




# Object Detection Based Handwriting Localization

Yuli Wu<sup>1\*</sup> , Yucheng Hu<sup>2\*</sup> , and Suting Miao<sup>3</sup> 

<sup>1</sup> Rheinisch-Westfälische Technische Hochschule Aachen, Germany

<sup>2</sup> Nanjing Normal University, China

<sup>3</sup> SAP Innovation Center Network (ICN) Nanjing, China

**Abstract.** We present an object detection based approach to localize handwritten regions from documents, which initially aims to enhance the anonymization during the data transmission. The concatenated fusion of original and preprocessed images containing both printed texts and handwritten notes or signatures are fed into the convolutional neural network, where the bounding boxes are learned to detect the handwriting. Afterwards, the handwritten regions can be processed (*e.g.* replaced with redacted signatures) to conceal the *personally identifiable information* (PII). This processing pipeline based on the deep learning network Cascade R-CNN works at 10 fps on a GPU during the inference, which ensures the enhanced anonymization with minimal computational overheads. Furthermore, the impressive generalizability has been empirically showcased: the trained model based on the English-dominant dataset works well on the fictitious unseen invoices, even in Chinese. The proposed approach is also expected to facilitate other tasks such as handwriting recognition and signature verification.

**Keywords:** handwriting localization, object detection, regional convolutional neural network, anonymization enhancement

## 1 Introduction

Handwriting localization plays an important role in the following scenarios: First, the handwritten regions in the documents may contain sensitive information, which must be anonymized before transmission. Second, handwriting localization can be naturally served as the first stage to achieve *handwriting-to-text* recognition. Third, signatures to be verified must be extracted via localization from their surrounding texts or lines in the documents.

This work is initially motivated by the demanding case of anonymization enhancement. Access to data is vital to undertake enterprise today. One of the most common data types would be the invoices: with digitalized invoices and all kinds of powerful AI-driven technologies, the companies would be able to analyze customers' behaviors and extract business intelligence automatically, offering

---

\* This work was done during their internship at SAP ICN Nanjing.

utmost help to refine strategies and make decisions. However, the *personally identifiable information* (PII) must be anonymized beforehand, as it is not worth the risk of any privacy exposure.

Ostensibly, tabular texts are of overwhelming majority in the business processing. In fact, the documents containing handwritten notes or signatures, such as invoices, also play an important role. The goal of this work is to localize the handwritten regions from the full-page invoice images for anonymization enhancement. The detected handwriting shall be anonymized afterwards, while the detailed implementation of which is beyond the scope of this work. As the expected results of the whole processing, handwriting should be excluded from the anonymized invoices, where it is assumed all handwritten regions would contain PII. One trivial way to realize this would be replacing the handwritten boxes with redacted signatures or notes.

Tesseract [24], the de facto paradigm of *optical character recognition* (OCR) engines, is nowadays widely used in the industry to extract textual information from images. OCR engines are competent to deal with the *optimal* data, which is referred to as the image data of documents, where all items of interest are regularly printed texts under the context of this work. In contrast, the real-world documents are usually the ones containing not only the regularly printed texts, but also some irregular patterns, such as handwritten notes, signatures, logos etc, which might also be desired.

In this work, we adopt object detection approaches with deep learning networks to localize handwritten regions in the document data based on the SAP’s Data Anonymization Challenge<sup>1</sup>. The feasibility and effectiveness of such algorithms have been empirically shown on those scenarios, where the *objects* (handwriting) and the *backgrounds* (printed texts) are extremely similar. Besides, the improvement from Faster R-CNN [23] to Cascade R-CNN [1] can be effortlessly reproduced. In addition, the new baseline of the handwriting localization as the subtask from the SAP’s Data Anonymization Challenge<sup>1</sup> has been released. Last but not least, the proposed deep learning approach with Cascade R-CNN [1] has demonstrated impressive generalizability. The trained model based on the English-dominant dataset works well on the fictitious unseen invoices, even for those in Chinese as toy examples. Empirically, it is believed that the deep learning model has learned the *irregularity* of the images.

Since the detailed types of handwritten regions, such as signatures or notes, are not discriminated during the experiments, we term *detection* and *localization* in this work interchangeably. The one-class detection merely consists of the localization regression task without classification. Despite the simplicity of the task description, it is still challenging to distinguish the handwritten notes from the printed texts, as they are similar regarding the contextual information. Furthermore, the detected bounding boxes, which should contain PII, are expected to be more accurate, compared to the general object detection tasks, *i.e.*, the primary evaluation score  $AP^{FP}$  (average precision with penalty of false positive, see Section 4.4) is thresholded with the IoU of 80%.

<sup>1</sup> <https://www.herox.com/SAPAI/>

## 2 Related Work

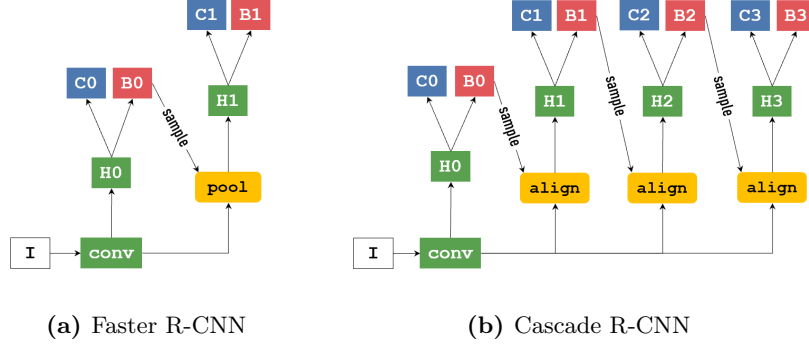
The input images are usually in the format of the cropped handwritten regions in the signature verification competitions [15,19]. Likewise, some handwritten text recognition datasets provide the option of the images labeled with divided lines [16]. This work is expected to bridge the gap between these researches and industrial applications through handwriting localization. Besides, text detection in natural scene images is close to our work. One significant difference between these two tasks is the *target objects*: All texts should be detected in scene text detection task (*e.g.* [18]), while only the handwritten texts in this work. The other difference is the background: The background in scene text detection task is the natural view. In this work, the background is the printed texts and tables on the blank document. Also based on Faster R-CNN [23], Zhong et al. [26] uses LocNet [6] to improve the accuracy in scene text detection, whereas we use Cascade R-CNN [1], the cascade version of Faster R-CNN.

There are two main categories of methods to localize the handwritten regions in the documents. The OCR based approaches recognize and then exclude printed texts. As a result, the unrecognizable parts are believed to be the handwriting. In contrast, the object detection based approaches regard this as a localization task, where the handwriting is the target and all other items (such as printed texts, logos, tables, etc.) are considered as the background. Thanks to the datasets and detection challenges on common objects (*e.g.* [5,13]), a considerable number of novel algorithms about object detection have been productively proposed in the recent years, *e.g.* Faster R-CNN [23], YOLO [21], SSD [14], RetinaNet [12], Cascade R-CNN [1], etc.

Three different approaches submitted to the Data Anonymization Challenge<sup>1</sup> are also briefly introduced in the following sections, including an OCR based approach and two deep learning based approaches (one with YOLOv3 [22], one with Google’s paid cloud service).

**OCR based approaches.** In this section, an example proposal from the challenge<sup>1</sup> is demonstrated. First, the images are sequentially preprocessed, including removing the horizontal and vertical lines, median filtering (to remove salt-and-pepper noises), thresholding and morphological filtering (*e.g.* dilation and erosion). The handwritten parts are then discriminated from the printed ones with respect to the manually chosen features like the heights and widths of the text boxes, text contents and confidence scores recognized by OCR. In the experiments, this approach brings in the results on a par with those using deep learning approaches. However, the robustness and the generalizability of the deep learning approaches are believed to be advantageous.

**Object detection based approaches with deep learning.** Since object detection is an intensively researched area in the field of computer vision, it is natural to directly apply the deep learning algorithms to the handwriting localization task. With the deep learning engine ImageAI [17], the networks like YOLOv3 [22] can be trained in an end-to-end manner. Moreover, some deep learning services like Google’s Cloud AutoML Vision API take it further, managing the training process even without specifically assigning an algorithm.



**Fig. 1:** Network Architectures of Faster R-CNN [23] and Cascade R-CNN [1]. Figures are adapted from [1].

### 3 Method

#### 3.1 Faster R-CNN

Faster R-CNN [23] consists of two modules: the Region Proposal Network (RPN) that proposes rectangular regions containing the desired objects, and the Fast R-CNN detector [7] that predicts the classes and the locations.

The processing pipeline is demonstrated based on Fig. 1a. The input images (I) are first fed into a convolutional neural network (conv), where the shared features are extracted for both RPN and Fast R-CNN detector. Given the shared convolutional feature map of a size  $w \times h \times d$  and the number of the anchors  $k$  for each location in the feature map, the RPN head (H0) transforms it into two proposal features of  $w \times h \times 2k$  (C0) and  $w \times h \times 4k$  (B0) with one *e.g.*  $3 \times 3$  convolutional layer followed by two sibling  $1 \times 1$  convolutional layers.

Now,  $w \times h \times k$  proposals have been generated, each in the form of 6 representative values: 2 objectness scores and 4 coordinate offsets. The higher-scored proposals from B0 are selected as the inputs of the Fast R-CNN detector, together with the shared convolutional feature map. The transform of the proposals' coordinates between the original images and the feature maps is calculated via *e.g.* RoIPool [7] (pool) or RoIAlign [9] (align). The pooled or aligned *region of interest* (RoI) feature map of some fixed size is flattened then projected onto a feature vector via the RoI head (H1). Finally, two vectors of classes (C1) and locations (B1) are obtained by fully connected layers upon the feature vector.

There are two places where multi-task loss functions are calculated: RPN (C0 and B0) and Fast R-CNN detector (C1 and B1). First, log loss is used for both classification tasks (specifically, sigmoid activation function plus binary cross entropy loss for C0 and softmax activation function plus cross entropy loss for C1). Second, smooth L1 loss [7] is used for both bounding box regression tasks (B0 and B1), which is defined as:  $\text{smooth}_{L1}(x) = 0.5x^2$  if  $|x| < 1$  and  $\text{smooth}_{L1}(x) = |x| - 0.5$  otherwise.

### 3.2 Cascade R-CNN

Fig. 1 (adapted from [1]) depicts the differences of these two framework architectures. First, a cascade network is used to train the regressors and classifiers in a multi-stage manner. A four-stage version is illustrated in Fig. 1b, including one RPN stage and three detection stages. Second, the IoU threshold is different for each detection stage, which is increasingly set to  $\{0.5, 0.6, 0.7\}$ . Note that the mentioned IoU threshold does *not* refer to the one in the RPN or the one when calculating *mAP* (mean Average Precision [13]). It is used to define the positive or negative candidates during the mini-batch sampling (**sample** in Fig. 1).

Thanks to the cascade architecture with progressively increasing IoU thresholds for sampling, Cascade R-CNN can accomplish object detection of *high quality*, which is exactly desired in the handwriting localization task to enhance anonymization.

### 3.3 Other Techniques

**Canny Edge Detection.** The gradient intensity based Canny edge detector [2] generates preliminary edges for the following processes. The detected edges might be truncated, *e.g.* under different optical circumstances. Thus, further processes of refinement or extraction are normally applied to the edges detected by Canny.

**Hough Transform.** Through the transform of line parameterizations, straight lines can be efficiently detected by the voting based Hough Transform [4]. The images are usually first processed by edge detectors like Canny, followed by thresholding. Next, each edge pixel  $(x, y)$  in the binary images is represented by  $k$  evenly rotating lines through it in the Hesse normal form:  $r = x \cos(\theta) + y \sin(\theta)$ . In the so-called *accumulator space* of  $(r, \theta)$ , each edge pixel would have  $k$  votes. The peaks of the accumulator space are thus the desired lines. In this work, lines detected by Hough Transform are removed in the preprocessing step to generate clearer input images for the deep learning network.

**Tesseract OCR Engine.** As an open source paradigm OCR engine, Tesseract [24] has been widely used to recognize textual information in the industry. In this work, a Python wrapper for the tesseract-ocr API has been used (<https://github.com/sirfz/tesseractocr>) to detect and eliminate the printed texts in the preprocessing step.

## 4 Experiments

### 4.1 Dataset

The dataset used in this work is the scanned full-page low-quality invoices in the tobacco industry from the 1990s (<http://legacy.library.ucsf.edu/>), which was once served in a document classification challenge [8]. Based on the invoice (or invoice-like) images from the same dataset, the labels and the bounding boxes of names and handwritten notes are manually annotated for the Data Anonymization Challenge<sup>1</sup>.

In total, we have access to 998 gray-scale images with ground-truth labels, which are randomly split to 600, 198 and 200 images as training, validation and testing set, respectively (denoted below as **600train+198val+200test**). The hidden evaluation set consists of 400 images. In this case the training set covers 800 images, the validation set remains unchanged and the testing set covers 400 unseen images (denoted below as **800train+198val+400test**). The sizes of the images are varied around  $700 \times 1000$ , which are resized to  $768 \times 768$ .

## 4.2 Preprocessing

Despite the powerful capability of extracting features *automatically* being one of the benefits when using deep learning algorithms, it is believed that the elementarily preprocessed inputs or their fusions might improve the performance. Intuitively, if some non-handwritten parts could be omitted in the preprocessing step, the following handwriting localization task could be facilitated. Based on this assumption, texts recognized by the OCR engine (**tesseract-ocr** [24]) of high confidence and the straight lines detected by Hough transform [4] are excluded. In the experiments, the threshold confidence for the OCR engine is set to 0.7. The preprocessed images without highly confident textual information or straight lines of tables are denoted as "**pre**" in the following, while the original ones as "**o**".

Besides, the documents usually consist of a white background (of the highest intensity values, *e.g.* 1) and black texts (of the lowest intensity values, *e.g.* 0). As the background is dominant in terms of the number of pixels, it is natural to negate the images to obtain the inputs of sparse tensors, which is believed to make the learning progress more effectively. Given an image ranged in  $[0, 1]$ , the negated image is calculated as the original image element-wise subtracted by 1. With this in mind, the negated original and preprocessed images are denoted as "**o-**" and "**pre-**", respectively.

In addition, the inputs of the deep learning networks can be usually of an arbitrary number of channels. The original and preprocessed images are concatenated to create fused inputs, where the preprocessed layer can be interpreted as an *attention* mechanism, which highlights the most likely regions being the target objects. The concatenated inputs are denoted as *e.g.* "**o/pre**", the two dimensions of which are original and preprocessed images. The influences of the different inputs are investigated in Section 4.6.

## 4.3 Training with Deep Learning Networks

All experiments were running on a single RTX 2080 Ti GPU. The implementation of the deep learning networks are adopted by the open source toolbox from OpenMMLab for object detection: MMDetection [3].

The default optimizer is *Stochastic Gradient Descent* (SGD [11]) with a learning rate of 0.001, a momentum of 0.9, and a weight decay of 0.0001. Since the training set of 600 images is relatively small, the default number of epochs is

set to a relatively large one (200) to make full use of the computational capacity if the training lasts overnight. During the experiments, it is observed that 200 epochs are appropriate (Fig. 4). The **train/val/test** sets are defined as in Section 4.1. Model weights after each epoch with the best results of **val** set are chosen to make predictions on **test** set. The different preprocessing steps are introduced in Section 4.2 and they are compared with Faster R-CNN [23] and Cascade R-CNN [1] in details. Next, additional two deep learning networks, RetinaNet [12] and YOLOv3 [22], have been tested with the preprocessing step which yields the best result on Cascade R-CNN. Detailed experimental results can be found in Section 4.6.

#### 4.4 Evaluation Scores

**IoU.** Intersection over Union of two bounding boxes  $p$  and  $g$  is defined as below:

$$\text{IoU}(p, g) = \frac{\text{Area}\{p \cap g\}}{\text{Area}\{p \cup g\}} \quad (1)$$

**GIoU.** Global IoU of two lists of bounding boxes  $P = \{p_1, p_2, \dots\}$  and  $G = \{g_1, g_2, \dots\}$  is defined as below:

$$\text{GIoU}(P, G) = \frac{\text{Area}\{(p_1 \cap p_2 \cap \dots) \cap (g_1 \cap g_2 \cap \dots)\}}{\text{Area}\{(p_1 \cap p_2 \cap \dots) \cup (g_1 \cap g_2 \cap \dots)\}} \quad (2)$$

**AP<sup>FP</sup>.** Average Precision with penalty of False Positive is the original evaluation score for the handwriting detection used in the Data Anonymization Challenge<sup>1</sup>, which is defined as follows:

$$AP^{FP} = \begin{cases} \frac{|\mathcal{M}^G|}{|\mathcal{G}|} \cdot 0.75^{|\mathcal{P}| - |\mathcal{M}^P|}, & \text{if } |\mathcal{G}| \neq 0; \\ 0.75^{|\mathcal{P}| - |\mathcal{M}^P|}, & \text{otherwise.} \end{cases} \quad (3)$$

In Eq. 3,  $\mathcal{P}$ ,  $\mathcal{G}$ ,  $\mathcal{M}^G$ ,  $\mathcal{M}^P$  denote the sets of predicted, ground-truth, matched *w.r.t.* ground-truth and matched *w.r.t.* predicted bounding boxes, respectively, and  $|\cdot|$  denotes the number of the bounding boxes in this set.

The criterion to call some predicted bounding box  $p_i \in \mathcal{P}$  a match *w.r.t.* the ground-truth bounding box  $g \in \mathcal{G}$  is, when the IoU between  $p_i$  and  $g$  is greater than a threshold  $T$ , *i.e.*:

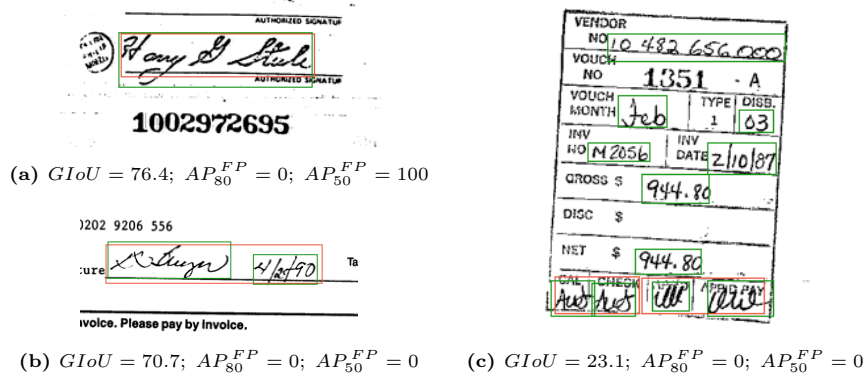
$$\mathcal{M}^G = \{g \in \mathcal{G} \mid \exists p_i \in \mathcal{P} (\text{IoU}(p_i, g) > T)\}. \quad (4)$$

Analogously,

$$\mathcal{M}^P = \{p \in \mathcal{P} \mid \exists g_i \in \mathcal{G} (\text{IoU}(p, g_i) > T)\}. \quad (5)$$

It differs from the popular evaluation score AP (Average Precision) for object detection in COCO [13], where the false positive (FP) has not been particularly

punished. Moreover, considering the potential application on the anonymization enhancement, the IoU threshold  $T$  in the Data Anonymization Challenge<sup>1</sup> is set to 0.8, which is more strict than the common single threshold of 0.5 in COCO. In this work, the results are evaluated both with  $T = 0.8$  and  $T = 0.5$ , which are denoted with  $AP_{80}^{FP}$  and  $AP_{50}^{FP}$ , respectively. In the Data Anonymization Challenge, there is a mechanism of **Bad-Quality**: if an image is marked as **Bad-Quality**, its  $AP_{80}^{FP}$  is assigned with 35%. We therefore record another two evaluation scores  $AP_{80}^{FP*}$  and  $AP_{80}^{FP+}$ , where  $*$  denotes the **Bad-Quality** mechanism is used when calculating  $AP_{80}^{FP}$  and  $+$  denotes the images marked as **Bad-Quality** are excluded when calculating  $AP_{80}^{FP}$ .



**Fig. 2:** Comparison of different evaluation scores (in %). *Green*: ground-truth box; *Red*: predicted box. See Section 4.4 for the definitions of  $GIoU$ ,  $AP_{80}^{FP}$ ,  $AP_{50}^{FP}$ . (a) shows the IoU threshold of 80% might be too strict; (b) shows the  $AP$  family might not effectively indicate the decent quality of the prediction if the ground-truth contains multiple adjacent boxes and a larger one is detected, while  $GIoU$  could; (c) shows an unacceptable prediction where most of sensitive data are exposed, in which case the evaluation score of  $AP$  family (with a high threshold) is desired.

In addition, the overall evaluation score is calculated by averaging all image level scores, which applies to  $GIoU$  and  $AP^{FP}$ . These 3 evaluation scores are illustrated with visual results of ground-truth and predicted bounding boxes in Fig. 2. It is shown that there is no single evaluation score considered as *silver bullet*, especially if the ground-truth annotations are not perfect. In this dataset, *imperfect* is referred to as the fact that the neighboring handwritten regions are sometimes annotated as a single large box, sometimes as multiple separate small boxes. With this in mind, the results are evaluated with the following five scores:  $AP_{80}^{FP}$ ,  $AP_{80}^{FP*}$ ,  $AP_{80}^{FP+}$ ,  $AP_{50}^{FP}$  and  $GIoU$ . Note that the first two evaluation scores ( $AP_{80}^{FP}$  and  $AP_{80}^{FP*}$ ) are eligible in the SAP’s Data Anonymization Challenge<sup>1</sup>.



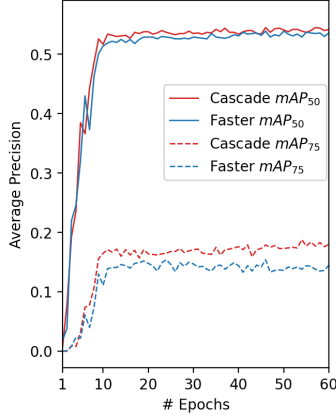
#### 4.5 Postprocessing

The deep learning classifier outputs a confidence score for each corresponding class. This confidence score can be thresholded as a hyperparameter to control the false positive rate in the postprocessing step. It has been observed during the experiments that the best results (*w.r.t.*  $AP_{80}^{FP}$ ) can be achieved, if this confidence is thresholded as 0.8, which is thus chosen as default.

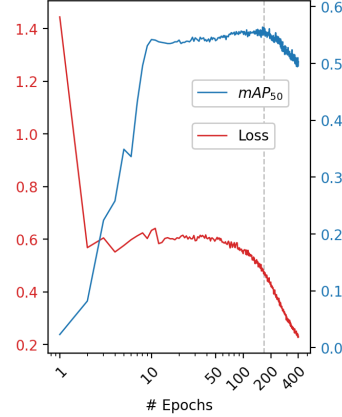
In addition, to reduce the overlapped bounding boxes, some postprocessing steps are applied in the end, which follows the simple criterion: one large box is preferable to multiple small ones. First, all the predicted bounding boxes from each image are sorted in ascending order by their areas. Second, starting from the smallest one, each box is checked if the intersection area over the smaller box area is greater than a threshold (chosen as 0.9). If it is the case, the smaller box is omitted. This postprocessing is applied by default and we observed a minimal improvement, namely around 0.3% in terms of  $AP_{80}^{FP}$ .

#### 4.6 Results

In this section, the experimental results are presented regarding the different preprocessing steps (Table 1), the influences of the mechanism of **Bad-Quality** (Table 1, 3), the comparison of various deep learning networks (Table 2) and the results released on the leaderboard of Data Anonymization Challenge<sup>1</sup> (Table 3). The improved performance from Fast R-CNN and Cascade R-CNN is specifically demonstrated (Fig. 3, 4). Furthermore, the examples of predicted handwritten regions from the **val** set are visualized in Fig. 5, together with the ground-truth bounding boxes.



**Fig. 3:**  $mAP_{50}$  and  $mAP_{75}$  (**val**) of Cascade and Faster R-CNN. The cascade version surpasses Faster R-CNN by a larger margin under the more strict criterion.



**Fig. 4:** Overall loss (bottom; left axis) and  $mAP_{50}$  (top; right axis) during the training. The dashed vertical line indicates the boundary of overfitting after around 170 epochs.

**Preprocessing.** To begin with, different types of preprocessed images as the inputs of the deep learning networks are investigated based on Faster R-CNN and Cascade R-CNN. As shown in Table 1, the preprocessed images alone can bring in worse results, compared to the original ones, while the concatenated preprocessed and original images as inputs ("o/o-/pre-") have achieved the best result *w.r.t.*  $AP_{80}^{FP}$ , which are therefore used as default in the following experiments (Table 2, 3). It is believed that the fused preprocessed images could be served as an *attention* mechanism, which highlights the regions more likely containing the desired objects. Moreover, it is empirically shown in Table 1 that sparse inputs have not always yielded better results, if compared the negated inputs with the original ones.

**Table 1:** Handwriting detection results (in %) with different preprocessing steps as inputs. Images which contain more than 3 detected boxes are marked as **Bad-Quality**. Dataset: 600train+198val+200test. Input abbreviations see Section 4.2. Column-wise best results are made bold.

Network	Input	$AP_{80}^{FP}$	$AP_{80}^{FP*}$	$AP_{80}^{FP+}$	$AP_{50}^{FP}$	$GIoU$
Faster R-CNN	o	34.2	45.4	59.2	65.5	64.6
Faster R-CNN	o-	35.1	45.7	58.2	64.9	65.1
Faster R-CNN	pre	31.3	43.1	55.9	54.0	56.3
Faster R-CNN	pre-	29.6	42.4	54.0	54.6	55.3
Faster R-CNN	o/pre	34.6	43.9	56.3	64.1	64.4
Faster R-CNN	o-/pre-	35.1	44.5	53.3	<b>66.7</b>	65.4
Faster R-CNN	o/o-/pre	37.2	45.6	59.6	63.3	64.9
Faster R-CNN	o/o-/pre-	35.1	45.0	57.4	62.4	64.3
Cascade R-CNN	o	37.7	45.7	56.6	65.6	66.4
Cascade R-CNN	o-	34.7	42.9	49.2	65.1	66.5
Cascade R-CNN	pre	31.9	41.6	47.5	55.4	56.7
Cascade R-CNN	pre-	32.2	42.0	47.6	57.0	58.7
Cascade R-CNN	o/pre	36.3	44.3	56.0	64.4	66.4
Cascade R-CNN	o-/pre-	35.0	44.7	56.6	64.1	64.3
Cascade R-CNN	o/o-/pre	37.2	<b>46.9</b>	<b>60.0</b>	64.4	66.3
Cascade R-CNN	o/o-/pre-	<b>38.3</b>	46.0	56.5	65.0	<b>66.8</b>

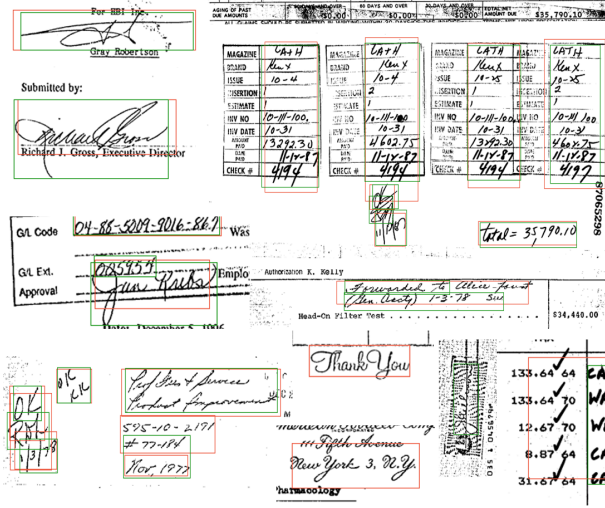
**Mechanism of Bad-Quality.** As introduced in Section 4.4, the mechanism of **Bad-Quality** is provided in the challenge<sup>1</sup>, which has been made full use of with the evaluation scores  $AP_{80}^{FP*}$  and  $AP_{80}^{FP+}$ . It is observed based on the val set that the performance tends to be worse with the increasing number of the detected bounding boxes. With this in mind, all images with more than 3 detected bounding boxes are marked as **Bad-Quality**. The purpose of this mechanism is to raise the abnormal or complicated cases and turn to the manual process, which is practical in the industry. However, if the number of images marked as **Bad-Quality** is too large to maintain the productive advantage of

**Table 2:** Handwriting detection results (in %) using different deep learning networks. Inference speed is tested on a single RTX 2080 Ti GPU. Images which contain more than 3 detected boxes are marked as Bad-Quality. Dataset: 800train+198val+400test. Column-wise best results are made bold.

Network	Inference	$AP_{80}^{FP}$	$AP_{80}^{FP} *$	$AP_{80}^{FP} +$	$AP_{50}^{FP}$	$GIoU$
YOLOv3	42 fps	36.6	43.2	47.4	61.0	62.2
RetinaNet	11 fps	27.9	40.8	44.4	51.7	54.9
Faster R-CNN	11 fps	37.1	45.3	57.2	62.1	66.6
Cascade R-CNN	10 fps	<b>41.8</b>	<b>47.5</b>	<b>57.5</b>	<b>66.9</b>	<b>68.2</b>

**Table 3:** Comparison with the leaderboard. *OCR*: Tesseract with manual engineering. *Service*: Google’s Cloud API. \* and † denote YOLOv3 results from the leaderboard and ours, respectively. BQ: if Bad-Quality is used.

Method	$AP_{80}^{FP}$	BQ
YOLOv3 *	26.3	✗
OCR	37.5	✗
Service	42.5	✗
YOLOv3 †	36.6	✗
YOLOv3 †	43.2	✓
Cascade	41.8	✗
Cascade	47.5	✓



**Fig. 5:** Visual results of cropped handwritten regions from val set (Cascade R-CNN with "o/o-/pre-"). *Green*: ground-truth box; *Red*: predicted box.

machines, the evaluation scores  $AP_{80}^{FP} *$  and  $AP_{80}^{FP} +$  might bring in pseudo good results. Therefore, the number of images marked as Bad-Quality is loosely limited up to 50% of all images. Conclusively, the mechanism of Bad-Quality is believed to be a flexible trick to deal with the hard cases.

**Faster and Cascade R-CNN.** Not surprisingly, it is shown in Table 1 that Cascade R-CNN outperforms Faster R-CNN in general except for  $AP_{50}^{FP}$ , as Cascade R-CNN focuses on the object detection of higher quality and might perform worse than Faster R-CNN in terms of  $AP_{50}^{FP}$ . As shown in Fig. 3, the cascade version surpassed Faster R-CNN by a larger margin under the more strict criterion, when compared  $mAP_{75}$  to  $mAP_{50}$  (using COCO’s evaluation

scores [13] for brevity) in the `val` set. In addition, as depicted in Fig. 4, the training progress has been overfitted after around 170 epochs, from where the loss values and  $mAP_{50}$  start decreasing. Thus, the chosen 200 epochs during all the experiments are appropriate.

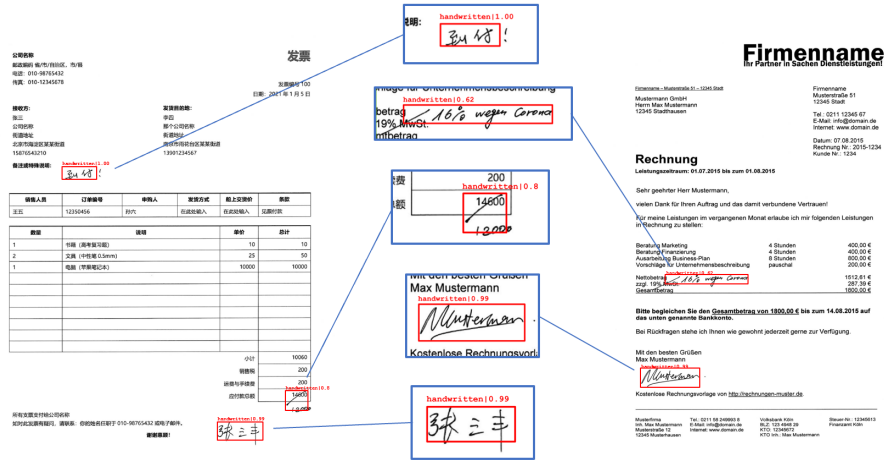
**Comparison with other approaches.** Some of the popular deep learning networks for object detection are compared in Table 2, including YOLOv3 [22], RetinaNet [12], Faster R-CNN [23] and Cascade R-CNN [1]. They are implemented with [3]. Darknet-53 [20] is used as the backbone for YOLOv3 and ResNeXt-101 [25] for the other three. The dataset used in this experiment, `800train+198val+400test` (see Section 4.1), is identical with the one used in the leaderboard. The images are preprocessed to the form of "o/o-/pre-" as described in Section 4.2. The results show that Cascade R-CNN outperforms other networks, with the trade-off regarding the inference speed on a single RTX 2080 Ti GPU due to the extra computational overhead though. As showcased in Table 3, the results of our approaches are compared with those submitted to the leaderboard. The first three rows are the results from the leaderboard, and the following four rows are our approaches. Our best achieved result (with Cascade R-CNN and BQ) has surpasses previous submissions on the leaderboard. Without considering the mechanism of BQ, however, the paid Google AutoML Vision API is slightly more advantageous (by 0.7%). Besides, it is noticeable that our YOLOv3 result (implemented by [3]) has outperformed the version submitted to the leaderboard (implemented by [17]) by more than 10% in terms of  $AP_{80}^{FP}$ .

#### 4.7 Generalizability

English is the vast majority of the languages used in the dataset. Other languages such as Dutch or German are also included. However, the deep learning network is not expected to recognize the discrepancy of different languages. It is natural to categorize the languages using Latin alphabets indiscriminately. In this section, it is tested if the trained model works on the redacted real-world images in *foreign* languages.

Fig. 6 illustrates two toy examples of fictitious and unseen invoices to evaluate the generalizability of the trained model. The model used to localize the handwritten regions is Cascade R-CNN. It is noteworthy that the language in the left image in Fig. 6 is Chinese, which can be considered as a foreign language in the dataset. Analogous to the German invoice demonstrated in the right one, the handwritten regions of both images are accurately detected as desired. The generalizability of the R-CNN family has also been observed by [26] during the text detection in natural scene images.

It is believed to be beneficial in the industry, if the model can be trained once and applicable to various cases. Additionally, it has also raised the common question of what the deep learning network has learned. In this case, it is supposed that the *irregularity* might be learned to discriminate the printed and handwritten texts.



**Fig. 6:** Test of generalizability on toy examples with fictitious and unseen invoices. The handwritten regions are accurately localized from the images in Chinese and German.

## 5 Discussion

### 5.1 Conclusion

In this work, we present an object detection based approach to localize the handwritten regions, which is effective, fast and applicable to the unseen languages. First, the influences of the preprocessing steps are investigated. It has been empirically found that the fused concatenation of original and preprocessed images as the inputs can achieve the best performance. Second, different deep learning networks are compared. It is noticeable that the improvement from Faster R-CNN to Cascade R-CNN can be reproduced and the *high quality* characteristic of the cascade version suits the problem of the handwriting localization well. The results of our approaches can be served as a baseline of deep learning approaches in the handwriting localization problem. At last, the generalizability of the deep learning approach is impressive. The learned model is capable to successfully detect the handwritten regions on the real-world unseen images, even for those in the unseen language of Chinese. We believe it is of great interest both for the future research and for the industrial applications.

### 5.2 Outlook

As showcased in Fig. 5, some printed cursive texts are also detected as handwriting. It remains challenging to distinguish such nuanced discrepancies. Furthermore, apart from the object detection approaches, other proposals in the field of computer vision can also be adopted to differ the handwritten texts from the printed ones. One example is the anomaly detection, where the printed

texts can be considered as the *normal* instances, since they are more regularly shaped. Thanks to the algorithms like *variational autoencoder* (VAE) [10], it is also promising to accomplish such tasks in a semi-supervised or even unsupervised manner. The other benefit of using the algorithms like VAE is that the learned intermediate representations can also be exploited to synthesize the artificial signatures, further enhancing the anonymization without eliminating the existence of such entities.

## References

1. Cai, Z., Vasconcelos, N.: Cascade r-cnn: high quality object detection and instance segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019)
2. Canny, J.: A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence* (6), 679–698 (1986)
3. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155* (2019)
4. Duda, R.O., Hart, P.E.: Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM* **15**(1), 11–15 (1972)
5. Everingham, M., Eslami, S.M.A., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision* **111**(1), 98–136 (Jan 2015)
6. Gidaris, S., Komodakis, N.: Locnet: Improving localization accuracy for object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 789–798 (2016)
7. Girshick, R.: Fast r-cnn. In: *Proceedings of the IEEE international conference on computer vision*. pp. 1440–1448 (2015)
8. Harley, A.W., Ufkes, A., Derpanis, K.G.: Evaluation of deep convolutional nets for document image classification and retrieval. In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. pp. 991–995. *IEEE* (2015)
9. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: *Proceedings of the IEEE international conference on computer vision*. pp. 2961–2969 (2017)
10. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013)
11. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. *Neural computation* **1**(4), 541–551 (1989)
12. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: *Proceedings of the IEEE international conference on computer vision*. pp. 2980–2988 (2017)
13. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: *European conference on computer vision*. pp. 740–755. *Springer* (2014)
14. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: *European conference on computer vision*. pp. 21–37. *Springer* (2016)

15. Malik, M.I., Liwicki, M., Alewijnse, L., Ohyama, W., Blumenstein, M., Found, B.: Icdar 2013 competitions on signature verification and writer identification for on-and offline skilled forgeries (sigwicom 2013). In: 2013 12th International Conference on Document Analysis and Recognition. pp. 1477–1483. IEEE (2013)
16. Marti, U.V., Bunke, H.: The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition* **5**(1), 39–46 (2002)
17. Moses, Olafenwa, J.: Imageai, an open source python library built to empower developers to build applications and systems with self-contained computer vision capabilities (mar 2018–), <https://github.com/OlafenwaMoses/ImageAI>
18. Nayef, N., Patel, Y., Busta, M., Chowdhury, P.N., Karatzas, D., Khelif, W., Matas, J., Pal, U., Burie, J.C., Liu, C.I., 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). pp. 1582–1587. IEEE (2019)
19. Ortega-Garcia, J., Fierrez-Aguilar, J., Simon, D., Gonzalez, J., Faundez-Zanuy, M., Espinosa, V., Satue, A., Hernaez, I., Igarza, J.J., Vivaracho, C., et al.: Mcyt baseline corpus: a bimodal biometric database. *IEE Proceedings-Vision, Image and Signal Processing* **150**(6), 395–401 (2003)
20. Redmon, J.: Darknet: Open source neural networks in c. <http://pjreddie.com/darknet/> (2013–2016)
21. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 779–788 (2016)
22. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018)
23. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence* **39**(6), 1137–1149 (2016)
24. Smith, R.: An overview of the tesseract ocr engine. In: *Ninth international conference on document analysis and recognition (ICDAR 2007)*. vol. 2, pp. 629–633. IEEE (2007)
25. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431* (2016)
26. Zhong, Z., Sun, L., Huo, Q.: Improved localization accuracy by locnet for faster r-cnn based text detection. In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. vol. 1, pp. 923–928. IEEE (2017)