CLIPTER: Looking at the Bigger Picture in Scene Text Recognition Supplementary Material

Aviad Aberdam[1*] David Bensaïd[2] Alona Golts[1] Roy Ganz[2†] Oren Nuriel[1]
Royee Tichauer[1] Shai Mazor[1] Ron Litman[1]

[1]AWS AI Labs [2]Technion, Israel

A. Pseudocode Algorithm

The pseudocode for integrating CLIPTER into a recognizer is presented in Algorithm 1. This algorithm outlines the key components of our method, including image encoding, pooling, fusion mechanism, and integration point that divides the recognizer into encoder and decoder. In particular, the algorithm highlights that the image encoding operation is performed only once per image, regardless of its word count, and can be executed in parallel with the detection operation.

Algorithm 1: CLIPTER PyTorch-like pseudocode

```
"""
img: scene image
text_crops: all text images cropped from image
img_encoder: frozen VL image encoder
k: kernel of average pooling
fusion_ca: nn.MultiHeadAttention()
alpha: gated parameter (init as 0)
recog_encdoer, recog_decoder: the recognition
    modules before and after the integration point
"""

# image encoding (in parallel to detection)
with torch.no_grad():
   img_f = img_encoder(img) # (1 + HW, d)
   img_f = [img_f[0], avg_pool2d(img_f[1:], k)]

preds = []
for crop in text_crops:
   # recognizer encoding
   crop_f = recog_encoder(crop)

   # fusion by gated cross attention
   merged_f = fusion_ca(query=crop_f, key=img_f,
       value=img_f)
   c = torch.tanh(alpha)
   fused_f = (1 - c) * crop_f + c * merged_f

   # recognizer decoding
   preds.append(recog_decoder(fused_f))
```

B. Datasets

Our work utilizes a highly-diverse collection of 13 public benchmarks, depicted in Fig. 1 and Fig. 2. Since CLIPTER relies on the whole image together with the cropped words, we use datasets that have recognition and detection annotations, usually intended for the task of end-to-end text spotting. Therefore, we could not utilize some public test sets which contain only full images without localization annotations or cropped words without the full images. To mitigate this, we evaluate our method in these cases on the validation set or part of the training set. Nevertheless, we needed to omit IIIT-5k [16] which contains only cropped text images and CUTE-80 [18] which does not contain end-to-end annotations. Below, we describe our data pre-processing and then, provide details on each dataset.

B.1. Data Pre-Processing

Our work applies the same data filters on all datasets. In particular, we filter out words with the flag of `illegible` and words that have ignore labels, i.e., "#", "##", "###", "####" in general, "." in TextOCR, and "*" in Uber. From the training data, we follow [5] and also exclude text that consists of non-alphanumeric characters, long words that contain more than 25 characters, and vertical text by filtering words with more than two characters that their image height is greater than their image width.

B.2. Dataset Details

Below, we provide general details on each dataset and describe our data split into train, validation, and evaluation sets. A summary of these splits appears in Tab. 1, containing also data sizes. As we work on entire images as well as crops, we perform the splits at the entire image level.

ArT[8] is a dataset of arbitrary-shaped text, collected from the train set of Task 3[1]. The train set is divided into 80% for training, 10% for validation, and 10% for evaluation.

COCO-Text[23] is based on COCO dataset [2], containing text in natural images[3]. We consider the training and validation sets that are published with bounding boxes, and split

*Corresponding author aaberdam@amazon.com.

[1]https://rrc.cvc.uab.es
[2]https://cocodataset.org
[3]https://vision.cornell.edu/se3/coco-text-2

Figure 1: **Datasets Part 1.** We provide examples from each of the datasets used in this work.

Figure 2: **Datasets Part 2.** We provide examples from each of the datasets used in this work.

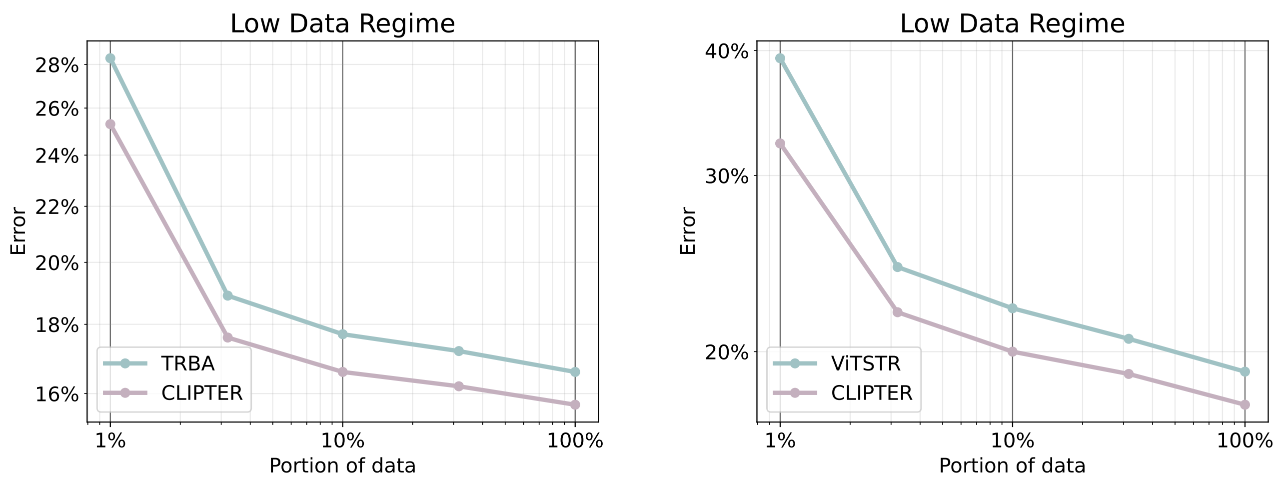

Figure 3: **Low Data Regime – TRBA & ViT-STR.** We evaluate the effect of CLIPTER with limited training data on TRBA [4] (left) and ViTSTR [3] (right). Roughly speaking, adding CLIPTER to these architectures has more impact than doubling the training data amount in terms of reducing the error rate.

the training set into 90% for training and 10% for evaluation.

HierText[15] features hierarchical annotations of text in natural scenes and documents[4]. We consider the training and validation sets which have available bounding boxes, and split the training set into 90% for training and 10% for evaluation. In this dataset, we filtered words that are annotated as vertical.

IC13[14] contains images that are focused around the text content[1]. Since only the training set is provided with full annotations, we use it all for evaluation.

IC15[13] contains incidental scene text and therefore is more challenging[1]. The test set here is the official one, while the training set is divided into 90% for training and 10% for validation.

LSVT[22] contains scene text in street view images[1]. Here, only the training set has full annotations. Therefore, we divide it into 80% for training 10% for validation, and 10% for evaluation.

MLT19[17] is a multilingual dataset[1]. The training set is divided into language subsets, from which we consider English, French, German, and Italian. We split these data into 80% for training, 10% for validation, and 10% for evaluation.

OOV[10] is a new dataset containing out-of-vocabulary scene text[1]. Since this dataset is based on other datasets, we did not use its training set, but use its validation set for evaluation. In this dataset, we filter words that are annotated as non-English or vertical.

	Public E2E Annotations			Number of Words		
	Train.	Valid.	Eval.	Train.	Valid.	Eval.
ArT	✓	✗	✗	25K	2,701	3,667
COCO-Text	✓	✓	✗	51K	13K	5,716
HierText	✓	✓	✗	711K	163K	76K
IC13	✓	✗	✗	–	–	757
IC15	✓	✗	✓	3,741	349	2,077
LSVT	✓	✗	✗	32K	3,937	3,911
MLT19	✓	✗	✗	34K	3,970	4,100
RCTW	✓	✗	✗	7,837	1,017	962
ReCTS	✓	✗	✗	18K	2,331	2,219
SVT	✓	✗	✓	232	24	647
TextOCR	✓	✓	✗	566K	96K	71K
Uber	✓	✓	✓	75K	30K	50K
All				**1,516K**	**316K**	**220K**

Table 1: **Dataset Partition.** Number of cropped word images after pre-processing and splitting into training, validation, and evaluation sets.

RCTW[20] is a dataset for reading Chinese text in images[5]. We split the published training set in 80% for training, 10% for validation and 10% for evaluation.

ReCTS[25] contains Chinese text on signboard[1]. We split the published training set in 80% for training, 10% for validation and 10% for evaluation. In this dataset, we ignore words that are annotated with the flag of `ignore`.

SVT[24] contains street view text in images from Google Street View[6]. Here, we use the official test set and divide the

[4] https://github.com/google-research-datasets/hiertext

[5] https://rctw.vlrlab.net

[6] https://tc11.cvc.uab.es/datasets/SVT_1

CA Model	# Attention Heads	# Hidden Layers	Hidden Size	Intermediate Size	# Parameters
Gated-Attention	–	–	–	–	328K
MH-CA Tiny	2	2	128	512	923K
MH-CA Mini	4	4	256	1,024	5.3M
MH-CA Small	8	4	512	2,048	18.1M

Table 2: **Cross-Attention Model Size.**

training set into 90% for training and 10% for validation.
TextOCR[21] contains high quality images from OpenImages[7] with an average of 30 words per image[8]. Here, we use the published validation set and divide the training set into 90% for training and 10% for evaluation.
Uber[26] contains street-level images collected from car mounted sensors[9]. We keep the original split of training, validation, and evaluation sets.

C. Implementation Details

Multi-head Cross-Attention fusion mechanism. Our implementation of the Multi-Head Cross-Attention (MH-CA) mechanism is based on the implementation of BERT [9, 7] proposed by HuggingFace. Table 2 presents further architectural details.

Training details. Baseline STR models are trained with the hyperparameters published by respective authors. CLIPTER is trained for 20 epochs with a learning rate varying from 1×10^{-5} to 3×10^{-5}. Specifically, gated-attention, MH-CA tiny, mini and small are trained with learning rates of 2×10^{-5}, 3×10^{-5}, 3×10^{-5} and 1×10^{-5} respectively.

D. Low Data Regime

Similarly to analysis performed in the main paper over PARSeq, we evaluate the effect of our method in the low data regimes on TRBA and ViTSTR architectures. As shown in Fig. 3, utilizing CLIPTER on these schemes achieves better results than the baseline model with doubled amount of training data.

E. Latency Analysis

To evaluate the impact of our solution on recognition latency, we conduct end-to-end (E2E) experiments on the ICDAR-15 and Total-Text datasets, and calculate the frames per second (FPS). To this end, we use the ResNet50-based detection model from GLASS [19][10] and exclude their recognition components. Our experiments are conducted on a single V100 NVidia GPU and a simple PyTorch implementation, without any optimizations, such as TensorRT, that could improve the latency results. We calculate the latency using PyTorch benchmarking code[11], with FPS calculated as the average of the median run-time per image. Evaluation metrics are in accordance with the protocol of [19].

F. Additional Experiments

F.1. Synthetic Data

In this part, we aim to analyze the effect of utilizing synthetic data. To this end, we train PARSeq with and without CLIPTER also on the large synthetic datasets of MJ [12] and ST [11]. As shown in Tab. 3, adding the large synthetic data, about 14M images, to the training set only marginally improves the results, indicating on the low impact of synthetic data when there is a lot of real-world data. That said, these datasets do lead to significant improvements on IC13 and IC15. This finding, revealed also in [1, 2], indicates that these datasets mainly represent specific types of natural scenarios.

F.2. Breaking-Down Results on Uber-Text

We utilize Uber-Text [26] word categories to break down the results of PARSeq with and without CLIPTER. As shown in Tab. 4, our method is especially efficient on business name (+1.3%) and street numbers (+1.3%). We believe that these improvements are thanks to the use of a vision-language model that was pretrained also on the textual descriptions of the images, which often contain such information as it is crucial for understanding the scene.

F.3. Dense Documents

We conduct both a quantitative (Figure 5) and qualitative (Figure 4) analysis on the text-dense HierText dataset. The results demonstrate that our model consistently improves accuracy, even in highly text-dense images with over 100 words.

G. Further qualitative analysis

Fig. 6 displays additional examples showcasing benefits of CLIPTER.

[7] https://storage.googleapis.com/openimages/web/index.html
[8] https://textvqa.org/textocr
[9] https://s3-us-west-2.amazonaws.com/uber-common-public/ubertext/index.html
[10] https://github.com/amazon-science/glass-text-spotting
[11] https://pytorch.org/tutorials/recipes/recipes/benchmark.html#pytorch-benchmark

Figure 4: **Quantitative Results on Rich-in-Text Images.** Images with dense text (>100) that benefit from integrating scene-level information using CLIPTER. Green boxes highlight words accurately transcribed by PARSeq+CLIPTER but not by PARSeq, while red boxes indicate the opposite.

	Method	SVT 647	IC13 757	IC15 2,077	COCO 5,716	RCTW 962	Uber 49,561	ArT 3,677	LSVT 3,911	RECTS 2,219	MLT19 4,100	TextOCR 70,597	HierText 75,829	Average 220,053	Weighted Average
Real	PARSeq [6]	96.1	98.9	85.7	80.5	81.4	83.2	91.2	80.2	91.8	91.5	85.2	87.4	87.8	85.6
	+ CLIPTER $_{\text{Vision}}$	96.6	99.1	85.9	81.0	82.1	**84.4**	**91.7**	**81.8**	91.8	91.6	**86.0**	88.0	88.3	**86.4**
	Δ	**+0.5**	**+0.2**	**+0.2**	**+0.5**	**+0.7**	**+1.2**	**+0.5**	**+1.6**	0	**+0.1**	**+0.8**	**+0.6**	**+0.5**	**+0.8**
+ Synth.	PARSeq [6]	97.2	**99.5**	86.4	80.6	82.8	82.1	91.1	80.2	91.9	91.7	85.1	87.5	88.0	85.4
	+ CLIPTER $_{\text{Vision}}$	**97.8**	**99.5**	**86.7**	**81.4**	**83.6**	83.1	91.4	81.3	**92.6**	**92.0**	85.9	**88.4**	**88.6**	86.3
	Δ	**+0.6**	0	**+0.3**	**+0.8**	**+0.8**	**+1.0**	**+0.3**	**+1.1**	**+0.7**	**+0.3**	**+0.8**	**+0.9**	**+0.6**	**+0.9**

Table 3: **Accuracy on Scene Text Benchmarks With and Without using Synthetic Data.** Utilizing the large synthetic datasets of MJ [12] and ST [11] improves performance on the more common benchmarks of SVT, IC13, and IC15. However, the averaged performance across all datasets is marginally better due to the existence of many real-world images.

	Street Number	**Business Name**	**Street Name**	**None**	**Street Number Range**	**Secondary Unit Designator**	**Phone Number**	**Traffic Sign**	**License Plate**
	22,701	14,254	5,885	4,866	1,708	98	32	16	1
Parseq	78.3	85.7	95	82.4	96.3	86.7	50	93.8	0
+ CLIPTER $_{\text{Vision}}$	79.6	87	95.4	83.7	96.5	88.8	46.9	93.8	0
Δ	**+1.3**	**+1.3**	**+0.4**	**+1.3**	**+0.2**	**+2.1**	-3.1	0	0

Table 4: **Accuracy on Uber-Text per Word Category.** The number of words in each category is listed below its name. CLIPTER is mostly effective on street numbers and business names, often critical information for scene understanding.

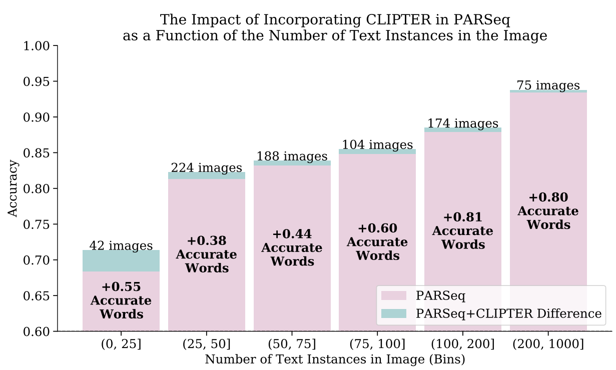

Figure 5: **Enhancing Performance in Dense-Text Images.** This figure illustrates the averaged improvement in accuracy and the number of accurately transcribed words relative to the total number of words in the image. Our algorithm demonstrates remarkable success even in densely-packed text images.

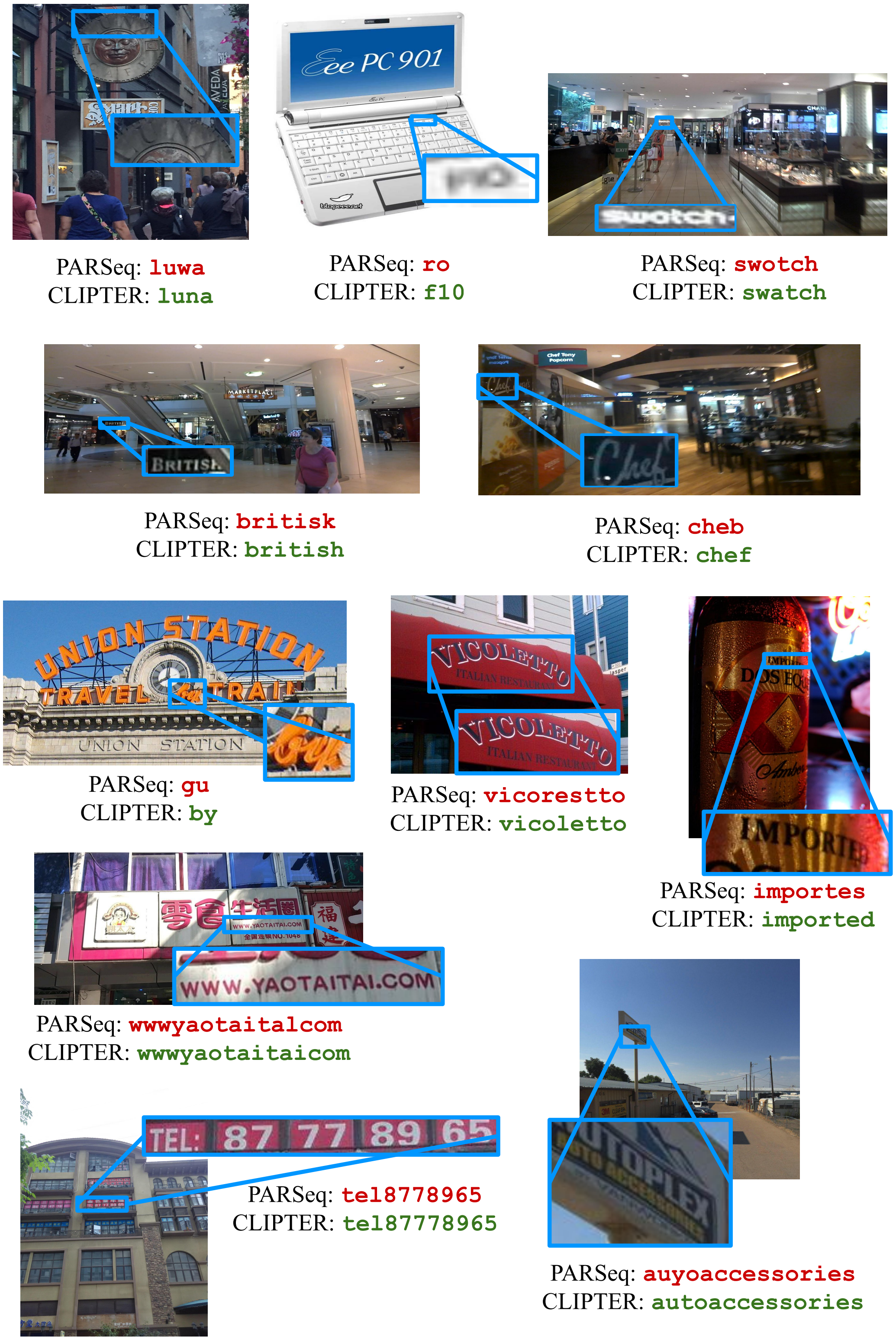

Figure 6: **Positive flips.** Examples in which CLIPTER corrected the prediction of PARSeq and matched the GT annotation.

Figure 7: **Negative flips.** Examples in which CLIPTER harmed the prediction of PARSeq which previously matched the GT annotation.

References

[1] Aviad Aberdam, Roy Ganz, Shai Mazor, and Ron Litman. Multimodal semi-supervised learning for text recognition. *arXiv preprint arXiv:2205.03873*, 2022. 5

[2] Aviad Aberdam, Ron Litman, Shahar Tsiper, Oron Anschel, Ron Slossberg, Shai Mazor, R Manmatha, and Pietro Perona. Sequence-to-sequence contrastive learning for text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15302–15312, 2021. 5

[3] Rowel Atienza. Vision transformer for fast and efficient scene text recognition. In *International Conference on Document Analysis and Recognition*, pages 319–334. Springer, 2021. 4

[4] Jeonghun Baek, Geewook Kim, Junyeop Lee, Sungrae Park, Dongyoon Han, Sangdoo Yun, Seong Joon Oh, and Hwalsuk Lee. What is wrong with scene text recognition model comparisons? dataset and model analysis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4715–4723, 2019. 4

[5] Jeonghun Baek, Yusuke Matsui, and Kiyoharu Aizawa. What if we only use real datasets for scene text recognition? toward scene text recognition with fewer labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3113–3122, 2021. 1

[6] Darwin Bautista and Rowel Atienza. Scene text recognition with permuted autoregressive sequence models. In *Proceedings of the 17th European Conference on Computer Vision (ECCV)*, Cham, 10 2022. Springer International Publishing. 6

[7] Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. Generalization in nli: Ways (not) to go beyond simple heuristics, 2021. 5

[8] Chee Kheng Chng, Yuliang Liu, Yipeng Sun, Chun Chet Ng, Canjie Luo, Zihan Ni, ChuanMing Fang, Shuaitao Zhang, Junyu Han, Errui Ding, et al. Icdar2019 robust reading challenge on arbitrary-shaped text-rrc-art. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1571–1576. IEEE, 2019. 1

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 5

[10] Sergi Garcia-Bordils, Andrés Mafla, Ali Furkan Biten, Oren Nuriel, Aviad Aberdam, Shai Mazor, Ron Litman, and Dimosthenis Karatzas. Out-of-vocabulary challenge report. *arXiv preprint arXiv:2209.06717*, 2022. 4

[11] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2315–2324, 2016. 5, 6

[12] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014. 5, 6

[13] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. Icdar 2015 competition on robust reading. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1156–1160. IEEE, 2015. 4

[14] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, and Lluis Pere De Las Heras. Icdar 2013 robust reading competition. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1484–1493. IEEE, 2013. 4

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

[16] Anand Mishra, Karteek Alahari, and CV Jawahar. Scene text recognition using higher order language priors. 2012. 1

[17] Nibal Nayef, Yash Patel, Michal Busta, Pinaki Nath Chowdhury, Dimosthenis Karatzas, Wafa Khlif, Jiri Matas, Umapada Pal, Jean-Christophe Burie, Cheng-lin Liu, et al. Icdar2019 robust reading challenge on multi-lingual scene text detection and recognition—rrc-mlt-2019. In *2019 International conference on document analysis and recognition (ICDAR)*, pages 1582–1587. IEEE, 2019. 4

[18] Anhar Risnumawan, Palaiahankote Shivakumara, Chee Seng Chan, and Chew Lim Tan. A robust arbitrary text detection system for natural scene images. *Expert Systems with Applications*, 41(18):8027–8048, 2014. 1

[19] Roi Ronen, Shahar Tsiper, Oron Anschel, Inbal Lavi, Amir Markovitz, and R Manmatha. Glass: Global to local attention for scene-text spotting. *arXiv preprint arXiv:2208.03364*, 2022. 5

[20] Baoguang Shi, Cong Yao, Minghui Liao, Mingkun Yang, Pei Xu, Linyan Cui, Serge Belongie, Shijian Lu, and Xiang Bai. Icdar2017 competition on reading chinese text in the wild (rctw-17). In *2017 14th iapr international conference on document analysis and recognition (ICDAR)*, volume 1, pages 1429–1434. IEEE, 2017. 4

[21] Amanpreet Singh, Guan Pang, Mandy Toh, Jing Huang, Wojciech Galuba, and Tal Hassner. Textocr: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8802–8812, 2021. 5

[22] Yipeng Sun, Jiaming Liu, Wei Liu, Junyu Han, Errui Ding, and Jingtuo Liu. Chinese street view text: Large-scale chinese text reading with partially supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9086–9095, 2019. 4

[23] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. *arXiv preprint arXiv:1601.07140*, 2016. 1

[24] Kai Wang, Boris Babenko, and Serge Belongie. End-to-end scene text recognition. In *2011 International conference on computer vision*, pages 1457–1464. IEEE, 2011. 4

[25] 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)*, pages 1577–1581. IEEE, 2019. 4

[26] Ying Zhang, Lionel Gueguen, Ilya Zharkov, Peter Zhang, Keith Seifert, and Ben Kadlec. Uber-text: A large-scale dataset for optical character recognition from street-level imagery. In *SUNw: Scene Understanding Workshop-CVPR*, volume 2017, page 5, 2017. 5