22.10**
LINK-R101 (Xue et al., 2021b)	25.71	3.41	19.18	20.02	2.89	14.68	19.56	2.72	14.39
CUTE-R101	**55.71**	**39.38**	**32.62**	**40.61**	**29.04**	**22.77**	**39.95**	**28.30**	**22.69**

compare n-grams of the predicted consecutive integral text units with the n-grams of the ground-truth integral texts and count the number of matches, where n varies from 1 to 5. Especially for $n = 1$, we only consider the scenario that the contextual text block contains one integral text.

Global Accuracy (GA): Besides LA and LC which focus on local characteristics of integral text units ordering, we also evaluate the detection accuracy of contextual text blocks. TP is counted if all integral texts in a contextual text block are detected and the reading orders are accurately predicted. The global accuracy is hence computed by $GA = TP/N$ where N is the total number of contextual text blocks in ground-truth.

Besides, a detected integral text unit is determined to be matched with ground-truth text if the intersection-over-union (IoU) of these two bounding boxes are larger than a threshold. We adopt three IoU threshold standards that are widely-used in generic object detection task (Liu et al., 2020a) including $IoU = 0.5, IoU = 0.75$ and $IoU = 0.5 : 0.05 : 0.95$ for thorough evaluation.

6 Experiments

6.1 Comparing with State-of-the-art

Since there is little prior research on contextual text block detection, we develop a few baselines for comparisons. The first baseline is CLUSTERING that groups integral text units by mean shift clustering (Cheng, 1995). The second and the third baselines are CRAFT (Baek et al., 2019) and LINK (Xue et al., 2021b), two bottom-up scene text detection methods that group characters/words to text lines. Since both CRAFT and LINK do not have the concept of contextual text blocks, we sort integral text units within each contextual text block according to the common reading order of left-to-right and top-to-down. In addition, we evaluate with two backbones ResNet-50 and ResNet-101 (denoted by 'R50' and 'R101') to study the robustness of the proposed CUTE.

We compare CUTE with the three baselines over ReCTS where integral text units are at character level. As Table 2 shows, the clustering-based method cannot solve the contextual text detection problem effectively because the integral text units are usually with different sizes, positions, and orientations. The bottom-up scene text detectors work better by focusing on visual features only.

Table 4: Quantitative comparison of CUTE with state-of-the-art methods on integral text grouping and ordering task: The ground-truth integral text bounding boxes are adopted to evaluate different approaches on integral text grouping and ordering task only. LA: Local Accuracy; LC: Local Continuity; GA: Global Accuracy.

Model	SCUT-CTW			ReCTS		
	LA	LC	GA	LA	LC	GA
CLUSTERING (Cheng, 1995)	27.94	12.74	10.76	69.70	49.15	32.20
LINK-R50 (Xue et al., 2021b)	30.17	4.48	22.84	83.77	68.44	61.10
CUTE-R50	**71.48**	**58.53**	**49.67**	**92.08**	**82.79**	**76.02**
LINK-R101 (Xue et al., 2021b)	45.54	6.28	31.69	86.66	75.03	69.55
CUTE-R101	**71.54**	**58.68**	**52.57**	**93.12**	**83.70**	**77.81**

Table 5: Ablation studies of CUTE over SCUT-CTW dataset. LA: Local Accuracy; LC: Local Continuity; GA: Global Accuracy.

Model	Feature Embeddings	Spatial Embeddings	Indexing Embeddings	LA	LC	GA
1	✓			6.86	3.34	1.94
2	✓	✓		8.99	4.56	2.18
3	✓		✓	28.65	25.71	21.89
4	✓	✓	✓	71.48	58.53	49.67

The proposed CUTE performs the best consistently as it models the relation between each pair of integral text units by considering both visual representative features and contextual information.

We further conduct experiments over SCUT-CTW where integral text units are at word level. We compare CUTE with CLUSTERING and LINK only because CRAFT cannot group texts lying on different lines. As Table 3 shows, CLUSTERING achieves very low performance due to the complex contextual relations among integral text units. LINK obtains very low scores on LC, showing that only short contextual text blocks with small number of integral text units are detected. CUTE instead outperforms all three baselines by large margins consistently across LA, LC and GA. Note the detection performances over SCUT-CTW are relatively low because it contains many texts with more complex layouts as compared with ReCTS.

The performance of CUTE depends heavily on the detection of each integral text unit. To validate CUTE's effectiveness on the grouping and ordering of integral text units, we assume that all integral text units are accurately detected by feeding the bounding boxes of ground-truth integral text units to the *Integral Embedding Extractor* and *Contextual Text Block Generator*. As Table 4 shows, the proposed CUTE groups and orders integral text units effectively.

6.2 Ablation Study

The proposed CUTE detects contextual text blocks by using both visual features (i.e., feature embeddings) and contextual features that capture spatial and ordering information in spatial embeddings and indexing embeddings. We conduct an ablation study over SCUT-CTW-Context to identify the contribution of each embedding. We trained four detection models with different combinations of the three types of embeddings. As Table 5 shows, CUTE does not work well with either feature embeddings alone or feature embeddings plus spatial embeddings. However, combining feature embeddings with indexing embeddings improves the detection greatly as indexing embeddings introduce crucial text order information. The combination of all three embeddings outperforms other models by large margins, demonstrating the complementary nature of the three embeddings.

6.3 Discussion

The proposed detection setup for contextual text blocks can facilitate both scene text detection and many downstream tasks. We first study how the proposed contextual text detection can improve

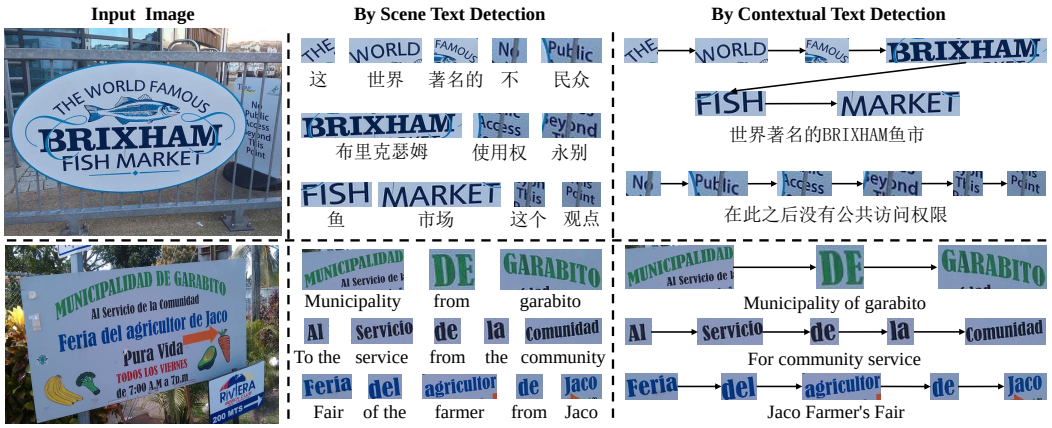

Figure 4: Contextual text detection facilitates scene text translation significantly: The output of CUTE conveys complete text messages which can be better translated to other languages as 'natural language' with rich contextual information as shown in column 3. As a comparison, scene text detectors produce individual text units which can not be translated well as shown in column 2.

Table 6: The significance of contextual text detection to scene text detection task: The proposed CUTE effectively helps to improve scene text detection performance of different detectors by filtering out the false alarms.

Model	PSENet (Wang et al., 2019)	MSR (Xue et al., 2019)	DETR (Carion et al., 2020)	LINK (Xue et al., 2021b)
w/o CUTE	41.20	60.07	47.48	64.20
with CUTE	41.51	61.80	49.53	65.05

the scene text detection task over SCUT-CTW. Specifically, traditional scene text detectors tend to produce false detection at image background that has similar visual representations as scene texts. CUTE can suppress such false detection effectively as it considers not only visual features but also contextual information of texts (details in Section 4.3). As Table 6 shows, CUTE improves the scene text detection performance consistently across a number of scene text detectors that adopt different backbones and detection strategies.

We also study how the proposed contextual text detection can facilitate various downstream tasks. We focus on the scene text translation task that is very useful to scene understanding for visitors with different home languages. Specifically, we feed each detected text (i.e. a character, word, or contextual text block) to a neural machine translator (Google Translator) for translation across different languages. As Fig. 4 shows, CUTE groups and orders scene texts into contextual text blocks (delivering complete text messages) which facilitates scene text translation greatly as compared with traditional scene text detectors without the concept of contextual text blocks.

7 Conclusion and Future Work

We study contextual text detection, a new text detection setup that first detects integral text units and then groups them into contextual text blocks. We design CUTE, a novel method that detects contextual text blocks effectively by combining both visual and contextual features. In addition, we create two contextual text detection datasets within which each contextual text block is well annotated by an ordered text sequence. Extensive experiments show that CUTE achieves superior contextual text detection performance and it also improves scene text detection as well as many downstream tasks greatly. We will continue to study contextual text detection problem when scene texts have complex layouts. Specifically, we will expand and balance our datasets by including more complex scenes and text layouts. We will also study how to leverage text semantics (derived by scene text recognition) for better contextual text detection.

Reproducibility Statement This development including codes, datasets and annotation tools will be published at http://xxxxxxx. The implementation details for reproducing of the proposed method are available in Appendix. The details of the datasets used in the experiments are provided in main manucript and Appendix.

REFERENCES

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

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*, pp. 9365–9374, 2019.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. ISSN 2307-387X.

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*, pp. 213–229. Springer, 2020.

Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799, 1995.

Pengwen Dai, Sanyi Zhang, Hua Zhang, and Xiaochun Cao. Progressive contour regression for arbitrary-shape scene text detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7393–7402, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Minghang He, Minghui Liao, Zhibo Yang, Humen Zhong, Jun Tang, Wenqing Cheng, Cong Yao, Yongpan Wang, and Xiang Bai. Most: A multi-oriented scene text detector with localization refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8813–8822, 2021.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1181. URL `http://www.aclweb.org/anthology/D14-1181`.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Liangcheng Li, Feiyu Gao, Jiajun Bu, Yongpan Wang, Zhi Yu, and Qi Zheng. An end-to-end ocr text re-organization sequence learning for rich-text detail image comprehension. In *European Conference on Computer Vision*, pp. 85–100. Springer, 2020.

Minghui Liao, Baoguang Shi, and Xiang Bai. Textboxes++: A single-shot oriented scene text detector. *IEEE transactions on image processing*, 27(8):3676–3690, 2018a.

Minghui Liao, Zhen Zhu, Baoguang Shi, Gui-song Xia, and Xiang Bai. Rotation-sensitive regression for oriented scene text detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5909–5918, 2018b.

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*, pp. 706–722. Springer, 2020a.

Minghui Liao, Zhaoyi Wan, Cong Yao, Kai Chen, and Xiang Bai. Real-time scene text detection with differentiable binarization. In *Proc. AAAI*, 2020b.

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

Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020a.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*, 2016.

Yuliang Liu, Hao Chen, Chunhua Shen, Tong He, Lianwen Jin, and Liangwei Wang. Abcnet: Real-time scene text spotting with adaptive bezier-curve network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9809–9818, 2020b.

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.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pp. 69–84. Springer, 2016.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

Rodrigo Santa Cruz, Basura Fernando, Anoop Cherian, and Stephen Gould. Deeppermnet: Visual permutation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3949–3957, 2017.

Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2016.

Baoguang Shi, Xiang Bai, and Serge Belongie. Detecting oriented text in natural images by linking segments. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Bolan Su and Shijian Lu. Accurate scene text recognition based on recurrent neural network. In *Asian Conference on Computer Vision*, pp. 35–48. Springer, 2014.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Jun Tang, Zhibo Yang, Yongpan Wang, Qi Zheng, Yongchao Xu, and Xiang Bai. Seglink++: Detecting dense and arbitrary-shaped scene text by instance-aware component grouping. *Pattern recognition*, 96:106954, 2019.

Shangxuan Tian, Yifeng Pan, Chang Huang, Shijian Lu, Kai Yu, and Chew Lim Tan. Text flow: A unified text detection system in natural scene images. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4651–4659, 2015.

Zhi Tian, Weilin Huang, Tong He, Pan He, and Yu Qiao. Detecting text in natural image with connectionist text proposal network. In *European Conference on Computer Vision*, pp. 56–72. Springer, 2016.

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*, pp. 5998–6008, 2017.

Fangfang Wang, Liming Zhao, Xi Li, Xinchao Wang, and Dacheng Tao. Geometry-aware scene text detection with instance transformation network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1381–1389, 2018.

Renshen Wang, Yasuhisa Fujii, and Ashok C Popat. General-purpose ocr paragraph identification by graph convolutional neural networks. *arXiv preprint arXiv:2101.12741*, 2021.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9336–9345, 2019.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11753–11762, 2020.

Shanyu Xiao, Liangrui Peng, Ruijie Yan, Keyu An, Gang Yao, and Jaesik Min. Sequential deformation for accurate scene text detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pp. 108–124. Springer, 2020.

Chuhui Xue, Shijian Lu, and Fangneng Zhan. Accurate scene text detection through border semantics awareness and bootstrapping. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 355–372, 2018.

Chuhui Xue, Shijian Lu, and Wei Zhang. Msr: Multi-scale shape regression for scene text detection. *arXiv preprint arXiv:1901.02596*, 2019.

Chuhui Xue, Shijian Lu, Song Bai, Wenqing Zhang, and Changhu Wang. I2c2w: Image-to-character-to-word transformers for accurate scene text recognition. *arXiv preprint arXiv:2105.08383*, 2021a.

Chuhui Xue, Shijian Lu, and Steven Hoi. Detection and rectification of arbitrary shaped scene texts by using text keypoints and links. *arXiv preprint arXiv:2103.00785*, 2021b.

Deli Yu, Xuan Li, Chengquan Zhang, Tao Liu, Junyu Han, Jingtuo Liu, and Errui Ding. Towards accurate scene text recognition with semantic reasoning networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

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.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10552–10561, 2019a.

Rui Zhang, Yongsheng Zhou, Qianyi Jiang, Qi Song, Nan Li, Kai Zhou, Lei Wang, Dong Wang, Minghui Liao, Mingkun Yang, et al. Icdar 2019 robust reading challenge on reading chinese text on signboard. In *2019 international conference on document analysis and recognition (ICDAR)*, pp. 1577–1581. IEEE, 2019b.

Wenqing Zhang, Yang Qiu, Minghui Liao, Rui Zhang, Xiaolin Wei, and Xiang Bai. Scene text detection with scribble line. In *International Conference on Document Analysis and Recognition*, pp. 79–94. Springer, 2021.

Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. East: An efficient and accurate scene text detector. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Yiqin Zhu, Jianyong Chen, Lingyu Liang, Zhanghui Kuang, Lianwen Jin, and Wayne Zhang. Fourier contour embedding for arbitrary-shaped text detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3123–3131, 2021.

A Appendix

A.1 Attention Preliminaries

A.1.1 Attention Layers:

We adopt the attention layers (Vaswani et al., 2017) which are defined by:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V,$$

where Q, K and V refer to input ***Queries***, ***Keys*** and ***Values***, respectively. d_k is the dimension of the Q and K.

A.1.2 Multi-Head Attention Layers:

The multi-head attention layer (Vaswani et al., 2017) linearly projects queries, keys and values M times by using different learnt linear projections. It is a concatenation of several single attention heads which is defined by:

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_M)W^O,$$
$$\text{where } head_i = Attention(QW_i^Q, KW_i^K, VW_i^V),$$

where W_i^Q, W_i^K and W_i^V are learnt projections of queries, keys and values in head i, respectively.

A.2 Dataset Details

Besides contextual text annotation, we additionally label each integral to 'Normal', 'Hard' or 'Ignore' category. Mostly, an integral text unit is labelled with 'Normal' if it belongs to a contextual text block. In some special cases, a contextual text blocks (e.g. 'POPYEYES CHICKEN & BISCUITS in Fig. 5) can be split into multiple sub-blocks (e.g. 'POPYEYES' and 'CHICKEN & BISCUITS' in Fig. 5) where each sub-block conveys a textual message and is significant to be an independent contextual text block. Here the last integral text (e.g. 'POPYEYES' in Fig. 5) in each sub-block (except the last sub-block) is label to 'Hard'. Furthermore, if an integral text or its reading order is hardly recognized, it will be labeled as 'Ignore' (e.g. '®' of the second sample in Fig. 5).

A.3 Implementation Details

The training of the proposed CUTE consists of two stages.

First, we train the integral text detector over ReCTS and SCUT-CTW datasets, respectively which is pre-trained on COCO (Lin et al., 2014) dataset for fast convergence. We optimize the integral text detector by AdamW optimizer with batch size of 8. The learning rate is set to 10^{-5} for backbone and 10^{-4} for transformers. The numbers of the transformer encoder/decoder layers are set to 6 and the number of heads in multi-head attention is set to 8. We apply dropout of 0.1 for every multi-head attention and FFN layers before the normalization layers. For data augmentation, we apply random scaling and cropping on the input images during training. The training loss follows (Carion et al., 2020).

Second, we train the overall CUTE by freezing the parameters in backbone. We optimize the integral embedding extractor and the contextual text block generator by AdamW optimizer with batch size

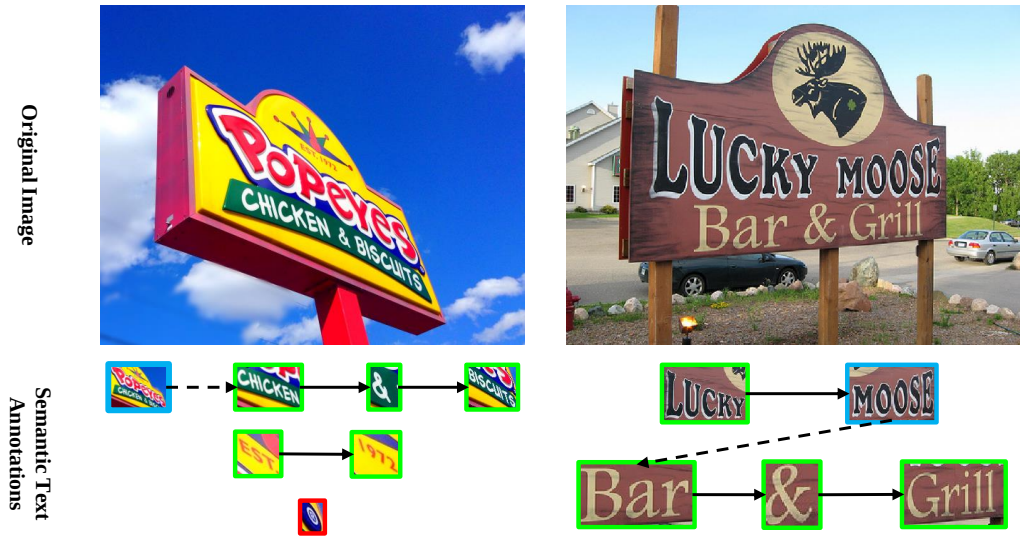

Figure 5: Illustration of the proposed SCUT-CTW-Context dataset with different annotations: Integral texts that belong to 'Normal', 'Hard' or 'Ignore' classes are highlighted in green, blue and red, respectively. In 'Hard' cases, a contextual text block can be split into multiple sub-blocks (links shown in dotted) each of which conveys a significant textual message.

Table 7: Integral text grouping and ordering performance of CUTE by using embeddings with different dimensions.

Dim	#param	LA	LC	GA
64	**47.97M**	70.48	57.19	48.21
128	55.67M	**71.48**	**58.53**	49.67
256	76.37M	70.64	56.38	**50.18**

of 16. The learning rate is set to 10^{-4} with. The number of the mutlti-head attention layer is set to 6 and the number of heads in multi-head attention is set to 8. The dimension for constituent tokens, spatial embeddings and indexing embeddings is set to 128 and the maximum number of indices is 200. For data augmentation, we apply random scaling on the input images during training. The training loss is defined by:

$$\mathcal{L} = -\frac{1}{r} \cdot \sum_{i=0}^{r-1} y_i \cdot \log(\hat{y_i}), \tag{9}$$

where r, y_i and $\hat{y_i}$ refer to the number of integral text in image, ground-truth indices and predicted indices, respectively.

A.4 Additional Experiments

A.4.1 Hyper-parameters

Two hyper-parameters are adopted which are commonly used in transformer-based applications.

Dimension of Embeddings: We introduce a hyper-parameter d which refers to the dimension of feature, indexing and spatial embeddings. We study the influence of hyper-parameter d on the integral text grouping and ordering task as it doesn't affect the integral detector. As Table 7 shows, the increase on d will lead to improvement on GA but also increase of parameter number consistently. By considering LA, LC, GA and number of parameters, we finally adopt $d = 128$ in our experiment.

Number of Attention Layers: We also introduce a hyper-parameter which refers to the number of attention layers. As Table 8 shows, we both model size and performances vary depending on the

Table 8: Integral text grouping and ordering performance of CUTE by using different numbers of attention layers.

#layers	#param	LA	LC	GA
1	**44.83M**	64.26	50.14	44.78
3	49.16M	70.63	56.76	47.97
6	55.67M	**71.48**	**58.53**	**49.67**
9	62.17M	71.16	56.18	48.19

Table 9: The significance of contextual text detection to text classification task: The proposed CUTE effectively helps to improve text classification performance of different text classifiers by learning from recognized texts in contextual text blocks.

Model	TextCNN (Kim, 2014)	TextRNN (Liu et al., 2016)	Fast Text (Bojanowski et al., 2017)	Transformer (Vaswani et al., 2017)
w/o CUTE	90.56	79.55	90.96	89.69
with CUTE	92.40	87.20	91.82	92.54

number of attention layers. We choose 6 as the layer number in all our experiments by considering both model size and performances.

A.5 Discussion

A.5.1 Problem Significance

We additionally conduct an experiment on downstream text classification task in NLP by using advanced text classification techniques on ReCTS-Context dataset. Specifically, we classify and annotate the transcription of texts in ReCTS-Context images into three categories (i.e., 'Address', 'Phone Number' or 'Restaurant Name') according to the text semantics. We train models by different text classification techniques and test on detected and recognized texts from integral texts (denoted by 'w/o CUTE') and contextual text blocks (denoted by 'with CUTE'), respectively. As shown in Table 9 shows, the use of CUTE helps to improve the text classification performance by using different text classifiers, consistently. More discussions are available in Appendix.

A.5.2 Qualitative results

Fig. 6 demonstrates the qualitative results of the proposed CUTE on ReCTS-Context and SCUT-CTW-Context datasets. As Fig. 6 shows, the proposed CUTE successfully detects the contextual text blocks from input images, demonstrating its effectiveness.

A.5.3 Integral Detectors

We adopt Transformer-based detector as integral detector in our CUTE. One of the major advances of the Transformer-based detector is that the Transformer models all interactions between elements of image features for object detection. Specifically, the feature map $\boldsymbol{x}$ is first flattened to a sequence of elements (i.e., pixels) accompanied with 2D positional embeddings. The Transformer hence focuses on image regions for each object by learning the relationships between each pair of elements in feature map $\boldsymbol{x}$. As such, we adopt Transformer-based detector as the integral text detector in our CUTE for better modelling of element interactions in the visual features from network backbone.

A.5.4 Differences between Indexing Embedding and Positional Encoding

The objectives of indexing embeddings and positional encodings in transformer architecture are different. As the transformer architecture is permutation-invariant, the positional encodings are introduced to retain the crucial sequential information of elements in a sorted sequence. Differently, the indexing embeddings are indexing representations for a set of unsorted integral text units which serves as features for the learning and prediction of integral text ordering (i.e., prediction of indices).

Figure 6: Sample detection results by the proposed CUTE: Given input images from ReCTS-Context and SCUT-CTW-Context datasets, the proposed CUTE successfully detects the contextual text blocks in which each integral texts and their orders are shown by yellow boxes and green arrows, respectively.

Figure 7: Typical failure cases of the proposed CUTE: Correct, incorrect and missing integral texts (or orders) are shown by bounding boxes (or arrows) in yellow, red and blue, respectively.

We adopt the algorithm of positional encoding in our indexing embeddings as they are effective and efficient to convey the features of discrete integers (Vaswani et al., 2017).

A.5.5 Failure Cases

The proposed CUTE usually fails under several typical scenarios. First, if two integral text units (within a contextual text block) are far away from each other as shown in the first sample of Fig. 7. Second, it may fail if the images contain complex text layouts as shown in the second sample of Fig. 7. We are continually working on better contextual text detection approach and the future directions are discussed in Section 7.