Text Detection Forgot About Document OCR

Krzysztof Olejniczak[1,2], Milan Šulc[2]

[1]University of Oxford, United Kingdom
[2]Rossum.ai, Czech Republic

Abstract

Detection and recognition of text from scans and other images, commonly denoted as *Optical Character Recognition* (OCR), is a widely used form of automated document processing with a number of methods available. Yet OCR systems still do not achieve 100% accuracy, requiring human corrections in applications where correct readout is essential. Advances in machine learning enabled even more challenging scenarios of text detection and recognition *"in-the-wild"* – such as detecting text on objects from photographs of complex scenes. While the state-of-the-art methods for *in-the-wild* text recognition are typically evaluated on complex scenes, their performance in the domain of documents is typically not published, and a comprehensive comparison with methods for document OCR is missing. This paper compares several methods designed for *in-the-wild* text recognition and for document text recognition, and provides their evaluation on the domain of structured documents. The results suggest that state-of-the-art methods originally proposed for *in-the-wild* text detection also achieve competitive results on document text detection, outperforming available OCR methods. We argue that the application of document OCR should not be omitted in evaluation of text detection and recognition methods.

Keywords

Text Detection, Text Recognition, OCR, Optical Character Recognition, Text In The Wild

1. Introduction

Optical Character Recognition (OCR) is a classic problem in machine learning and computer vision with standard methods [1, 2] and surveys [3, 4, 5, 6] available. Recent advances in machine learning and its applications, such as autonomous driving, scene understanding or large-scale image retrieval, shifted the attention of Text Recognition research towards the more challenging *in-the-wild* text scenarios, with arbitrarily shaped and oriented instances of text appearing in complex scenes. Spotting text *in-the-wild* poses challenges such as extreme aspect ratios, curved or otherwise irregular text, complex backgrounds and clutter in the scenes. Recent methods [7, 8] achieve impressive results on challenging text *in-the-wild* datasets like TotalText [9] or CTW-1500 [10], with F1 reaching 90% and 87% respectively. Although automated document processing remains one of the major applications of OCR, to the best of our knowledge, the results of *in-the-wild* text detection models were never comprehensively evaluated on the domain of documents and compared with methods developed for document OCR. This paper reviews several recent Text Detection methods developed for the *in-the-wild* scenario [11, 12, 13, 7, 14, 8], evaluates their performance (out of the box and fine-tuned) on benchmark document datasets [15, 16, 17], and compares their scores against popular Document OCR engines [18, 19, 2]. Additionally, we adopt publicly available Text Recognition models [20, 21] and combine them with Text Detectors to perform two-stage end-to-end text recognition for a complete evaluation of text extraction.

26th Computer Vision Winter Workshop, Robert Sablatnig and Florian Kleber (eds.), Krems, Lower Austria, Austria, Feb. 15-17, 2023

† The work was done when Krzysztof Olejniczak was an intern at Rossum.

CEUR Workshop Proceedings (CEUR-WS.org)

2. Related Work

2.1. Document OCR

OCR engines designed for the "standard" application domain of documents range from open-source projects such as TesseractOCR [2] and PP-OCR [1] to commercial services, including AWS Textract [18] or Google Document AI [19]. Despite Document OCR being a classic problem with many practical applications, studied for decades [22, 23], it still cannot be considered 'solved' – even the best engines struggle to achieve perfect accuracy. The methodology behind the commercial cloud services is typically not disclosed. The most popular[1] open-source OCR engine at the time of publication, Tesseract [2] (v4 and v5), uses a *Long Short-Term Memory* (LSTM) neural network as the default recognition engine.

2.2. In-the-wild Text Detection

2.2.1. Regression-based Methods

Regression-based Methods follow the object classification approach, reduced to a single-class problem. TextBoxes [25] and TextBoxes++ [26] locate text instances with various lengths by using sets of anchors with different aspect ratios. Various regression-based methods utilize

[1]Based on the GitHub repository [24] statistics.

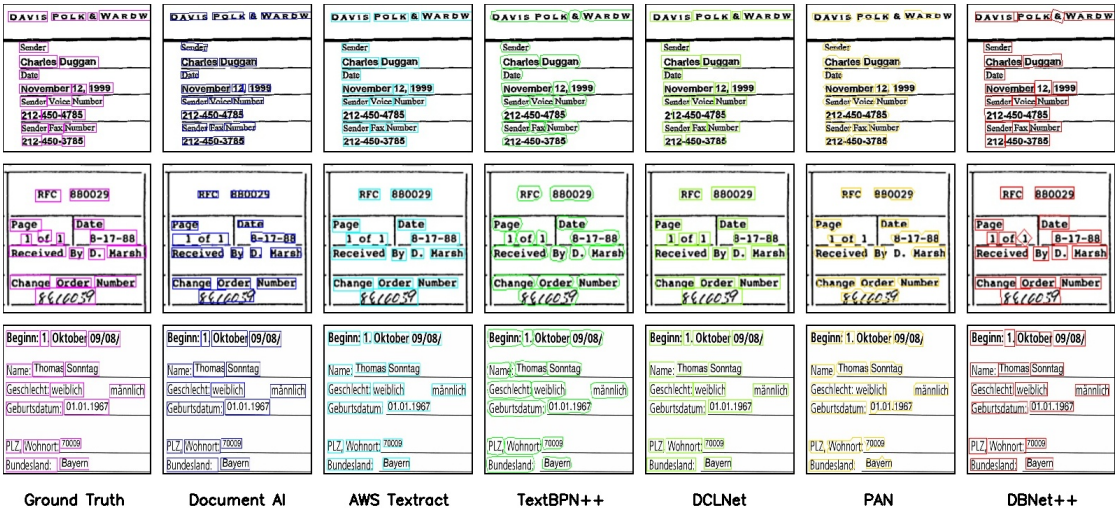

Figure 1: Comparison of detection results performed on documents from FUNSD dataset (cropped).

an iterative refinement strategy, iteratively enhancing the quality of detected boundaries. LOMO [27] uses an Iterative Refinement Module, which in every step regresses coordinates of each corner of the predicted boundary, with an attention mechanism. PCR [28] proposes a top-down approach, starting with predictions of centres and sizes of text instances, and iteratively improving the bounding boxes using its Contour Localisation Mechanism. TextBPN++ [8] introduces an Iterative Boundary Deformation Module, utilizing Transformer Blocks with multi-head attention [29] encoder and a multi-layer perceptron decoder, to iteratively adjust vertices of detected instances. Instead of considering vertices of the bounding boxes, DCLNet [12] predicts quadrilateral boundaries by locating four lines restricting the corresponding area, representing them in polar coordinates system. To address the problem of arbitrary-shaped text detection and accurately model the boundaries of irregular text regions, more sophisticated bounding boxes representation ideas have been developed. ABCNet [30] adapts cubic Bezier curves to parametrize curved text instances, gaining the possibility of fitting non-polygon shapes. FCENet [31] proposes Fourier Contour Embedding method, predicting the Fourier signature vectors corresponding to the representation of the boundary in Fourier domain, and uses them to generate the shape of the instance with Inverse Fourier Transformation.

2.2.2. Segmentation-based Methods

Segmentation-based Methods aim to classify each pixel as either text or non-text, and generate bounding boxes using post-processing on so obtained pixel maps. The binary, deterministic nature of such pixel classification problem may cause learning confusion on the borders of text instances. Numerous methods address this issue by predicting text kernels (central regions of instances) and appropriately gathering pixels around them. PSENet [32] predicts kernels of different sizes and forms bounding boxes by iteratively expanding their regions. PAN [14] generates pixel classification and kernel maps, linking each classified text pixel to the nearest kernel. CentripetalText [33] produces centripetal shift vectors that map pixels to correct text centres. KPN [34] creates pixel embedding vectors, for each instance locates the central pixel and retrieves the whole shapes by measuring the similarities in embedding vectors with scalar product. Vast majority of segmentation-based methods generate probability maps, representing how likely pixels are to be contained in some text region, and using certain binarization mechanism (e.g. by applying thresholding) convert them into binary pixel maps. However, the thresholds are often determined empirically, and uncareful choice of them may lead to drastic decrease in performance. To solve this problem, DBNet [13] proposes a Differentiable Binarization Equation, making the step between probability and classification maps end-to-end trainable and therefore letting the network learn how to accurately binarise predictions. DBNet++ [7] further improves on the baseline by extending the backbone network with an Adaptive Scale Fusion attention module, enhancing the upscaling process and obtaining deeper features. TextFuseNet [35] generates features on three different levels: global-, word- and character-level, and fuses them to gain

relevant context and deeper insight into the image structure. Instead of detecting words, CRAFT [11] locates text on character-level, predicting the areas covered by single letters, and links characters of each instance with respect to the generated affinity map.

3. Methods

3.1. Text Detection

To cover a wide range of text detectors, we selected methods from Section 2.2 with different approaches: for regression-based methods, we included TextBPN++ as a vertex-focused algorithm and DCLNet as an edge-focused approach. From segmentation-based methods, we selected DBNet and DBNet++ as pure segmentation and PAN as an approach linking text pixels to corresponding kernels. Finally, CRAFT was chosen as a character-level method.

3.2. Text Recognition

The ultimate goal of text detection, especially in the case of document processing, is to recognize the text within the detected instances. Therefore, to evaluate the suitability of popular *in-the-wild* detectors for document OCR, we perform end-to-end measurements with the following text recognition engines: SAR [20], MASTER [36] and CRNN [21]. The open-source engines were combined with the detection methods in a two-stage manner: the input image was initially processed by a detector, which returned bounding boxes. Afterwards, the corresponding cropped instances were passed to recognition models. As a point of reference, we compare both the detection and end-to-end recognition results of the selected methods with predictions of three common engines for end-to-end document OCR: Tesseract [2], Google Document AI [19] and AWS Textract [18].

3.3. Metric

To measure both detection and end-to-end performance, we used the CLEval [37] metric. Contrary to metrics such as *Intersection over Union* (IoU) perceiving text on word-level, CLEval measures precision and recall on character level. As a consequence, it slightly reduces the punishment for splitting or merging problematic instances (e.g dates), providing reliable and intuitive comparison of the quality of detection and recognition. Additionally, the *Recognition Score* evaluated by CLEval, approximately corresponding to the precision of character recognition, informs about the quality of the recognition engine specifically on the detected bounding boxes.

4. Experiments

4.1. Training Strategies

DBNet [13], DBNet++ [7] and PAN [14] were fine-tuned for 100 epochs (600 epochs in case of FUNSD) with batch size of 8 and initial learning rate set to 0.0001 and decreasing by a factor of 10 at the 60th and 80th epoch (200th and 400th for FUNSD). Baselines, pre-trained on SynthText [38] (DBNet, DBNet++) or ImageNet [39] (PAN), were downloaded from the MMOCR 0.6.2 Model Zoo [40]. DCLNet [12] was fine-tuned from a pre-trained model [41] on each dataset for 150 epochs with batch size of 4, initial learning rate of 0.001, decaying to 0.0001. For each dataset, TextBPN++ [8] was fine-tuned from a pre-trained model [42] for 50 epochs with batch size of 4, learning rate of 0.0001 and data augmentations consisting of flipping, cropping and rotations. Given no publicly-available training scripts for CRAFT, during the experiments, we used the MLT model from the github repository [43] without fine-tuning. All experiments were performed using Adam optimizer with momentum 0.9, on a single GPU with 11 GB of VRAM (GeForce GTX-1080Ti).

4.2. Detection Results

Results of the text detection methods selected in Section 3.1 on the datasets from Table 1 are presented in Table 2. On FUNSD dataset, DBNet++ achieves both the highest detection recall (97.40%) and F1-score (97.42%). The highest precision rate, 97.84% was scored by CRAFT. PAN performed the weakest out of all considered *in-the-wild* algorithms, scoring just 81.44% F1-score. Despite having achieved better results on FUNSD, segmentation-based approaches were outperformed by regression-based methods on CORD and XFUND. TextBPN++ proved to be the best performing algorithm on CORD in terms of recall and F1-score, scoring 99.74% and 99.19%, respectively. DCLNet, for which the best precision rate on CORD (98.67%) was recorded, achieved superior results on XFUND, outperforming the remaining methods with respect to all three measures: precision - 98.22%, recall - 98.17% and F1-score - 98.20%. Out of the considered popular engines for end-to-end document OCR, AWS Textract presented the best performance on the domain of scans of structured documents – FUNSD and XFUND – scoring 96.69% and 92.65% F1-score, respectively. Google Document AI generalized remarkably better to distorted photos of receipts from the CORD dataset, achieving 93.30% F1-score, surpassing the scores of AWS Textract and Tesseract. The results show that *in-the-wild* detectors fine-tuned on document datasets can outperform popular OCR engines on the domain of structured documents in terms of the CLEval detection metric. However, the results for the predictions of pre-trained detectors may not

Table 1
Document datasets used in the experiments for text detection and recognition.

Dataset	Training images	Test images	Document Types	Language
FUNSD [15]	149	50	Distorted forms, surveys, reports	English
CORD [16]	900	100	Photos of Indonesian receipts	English
XFUND [17]	1043	350	Clean scanned forms	Multilingual

Table 2
Comparison of the detection performance of the chosen methods on benchmark datasets, with respect to the CLEval metric. "P", "R" and "F1" represent the precision, recall and F1-score, respectively.

Method	FUNSD			CORD			XFUND		
	P	R	F1	P	R	F1	P	R	F1
PAN [14]	96.25	70.57	81.44	98.92	97.33	98.12	96.96	77.90	86.39
DBNet [13]	96.02	96.11	96.07	97.94	99.17	98.55	97.04	95.58	96.30
DBNet++ [7]	97.45	**97.40**	**97.42**	97.58	99.60	98.58	97.87	97.93	97.90
TextBPN++ [8]	96.63	95.59	96.11	98.65	**99.74**	**99.19**	97.88	94.29	96.05
DCLNet [12]	94.16	95.35	94.75	**98.67**	97.91	98.29	**98.22**	**98.17**	**98.20**
CRAFT [11]	**97.84**	95.72	96.77	94.25	88.46	91.26	89.75	93.02	91.36
Tesseract [2]	80.13	73.80	76.83	76.46	47.38	58.51	85.84	87.47	86.65
Document AI [19]	95.56	89.77	92.57	92.90	93.71	93.30	89.49	90.68	90.08
AWS Textract [18]	97.50	95.89	96.69	80.60	84.79	82.64	97.64	88.14	92.65

be fully representative due to differences in splitting rules. E.g. Document AI creates separate instances for special symbols, e.g. brackets, leading to undesired splitting of words like *"name(s)"* into several fragments, lowering precision and recall. On all experimented datasets, all fine-tuned *in-the-wild* text detection models reached high prediction scores, proving themselves capable of handling text in structured documents. Qualitative analysis of detectors' predictions revealed that the major sources of error were incorrect splitting of long text fragments (e.g e-mail addresses), merging instances in dense text regions and missing short stand-alone text, such as single-digit numbers.

4.3. Recognition Results

End-to-end text recognition results combining fine-tuned *in-the-wild* detectors with SAR [20] and MASTER [36] models from MMOCR 0.6.2 Model Zoo [46], and CRNN [21] from docTR [45] are listed in Table 3. The XFUND dataset was skipped for this experiment since it contains Chinese and Japanese characters, for which the recognition models were not trained. On FUNSD, the end-to-end measurement outcomes followed the patterns from detection: equipped with CRNN as the recognition engine, DBNet++ proved to be the best tuned model in terms of CLEval end-to-end Recall (93.52%) and F1-score (92.23%), losing only to CRAFT in terms of precision. Much higher F1-score (+2%) was measured for AWS Textract, whose end-to-end results outperformed all of the considered algorithms. It is important to note that the Recognition Score for AWS Textract reached almost 96%, surpassing CRNN's scores by c.a. 2%. This suggests that the recognition engine used in AWS Textract, performing much more accurately on FUNSD than the CRNN model, may have been a crucial reason for the good results. When evaluated on CORD, models with Differentiable Binarization scored the highest marks in all end-to-end measures: recall (DBNet++), precision and F1-score (DBNet); significantly surpassing the remaining methods. Interestingly, despite obtaining the best recall rate, DBNet++ did not beat the simpler DBNet in terms of end-to-end F1-score. The predictions of regression-based approaches, better than segmentation-based ones when pure detection scores were measured, appeared to combine slightly worse with CRNN. TextBPN++, however, remained competitive, achieving similar results to DBNet and DBNet++. Recognition Scores of CRNN, regardless the choice of *in-the-wild* detector, exceeded 93% on FUNSD and 98.5% on CORD, once again demonstrating the suitability of applying these algorithms to document text recognition. SAR model, not specifically trained on documents, presented poorer performance: the highest measured F1-scores on FUNSD and CORD were 86.36% and 85.25%, respectively, both obtained by the combination with TextBPN++. Fine-tuned SAR models achieved slightly higher F1-scores reaching 89.49% on FUNSD (equipped with DBNet++ as the detector) and 93.77% on CORD (combined with TextBPN++ detections). Despite gaining a noticeable advantage over the baseline, fine-tuned SAR models did not surpass the performance of the pre-trained CRNN. Similarly to SAR, the

Table 3
Comparison of the recognition performance of the chosen text detection methods combined with MMOCR's [44] SAR and MASTER default models, fine-tuned SAR, and docTR's [45] CRNN default model, on FUNSD and CORD, with respect to the CLEval metric. "P", "R", "F1" and "S" represent the end-to-end precision, recall, F1-score and Recognition Score, respectively.

Recognition	Detection	FUNSD				CORD			
		P	R	F1	S	P	R	F1	S
SAR [20] (baseline)	PAN [14]	76.14	74.17	75.14	79.79	82.04	84.27	83.14	84.76
	DBNet [13]	79.10	82.51	80.77	83.33	82.76	85.79	84.25	85.49
	CRAFT [11]	83.75	85.16	84.45	85.92	79.62	76.93	78.25	86.37
	TextBPN++ [8]	84.90	87.87	86.36	88.86	83.56	87.00	85.25	86.58
	DBNet++ [7]	80.04	83.53	81.75	82.85	82.95	86.66	84.76	85.89
	DCLNet [12]	77.67	82.27	79.91	81.80	82.75	85.53	84.11	86.16
SAR [20] (fine-tuned)	PAN [14]	86.37	76.61	81.20	90.23	87.73	88.95	88.34	90.59
	DBNet [13]	87.48	88.07	87.77	91.90	91.12	94.00	92.54	94.02
	CRAFT [11]	88.14	86.48	87.30	90.39	84.98	79.19	81.99	91.53
	TextBPN++ [8]	88.12	88.32	88.22	92.16	91.46	96.21	93.77	94.77
	DBNet++ [7]	89.15	89.83	89.49	92.13	90.40	93.83	92.09	93.54
	DCLNet [12]	86.10	87.30	86.70	90.46	87.69	90.02	88.84	91.58
MASTER [36]	PAN [14]	77.50	74.58	76.01	81.10	90.25	92.12	91.17	93.16
	DBNet [13]	80.30	83.11	81.68	84.55	91.94	94.31	93.11	94.62
	CRAFT [11]	82.06	82.90	82.48	84.22	85.81	81.86	83.79	92.93
	TextBPN++ [8]	82.10	83.93	83.00	85.96	91.77	94.79	93.26	94.78
	DBNet++ [7]	81.33	83.99	82.64	84.13	91.39	94.63	92.98	94.48
	DCLNet [12]	79.55	82.85	81.17	83.31	90.01	92.28	91.13	93.71
CRNN [21]	PAN [14]	90.31	87.14	88.70	94.00	95.70	96.52	96.10	98.65
	DBNet [13]	89.07	91.56	90.30	93.24	**96.00**	97.51	**96.75**	98.67
	CRAFT [11]	91.20	91.67	91.43	93.40	93.12	87.25	90.09	**98.73**
	TextBPN++ [8]	89.94	91.80	90.86	93.86	95.35	97.71	96.52	98.48
	DBNet++ [7]	90.97	93.52	92.23	93.71	95.43	**97.85**	96.62	98.51
	DCLNet [12]	89.84	92.95	91.37	93.16	95.04	96.34	95.69	98.52
Tesseract [2]		73.84	73.84	69.09	88.48	73.96	44.33	55.43	93.55
Google Document AI [19]		90.83	92.03	91.42	94.80	88.06	90.97	89.49	98.61
AWS Textract [18]		**93.61**	**95.46**	**94.53**	**95.78**	84.53	82.13	83.32	96.63

pre-trained MASTER model [46] worked the best in combination with TextBPN++, achieving F1 score of 83.00% on FUNSD and 93.26% on CORD.

5. Conclusions

Text detection research has witnessed great progress in recent years, thanks to advancements in deep machine learning. The recently introduced methods widened the range of possible applications of text detectors, making them viable for *in-the-wild* text spotting. This shifted the attention towards more complex scenarios, including arbitrarily-shaped text or instances with non-orthogonal orientations. With automated document processing remaining one of the most relevant commercial OCR applications, we stress the importance of determining whether the state-of-the-art methods for scene text spotting can also improve document OCR. Our experiments prove that detectors designed for *in-the-wild* text spotting can indeed be applied to documents with great success. In particular, fine-tuning models such as DBNet++ or TextBPN++ yielded over 96% detection F1-score on FUNSD, over 98% detection F1-score on CORD and over 96% detection F1-score on XFUND, with respect to the CLEval metric, outperforming Google Document AI and AWS Textract. Moreover, combining these detectors with a publicly-available CRNN recognition model in a two-stage manner consistently achieves over 90% CLEval end-to-end F1-score, even without explicit fine-tuning of CRNN. We hope the results will bring more attention to evaluating future Text Detection methods not only in the *text-in-the-wild* scenario, but also on the domain of documents.

Acknowledgement

We acknowledge the help of Bohumír Zámečník, an expert on OCR systems, who helped with the supervision of Krzysztof's internship project.

References

[1] Y. Du, C. Li, R. Guo, X. Yin, W. Liu, J. Zhou, Y. Bai, Z. Yu, Y. Yang, Q. Dang, H. Wang, PP-OCR: A practical ultra lightweight OCR system, CoRR abs/2009.09941 (2020). URL: https://arxiv.org/abs/2009.09941. arXiv:2009.09941.

[2] A. Kay, Tesseract: An open-source optical character recognition engine, Linux J. 2007 (2007) 2.

[3] K. Hamad, K. Mehmet, A detailed analysis of optical character recognition technology, International Journal of Applied Mathematics Electronics and Computers (2016) 244–249.

[4] T. Hegghammer, Ocr with tesseract, amazon textract, and google document ai: A benchmarking experiment, 2021. URL: osf.io/preprints/socarxiv/6zfvs. doi:10.31235/osf.io/6zfvs.

[5] N. Islam, Z. Islam, N. Noor, A survey on optical character recognition system, arXiv preprint arXiv:1710.05703 (2017).

[6] J. Memon, M. Sami, R. A. Khan, M. Uddin, Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr), IEEE Access 8 (2020) 142642–142668.

[7] M. Liao, Z. Zou, Z. Wan, C. Yao, X. Bai, Real-time scene text detection with differentiable binarization and adaptive scale fusion, IEEE Transactions on Pattern Analysis and Machine Intelligence (2022).

[8] S. Zhang, X. Zhu, C. Yang, H. Wang, X. Yin, Adaptive boundary proposal network for arbitrary shape text detection, in: 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021, IEEE, 2021, pp. 1285–1294.

[9] C. K. Ch'ng, C. S. Chan, C. Liu, Total-text: Towards orientation robustness in scene text detection, International Journal on Document Analysis and Recognition (IJDAR) 23 (2020) 31–52. doi:10.1007/s10032-019-00334-z.

[10] Y. Liu, L. Jin, S. Zhang, C. Luo, S. Zhang, Curved scene text detection via transverse and longitudinal sequence connection, Pattern Recognition 90 (2019) 337–345. URL: https://www.sciencedirect.com/science/article/pii/S0031320319300664. doi:https://doi.org/10.1016/j.patcog.2019.02.002.

[11] Y. Baek, B. Lee, D. Han, S. Yun, H. Lee, Character region awareness for text detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 9365–9374.

[12] Y. Bi, Z. Hu, Disentangled contour learning for quadrilateral text detection, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 909–918.

[13] M. Liao, Z. Wan, C. Yao, K. Chen, X. Bai, Real-time scene text detection with differentiable binarization, in: Proc. AAAI, 2020.

[14] W. Wang, E. Xie, X. Song, Y. Zang, W. Wang, T. Lu, G. Yu, C. Shen, Efficient and accurate arbitrary-shaped text detection with pixel aggregation network, CoRR abs/1908.05900 (2019). URL: http://arxiv.org/abs/1908.05900. arXiv:1908.05900.

[15] J.-P. T. Guillaume Jaume, Hazim Kemal Ekenel, Funsd: A dataset for form understanding in noisy scanned documents, in: Accepted to ICDAR-OST, 2019.

[16] S. Park, S. Shin, B. Lee, J. Lee, J. Surh, M. Seo, H. Lee, Cord: A consolidated receipt dataset for post-ocr parsing (2019).

[17] Y. Xu, T. Lv, L. Cui, G. Wang, Y. Lu, D. Florencio, C. Zhang, F. Wei, XFUND: A benchmark dataset for multilingual visually rich form understanding, in: Findings of the Association for Computational Linguistics: ACL 2022, Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 3214–3224. URL: https://aclanthology.org/2022.findings-acl.253. doi:10.18653/v1/2022.findings-acl.253.

[18] Amazon, Amazon textract, https://aws.amazon.com/textract, 2022. Accessed: 2022-09-25.

[19] Google, Google cloud document ai, https://cloud.google.com/document-ai, 2022. Accessed: 2022-09-25.

[20] H. Li, P. Wang, C. Shen, G. Zhang, Show, attend and read: A simple and strong baseline for irregular text recognition, CoRR abs/1811.00751 (2018). URL: http://arxiv.org/abs/1811.00751. arXiv:1811.00751.

[21] B. Shi, X. Bai, C. Yao, An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition, CoRR abs/1507.05717 (2015). URL: http://arxiv.org/abs/1507.05717. arXiv:1507.05717.

[22] S. Mori, H. Nishida, H. Yamada, Optical character recognition, John Wiley & Sons, Inc., 1999.

[23] H. F. Schantz, The history of ocr, optical character recognition, Manchester Center, VT: Recognition Technologies Users Association (1982).

[24] S. W. et al., Tesseract open source ocr engine (main repository), https://github.com/tesseract-ocr/tesseract, 2022. Accessed: 2022-10-14.

[25] M. Liao, B. Shi, X. Bai, X. Wang, W. Liu, Textboxes: A fast text detector with a single deep neural network, in: AAAI, 2017.

[26] B. S. Minghui Liao, X. Bai, TextBoxes++: A single-shot oriented scene text detector, IEEE Transactions on Image Processing 27 (2018) 3676–3690. URL: https://doi.org/10.1109/TIP.2018.2825107. doi:10.1109/TIP.2018.2825107.

[27] C. Zhang, B. Liang, Z. Huang, M. En, J. Han, E. Ding, X. Ding, Look more than once: An accu-

rate detector for text of arbitrary shapes, CoRR abs/1904.06535 (2019). URL: http://arxiv.org/abs/1904.06535. arXiv:1904.06535.

[28] P. Dai, S. Zhang, H. Zhang, X. Cao, Progressive contour regression for arbitrary-shape scene text detection, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 7389–7398. doi:10.1109/CVPR46437.2021.00731.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, CoRR abs/1706.03762 (2017). URL: http://arxiv.org/abs/1706.03762. arXiv:1706.03762.

[30] Y. Liu, H. Chen, C. Shen, T. He, L. Jin, L. Wang, Abcnet: Real-time scene text spotting with adaptive bezier-curve network, CoRR abs/2002.10200 (2020). URL: https://arxiv.org/abs/2002.10200. arXiv:2002.10200.

[31] Y. Zhu, J. Chen, L. Liang, Z. Kuang, L. Jin, W. Zhang, Fourier contour embedding for arbitrary-shaped text detection, in: CVPR, 2021.

[32] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, S. Shao, Shape robust text detection with progressive scale expansion network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 9336–9345.

[33] T. Sheng, J. Chen, Z. Lian, Centripetaltext: An efficient text instance representation for scene text detection, in: Thirty-Fifth Conference on Neural Information Processing Systems, 2021.

[34] S.-X. Zhang, X. Zhu, J.-B. Hou, C. Yang, X.-C. Yin, Kernel proposal network for arbitrary shape text detection, 2022. URL: https://arxiv.org/abs/2203.06410. doi:10.48550/ARXIV.2203.06410.

[35] J. Ye, Z. Chen, J. Liu, B. Du, Textfusenet: Scene text detection with richer fused features, in: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, International Joint Conferences on Artificial Intelligence Organization, 2020, pp. 516–522.

[36] N. Lu, W. Yu, X. Qi, Y. Chen, P. Gong, R. Xiao, MASTER: multi-aspect non-local network for scene text recognition, CoRR abs/1910.02562 (2019). URL: http://arxiv.org/abs/1910.02562. arXiv:1910.02562.

[37] Y. Baek, D. Nam, S. Park, J. Lee, S. Shin, J. Baek, C. Y. Lee, H. Lee, Cleval: Character-level evaluation for text detection and recognition tasks, CoRR abs/2006.06244 (2020). URL: https://arxiv.org/abs/2006.06244. arXiv:2006.06244.

[38] A. Gupta, A. Vedaldi, A. Zisserman, Synthetic data for text localisation in natural images, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[39] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE conference on computer vision and pattern recognition, Ieee, 2009, pp. 248–255.

[40] Z. Kuang, H. Sun, Z. Li, X. Yue, T. H. Lin, J. Chen, H. Wei, Y. Zhu, T. Gao, W. Zhang, K. Chen, W. Zhang, D. Lin, Text detection models - mmocr 0.6.2 documentation, https://mmocr.readthedocs.io/en/latest/textdet_models.html, 2022. Accessed: 2022-10-14.

[41] Y. Bi, Z. Hu, Pytorch implementation of dclnet "disentangled contour learning for quadrilateral text detection", https://github.com/SakuraRiven/DCLNet, 2021. Accessed: 2022-10-13.

[42] S. Zhang, X. Zhu, C. Yang, H. Wang, X. Yin, Arbitrary shape text detection via boundary transformer, https://github.com/GXYM/TextBPN-Plus-Plus, 2022. Accessed: 2022-09-29.

[43] Y. Baek, B. Lee, D. Han, S. Yun, H. Lee, Official implementation of character region awareness for text detection (craft), https://github.com/clovaai/CRAFT-pytorch, 2019. Accessed: 2022-10-13.

[44] Z. Kuang, H. Sun, Z. Li, X. Yue, T. H. Lin, J. Chen, H. Wei, Y. Zhu, T. Gao, W. Zhang, K. Chen, W. Zhang, D. Lin, Mmocr: A comprehensive toolbox for text detection, recognition and understanding, arXiv preprint arXiv:2108.06543 (2021).

[45] Mindee, doctr: Document text recognition, https://github.com/mindee/doctr, 2021.

[46] Z. Kuang, H. Sun, Z. Li, X. Yue, T. H. Lin, J. Chen, H. Wei, Y. Zhu, T. Gao, W. Zhang, K. Chen, W. Zhang, D. Lin, Text recognition models - mmocr 0.6.2 documentation, https://mmocr.readthedocs.io/en/latest/textrecog_models.html, 2021. Accessed: 2022-10-14.