Traditional Chinese Synthetic Datasets Verified with Labeled Data for Scene Text Recognition

Yi-Chang Chen[1], Yu-Chuan Chang[1], Yen-Cheng Chang[1] and Yi-Ren Yeh[2]

[1]E.SUN Financial Holding CO., LTD., Taiwan
[2]Department of Mathematics, National Kaohsiung Normal University, Taiwan
E-MAIL: ycc.tw.email@gmail.com, chuanworks@gmail.com, timtimchang9666@gmail.com, yryeh@nknu.edu.tw

Abstract—**Scene text recognition (STR) has been widely studied in academia and industry. Training a text recognition model often requires a large amount of labeled data, but data labeling can be difficult, expensive, or time-consuming, especially for Traditional Chinese text recognition. To the best of our knowledge, public datasets for Traditional Chinese text recognition are lacking. This paper presents a framework for a Traditional Chinese synthetic data engine which aims to improve text recognition model performance. We generated over 20 million synthetic data and collected over 7,000 manually labeled data `TC-STR 7k-word` as the benchmark[1]. Experimental results show that a text recognition model can achieve much better accuracy either by training from scratch with our generated synthetic data or by further fine-tuning with `TC-STR 7k-word`.**

I. Introduction

Scene text recognition is a challenging task due to the variety of text styles and backgrounds. Typically, a large amount of data is required to train a competitive recognition model. To the best of our knowledge, public datasets for Traditional Chinese text recognition are lacking, and the wide range of fonts and styles for Chinese characters causes additional difficulty for data collection, which is already difficult, expensive, and time-consuming.

Synthetic data is increasingly used to reduce the effort required for data collection and labeling for training deep learning models, especially in computer vision [1], [2]. However, collecting a sufficient volume and variety of characters for Traditional Chinese text recognition is difficult. To overcome this problem, we generate synthetic scene text images with greater diversity.

The proposed synthetic data engine for Traditional Chinese scene text images begins by designing different attributes for generating text images, such as font rendering, background rendering, and text properties. Based on these different attributes, the synthetic word data generator imitates scene text from the real world to generate text images without human labeling. In addition to the synthetic data engine, we also create a real-world Traditional Chinese scene text dataset for evaluation. All text boxes and labels are cropped and annotated manually from real-world images.

In our experiments, we investigate the influence of different dataset sizes and the impacts of different attributes. We also show that the model pre-trained with our synthetic dataset outperforms one trained with the target dataset (if the target data are available), and that our synthetic data engine could significantly reduce the cost of human labeling and enhance text recognition performance.

II. Related Work

As mentioned in Section I, synthetic data have been widely used in training deep neural networks. [1] proposed a popular synthetic dataset for real-world text recognition, the `MJSynth` dataset, containing 8.9 million text images and 1,400 different fonts. The `MJSynth` dataset is composed of three separate image layers: background, foreground, and optional shadow/border. Each text is synthesized with different font properties, such as kerning, weight, and underline. The `SynthText` dataset [3] is another popular synthetic dataset used in text recognition, consisting of 5.5 million word images for which the synthetic text is blended with existing background images according to the local 3D scene geometry.

The most popular methods for recognition models use multi-stage pipelines [4], including transformation, feature extraction, sequence modeling, and prediction stages. We use the thin-plate spline (TPS) transformation in the transformation stage. TPS is a variant of the spatial transformation network (STN) and has been shown to treat diverse aspect ratios of text lines [12], [5]. For the feature extraction stage, many CNN architectures have been proposed for the feature extraction. Based on the results in [4], we choose ResNet as our feature extractor due to its superior performance. According to suggestions from [4], we apply Bidirectional LSTM (BiLSTM) to catch contextualized information from the scene text images. In the last stage, Connectionist Temporal Classification (CTC) [6] brings more benefits for languages with large character sets, such as Traditional Chinese [13]. Departing from [4], we used CTC instead of the encoder-decoder framework at the prediction stage. In a nutshell, we choose the combination TPS-ResNet-BiLSTM-CTC for Chinese scene text recognition in our experiments.

---

[1]These two datasets are available at https://github.com/GitYCC/traditional-chinese-text-recogn-dataset
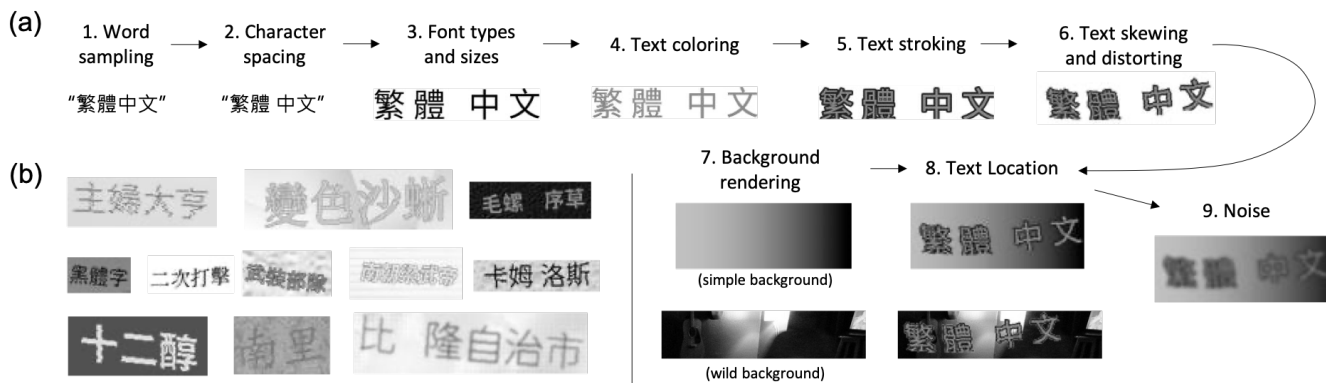
Fig. 1. (a) The nine steps of our Traditional Chinese synthetic text engine. (b) Some randomly sampled examples generated from our synthetic text engine.

Fig. 2. Some randomly sampled examples from the `TC-STR 7k-word` dataset.

III. PROPOSED SYNTHETIC AND REAL-WORLD TRADITIONAL CHINESE SCENE TEXT DATASETS

A. Traditional Chinese Synthetic Scene Text Engine

Inspired by `MJSynth` [1], `SynthText` [3] and Belval/TextRecognitionDataGenerator[2], we propose a framework for generating scene text images for Traditional Chinese. To produce synthetic text images similar to real-world ones, we use different kinds of mechanisms for rendering, as shown in Fig. 1(a). The details of our data generating pipeline are as follows:

1) **Word sampling** – In our synthetic scene text dataset, each synthesized text image is associated with a word that contains several characters as shown in Fig. 1(b). To obtain a more diverse word set for word sampling, we extract words from two sources: Taiwan Ministry of Education dictionary[3] and Wikipedia page titles. Our word set contains 1,076,772 words and 12,108 characters. On the other hand, to produce different appearances and backgrounds of the same word, we also repeatedly sample each word multiple times, each time applying

different rendering tricks, such as font types, font sizes, font colors, stroking, skewing, distorting, simple and wild backgrounds, word location, and noise.

2) **Character spacing** – It is common to have spaces between Chinese characters, especially in scene texts. To produce near-authentic Chinese scene texts, we randomly insert multiple spaces between characters within a word. As shown in the second step of Fig. 1(a), we insert a space between the third and fourth characters where the location and the number of spaces are randomly determined.

3) **Font types and sizes** – In our proposed synthetic dataset, we gathered 175 fonts with commercial-free authorization. The typefaces of those fonts include Gothic, Ming, Kai, Yuan, etc. For each synthetic image, one of the collected fonts is randomly selected for the rendering of font types. All fonts are sized from 20 to 50 points.

4) **Text coloring** – In our image preprocessing, all the scene text images will be converted to grayscale before training the text recognition model. Thus, we simply fill the text with 14 different grayscale intensities. The hex color codes of our text coloring are #000000, #141414, #282828, #3C3C3C, #505050, #646464, #787878, #8C8C8C, #A0A0A0, #B4B4B4, #C8C8C8, #DCDCDC, #F0F0F0, and #FFFFFF.

5) **Text stroking** – To produce different text outline styles, we also assign different widths of text stroking. The widths are randomly selected from 0 to 3 points as shown in the fourth step of Fig. 1(a).

6) **Text skewing and distorting** – Real-world scene texts are not often well-aligned horizontally due to appearance preferences or environmental constraints, such as the real-world examples in Fig. 2. To simulate these properties, we skew the text after coloring/stroking and distort it vertically and horizontally. The sixth step of Fig. 1(a) shows an example of our text skewing and distorting.

7) **Background rendering** – There are two kinds of raw background images in background rendering. The first

| Training Data | | Validation Data | | Test |
name	# of data	name	# of data	Accuracy
Subset of `TC-STR-train`	3,251	Subset of `TC-STR-train`	586	4.10%
Subset of `TC-STR-train` + Augmentation [15]	1,232,129	Subset of `TC-STR-train`	586	8.82%
Our Synthetic Data w/ Simple BGs	1,076,764	`TC-STR-train`	3,837	78.44%
Our Synthetic Data w/ Simple BGs	16,151,460	`TC-STR-train`	3,837	84.11%
Our Synthetic Data w/ Mixed BGs	21,535,280	`TC-STR-train`	3,837	**84.75%**
Our Synthetic Data w/ Mixed BGs	21,535,280	Our Synthetic Data w/ Mixed BGs	6,000	83.51%

is simple background images retrieved from Google image search with specific queried keywords, such as slide background and texture. The other kind is wild background images, which are extracted from the COCO [9] dataset. However, some images from the COCO dataset contain texts. For example, the COCO-Text [10] dataset selects images that contain texts from the COCO dataset, and labels those images with locations and texts. Thus, to avoid extracting wild background images with texts, we excluded those images in the COCO dataset by referencing the labels of the COCO-Text [10] dataset. The examples of simple and wild backgrounds are respectively presented at the upper and bottom images of the seventh step in Fig. 1(a).

8) **Text Location** – In scene text recognition, the text detection model might not crop a text box properly. That is, many scene texts would not be well-aligned to the center of the text box. To account for this, the foreground text is located at the background image with random margins. The margin is randomly set from 1 to 4 points in our synthetic dataset.

9) **Noise** - In our final rendering step, Gaussian blur is applied to the text image as shown in the ninth step of Fig. 1(a).

In our synthetic scene text data, all the synthesized images are preprocessed by following these steps. The goal is to produce scene text images similar to the real-world ones. To evidence the effectiveness of these rendering tricks, we also conduct experiments to address these issues. The detailed results are shown in Secton IV.

B. Real-world Traditional Chinese Scene Text Dataset: TC-STR 7k-word

It is worth repeating that the lack of public authentic Traditional Chinese scene text datasets makes evaluation more difficult. To overcome this problem, inspired by the IIIT 5K-word dataset [11], we create a real-world Traditional Chinese scene text recognition dataset (TC-STR 7k-word). Our TC-STR 7k-word dataset collects about 1,554 images from Google image search to produce 7,543 cropped text images. To increase the diversity in our collected scene text images, we search for images under different scenarios and query keywords. Since the collected scene text images are to be used in evaluating text recognition performance, we

manually crop text from the collected images and assign a label to each cropped text box.

To optimize the ease of use of the TC-STR 7k-word, we also split the dataset into training and testing sets by considering the distribution of characters. That is, we balance the distribution of each character between training and testing sets, making sure each character in the testing set is also found in the training set. This data splitting strategy produces a training set of 3,837 text images (TC-STR-train) and a testing set of 3,706 images (TC-STR-test) in our TC-STR 7k-word dataset.

IV. EXPERIMENTS

A. Experimental Settings

Our experiments focus on the effectiveness of the synthetic data generated by our proposed synthetic scene text engine. Several synthetic datasets are generated under different settings of the synthetic scene text engine, and the real-world scene text data, TC-STR-test, are used to evaluate recognition performance in all experiments. For image preprocessing, all scene text images are converted to grayscale and resized to 32×100 without aspect ratio preservation.

We choose the TPS-ResNet-BiLSTM-CTC framework as our base model, respectively setting the number of fiducial points of TPS, the number of output channels of ResNet, and the size of the BiLSTM hidden state to 20, 512, and 256. For the optimizer of the learning model, we used AdaDelta [14] with a learning rate of $lr = 1.0$ and a decay rate of $\rho = 0.95$. Gradient clipping is set to magnitude 5. Models are trained with a batch size of 192 images. We validate the model after every 2,000 iterations and chose the model with the highest accuracy based on a maximum of 300K iterations.

B. Trained from scratch with TC-STR-train

As a comparison for synthetic datasets, we use the real-world data to establish the baseline for model training. Thus, our baseline model only uses the TC-STR-train for training (3,251 for training and 586 for validation) and is evaluated on TC-STR-test. However, the baseline model only can achieve 4.10% accuracy since the fonts, backgrounds, text sizes, and distortions in TC-STR 7k-word are quite diverse.

Data augmentation is a common strategy to increase training data diversity. In our experiments, we adopted the data augmentation methods proposed in [15], which achieve significant
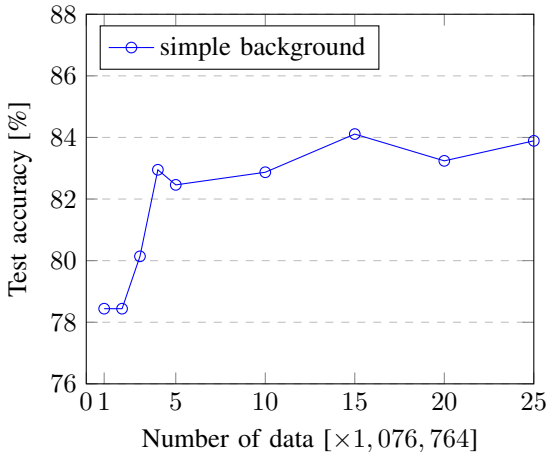
Fig. 3. Results on different data sizes of synthetic data with simple backgrounds.
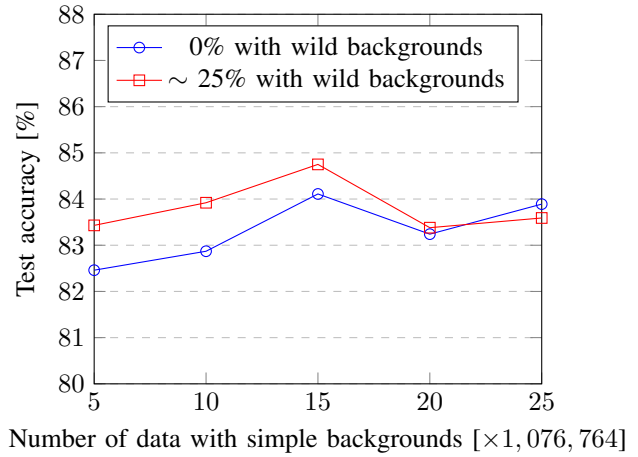

Fig. 5. Results on different data sizes of synthetic data by a fixed ratio (∼25%) of wild backgrounds to the whole synthetic data.
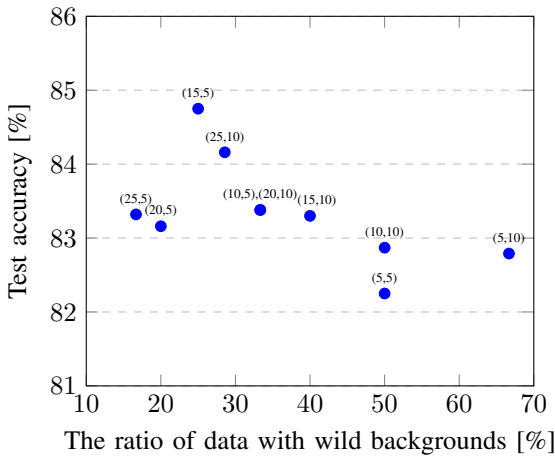

Fig. 4. Results on the different ratios of wild background images to the whole synthetic data. (n_s, n_w) represents that a training set contains n_s and n_w units of simple and wild background images respectively.

improvements on several public benchmark datasets [11], [7], [8]. We scaled the original images with seven different sizes and generated augmented images with respective distortion, stretch, and perspective factors of 24, 24, and 6. Thus, the number of total images for model learning is $(1 + (24 + 24 + 6) \times 7) \times 3,251 = 1,232,129$.

With the augmented images, the accuracy could be improved significantly from 4.10% to 8.82% as shown in the second row of TABLE I. However, the improvement is still not adequate for practical applications. To address this issue, we use the synthetic data to improve performance in the following experiments.

C. Trained from our proposed synthetic text data

We conducted synthetic training experiments under different settings of the synthetic scene text engine. We trained the model only using synthetic text data and validated it on

`TC-STR-train` to prevent the model from over-fitting on the synthetic text data in the following experiments.

The experiments described in this section were conducted with two purposes. First, we explore the sensitivity to synthetic data size and seek to identify a proper amount of synthetic data for our Traditional Chinese scene text recognition task. To this end, we gradually increased the synthetic text data with simple backgrounds, and evaluate the performance on `TC-STR-test`. The results are shown in Fig. 3, with the x-axis representing the synthetic data size. Each unit stands for 1,076,772 images according to the size of the lexicons in our synthetic scene text engine. Performance is found to significantly improve as we initially increase the synthetic data size and tends to plateau after more than four units of synthetic data are added. In this experiment, we achieve the highest accuracy 84.11% by using 15 units of synthetic data (see row 4 in TABLE I). This shows that our proposed synthetic datasets can make up for the lack of manually labeled data.

We also investigate the effect of using wild backgrounds by comparing them with images using only simple backgrounds. Different ratios of synthetic data with wild backgrounds are considered in this experiment. In the following, we use the notation (n_s, n_w) to represent that a training set has respectively n_s and n_w units of simple and wild background images. We design ten combinations of training sets, including (5,5), (10,5), (15,5), (20,5), (25,5), (5,10), (10,10), (15,10), (20,10), and (25,10). The results of these training sets are presented in Fig. 4, which shows that the recognition model achieves better accuracy roughly at a ratio of 25%. To investigate whether 25% is a proper ratio for mixing the wild backgrounds into our synthetic data or not, we also generate five training sets as follows: (5,2), (10,3), (15,5), (20,7), (25,8). All the ratios of wild backgrounds to the whole synthetic data are quite close to 25%. Fig. 5 presents the results of these five combinations and shows that generating synthetic data with a proper amount of wild background images improves performance over models trained only with simple backgrounds.

| Fine-tuning Training Data | | Fine-tuning Validation Data | | Test |
name	# of data	name	# of data	Accuracy
Subset of `TC-STR-train`	3,251	Subset of `TC-STR-train`	586	89.64%
Subset of `TC-STR-train` + Augmentation [15]	1,232,129	Subset of `TC-STR-train`	586	89.99%

TABLE III
Results on the ablations of background diversity, font diversity, scene
diversity and word diversity.

Training Data	Test Accuracy	
Our Synthetic Data with Mixed BGs	84.75%	
w/o background diversity	8.82%	(-75.93%)
w/o font diversity	62.09%	(-22.66%)
w/o scene diversity	81.14%	(-3.61%)
w/o word diversity	78.66%	(-6.09%)

D. Ablation study

We have shown that our proposed synthetic scene text engine significantly improves the performance of a scene text recognition model. The use of many rendering tricks used in our synthetic data generation raises the question of the contribution of these tricks to recognition performance. To address this question, we perform ablation studies in background diversity, font diversity, scene diversity, and word diversity in our experiments.

For the ablation of background diversity, we generate synthetic images only using black texts and white backgrounds as the training set. The results in TABLE III show that the unique background affected the performance significantly, and decreased accuracy. This implies that background rendering could produce synthetic scene text images similar to real-world ones.

Considering the font diversity, we used Source Han Sans as the single font without stroking in our ablation experiments. Results from TABLE III also show that performance is reduced without font diversity and that varied font types are beneficial for the generalization to other new fonts.

To test the ablation of scene diversity, we removed the rendering steps of skewing, distorting, and noise in our synthetic data generation. Compared with the previous ablation experiments, the influence of scene diversity is not significant from the results of TABLE III, possibly because there are fewer blurry images in the real world, such as the images of `TC-STR 7k-word`.

For the ablation of word diversity, we removed 9/10 words from the dictionary when generating synthetic images. The results again suggest that the diversity of the character set will affect recognition performance.

E. Trained and validated from our proposed synthetic text data

In previous sections, we use `TC-STR-train` or its subset as the validation set in our training procedure. However, the labeled scene text images of interest might not be easily available. Thus, we conduct an experiment which only uses synthetic data in the training procedure.

The recognition is learned by 20 units of synthetic data with mixed backgrounds ($20 \times 1,076,772$ images) as the training set and 6,000 synthetic images with mixed backgrounds as the validation set. Evaluating the learned model on `TC-STR-test`, we can achieve 83.51% accuracy as shown in the sixth row of TABLE I. The result shows the learned model still performs well only using synthetic images, which also suggests our synthetic engine can generate informative images similar to real ones.

F. Fine-tuning with our pre-trained model

In Section IV-B, we show that the recognition model trained from scratch with `TC-STR-train` did not perform well on the `TC-STR-test`, due to the lack of training data. To overcome this problem, pre-trained models are widely used as a good initialization for deep neural networks. In our fine-tuning procedure, we first obtain the pre-trained model learned from 20 units of synthetic data with mixed backgrounds, then train the model with the subset of `TC-STR-train`. The fine-tuned model achieves 89.64% accuracy as shown in TABLE II, outperforming the pre-trained model (84.75%). The model has learned the characteristics of real-world texts after pre-training, capturing more details from a small amount of labeled data.

Besides, we also apply data augmentation to the fine-tuning procedure. It can be easily observed that improvement of recognition with data augmentation is relatively small from TABLE I. The possible reason is that we has generated many synthetic images for pre-trained model. Thus, the effect of augmentation in the fine-tuning procedure is not significant.

V. CONCLUSION

Addressing the lack of a public Traditional Chinese scene text dataset, we introduce our manually labeled real-world `TC-STR 7k-word` dataset offering a benchmark dataset for further research. In addition to the proposed real-world dataset, we present a framework of a Traditional Chinese synthetic data engine using the synthetic data to improve performance. In our ablation studies, we conclude that background rendering is the most critical step for generating synthetic scene text images. A model pre-trained using our proposed synthetic dataset achieves optimal accuracy (89.64%) after fine-tuning and evaluation on the `TC-STR 7k-word` dataset.

REFERENCES

[1] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Synthetic data and artificial neural networks for natural scene text recognition," Workshop on Deep Learning, NIPS, 2014.

[2] A. Gupta, A. Vedaldi, A. Zisserman, "Synthetic Data for Text Localisation in Natural Images," CVPR, 2016.

[3] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," CVPR, 2016.

[4] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh and H. Lee, "What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis," IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

[5] M. Jaderberg, K. Simonyan, A. Zisserman and K. Kavukcuoglu, "Spatial transformer networks," Advances in neural information processing systems, 2015.

[6] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labeling unsegmented sequence data with recurrent neural networks," ICML, 2006.

[7] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. De Las Heras, "ICDAR 2013 robust reading competition," ICDAR, 2013.

[8] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, F. Shafait, S. Uchida and E. Valveny, "ICDAR 2015 competition on robust reading," ICDAR, 2015.

[9] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft COCO: Common objects in context," ECCV, 2014.

[10] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie, "Cocotext: Dataset and benchmark for text detection and recognition in natural images," arXiv, 2016.

[11] A. Mishra, K. Alahari and C.V. Jawahar, "Scene Text Recognition using Higher Order Language Priors," BMVC, 2012.

[12] W. Liu, C. Chen, K.-Y. K Wong, Z. Su, and J. Han, "Star-net: A spatial attention residue network for scene text recognition," BMVC, 2016.

[13] S. Long, X. He and C. Yao, "Scene text detection and recognition: The deep learning era," arXiv, 2018.

[14] M. D. Zeiler, "Adadelta: an adaptive learning rate method," arXiv, 2012.

[15] C. Luo and Y. Zhu, L. Jin and Y. Wang, "Learn to Augment: Joint Data Augmentation and Network Optimization for Text Recognition," CVPR, 2020.