*Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, pages 4639–4646
Marseille, 11–16 May 2020


# Building OCR/NER Test Collections

**Dawn Lawrie, James Mayfield, David Etter**
HLTCOE at Johns Hopkins University
Baltimore, MD 21211
{lawrie,mayfield}@jhu.edu, etterd@gmail.com

**Abstract**

Named entity recognition (NER) identifies spans of text that contain names. Many researchers have reported the results of NER on text created through optical character recognition (OCR) over the past two decades. Unfortunately, the test collections that support this research are annotated with named entities *after* optical character recognition (OCR) has been run. This means that the collection must be re-annotated if the OCR output changes. Instead, by tying annotations to character locations on the page, a collection can be built that supports OCR and NER research without requiring re-annotation when either improves. This means that named entities are annotated on the transcribed text. The transcribed text is all that is needed to evaluate the performance of OCR. For NER evaluation, the tagged OCR output is aligned to the transcription, and modified versions of each are created and scored. This paper presents a methodology for building such a test collection and releases a collection of Chinese OCR-NER data constructed using the methodology. The paper provides performance baselines for current OCR and NER systems applied to this new collection.

**Keywords**: OCR and NER test collection, Chinese named entities, OCR evaluation, NER evaluation

## 1. Introduction

Named entity recognition (NER) is the automatic recognition of spans of text as name mentions, and the categorization of those spans into a predefined set of types such as people, places, organizations, software, etc. Most NER research is performed using text that is created in digital form, such as newswire, blog posts, etc. When the text is derived from another medium, such as speech or images, it can be degraded in ways not usually seen in digital text. This paper presents a collection that can help answer the questions "how can one evaluate NER when the input is corrupted?" and "what is the impact of non-digital source medium on NER scores?"

The standard pipeline for performing NER on text images begins with layout analysis and optical character recognition (OCR). If the image is a page from a newspaper, it might include multiple articles, each with its own headline, perhaps laid out over multiple columns. Finding names on the page starts by recognizing characters and their locations on the page. Next, a reading order for those characters is established. The text can then be tokenized and split into sentences. Finally, an NER system labels the tokens by type. Until these inference steps can be handled jointly, a test collection that facilitates both optical character recognition (OCR) and NER research must account for the fact that OCR operates at the character level, whereas NER systems generally assume that they are processing tokenized text one sentence at a time.

Studying the performance of NER on documents that are derived from OCR is not new (Miller et al., 2000). Most such research has been driven by historic collections such as newspapers from the $18^{th}$ and $19^{th}$ centuries (Neudecker, 2016; Kettunen and Ruokolainen, 2017; Galibert et al., 2012; Packer et al., 2010). Collections have been built in English and several other European languages. One of the drawbacks of these existing collections is that NER annotations were done on OCR output. This means that new annotations would be required if a different OCR engine were used, because the recovered text would likely change. This is particularly problematic if the new OCR system introduces new tokens or omits tokens, because the token positions of named entities would change. The collection presented in this paper addresses this shortcoming by annotating transcribed text to include both named entity markings and position information. Thus, this collection supports both OCR and NER research independently as well as research into the combined task.

An OCR/NER test collection includes four types of annotation: transcribed characters associated with locations on the page; reading order of the characters; sentence and token boundaries; and NER tags on the tokens. The character transcriptions alone can be used for OCR evaluation. The test collection can also be used for NER evaluation independent of OCR by ordering the transcriptions based on reading order and breaking the text at sentence boundaries. The multi-use aspect of this collection sets it apart from prior collections created to support NER research over digitized text. By annotating named entities on the underlying ground truth text, and maintaining or deriving a mapping from that ground truth to locations on the page, the collection can be used to evaluate improved OCR techniques.

Such a test collection can be created for any language. We used the Chinese newspaper *Renmin Ribao* for this collection; to our knowledge, this is the first OCR/NER collection in this language. Chinese is particularly challenging for OCR because of the number and complexity of its characters. It is challenging for NER because of the lack of spaces between words, flexibility in word order, and the use of common nouns in names.

## 2. Collection Creation Methodology

First we will describe a generic process for building a collection that supports both OCR and NER research as well as their combination. Then we will describe how we implemented the process to create a new collection in Chinese.

The first step is to box each row of text in an image (that is, identify rectangles on the page that contain single rows

of characters), and then to transcribe the text in that box. The second step assigns a reading order to the boxes. Next, sentence and token boundaries are identified. Finally, NER annotations are added. These steps are outlined in Figure 1.

Our Renmin collection source material consisted of newspaper pages published in pdf form. The advantage of this input was that the "box and transcribe" step can be supported by the Linux utility `pdftotext`,[1] which can be used to extract the text. The utility provides locations of the text given a particular `dpi` as input.[2] Unfortunately, differences between the text in the image and the text that is extracted using `pdftotext` occur for several reasons. First, some characters are repeated in multiple boxes. Because their location is the same, they appear as a single character on the printed page. Second, sometimes the boxes are recognized, but the content is garbled with Latin characters, numbers, and punctuation. This second issue is readily apparent even to a non-Chinese speaker, but does require that a person transcribe the text in those boxes. On the other hand, the repeated characters can be detected programmatically because the boxes overlap, and therefore, the boxes can be merged. Two other features occasionally occurred in our source documents. One is that the text from multiple columns sometimes appears in a single box, which makes determining the reading order of the boxes challenging. Second, one box would occasionally overlay another box, thereby introducing new characters in the middle of the second box. Although these issues could be dealt with programmatically, we had annotators fix the problems while they transcribed boxes containing garbled characters.

Once all the boxes have been transcribed, the reading order of the boxes needs to be identified. The `pdftotext` tool does output a reading order; however, it is only generally accurate at the column level. Titles and captions can appear at any point in the box ordering. Our in-house annotation tool for boxing and transcription provides a mechanism to capture reading order by having an annotator mouse over a series of boxes; the order the boxes are moused over specifies the reading order.

The third step is to identify token and sentence boundaries. This is done because NER is generally performed on tokens at the sentence level, and names never span sentences. For Chinese text, due to the difficulty of identifying word boundaries, we treat each character as a separate token. If NER annotations are gathered over boxes rather than sentences, an annotator needs to be able to indicate that a name spans multiple boxes. Although this can be built into an NER annotation tool, it may be only obvious that a token is part of a name mention in one of the two boxes. The failure to recognize both halves of the name leads to many inconsistencies. For this collection, annotators were instructed to annotate the beginning of sentences by selecting the first character in a sentence, title, caption, or list. It was not necessary to identify token boundaries because we perform NER in Chinese at the character level.

Finally in Step 4, sentences are annotated to identify named entities. For this collection, an extended tag set is used. This set includes the standard types: person (PER); organization (ORG); geo-political entity (GPE); and natural location (LOC), as well as: facility (FAC); named event (EVNT); vehicle (VEH); computer hardware and software (COMP); chemical (CHEM); and weapon (WEAP). As in the original CoNLL annotations, a miscellaneous (MISC) type captures all names not covered by the other types. Annotated subtypes include: Commercial organization (COMM); political organization (POL); airport facility (AIR); and government facility (GOVT). Finally some non-names were included: date (DATE), time (TIME), money (MONEY), and title (TITLE).

Because the collection must support both OCR and NER, the collection format is a tab-separated file with columns for OCR-related and NER-related values. The first column of data is the token, and the second column is the named entity tag in BIO format. In this format, the first token in a named entity mention is tagged `B-<TYPE>`; subsequent tokens in the name are tagged `I-<TYPE>`. Non-names are tagged with `O`. Blank lines demarcate sentences. The file is organized in reading order. These first two columns are the standard columns in most NER collections.

Three other piece of information are necessary to support OCR evaluation as well as NER over OCR evaluation: the location of the box on the page; the page on which the box occurs; and the number of the token within its box. New columns have been added to account for this additional information. The page and token offset are encoded in a token ID, which is recorded in the third column. The format of the token ID is `<page-id>-<box-id>-<token-offset>`. The `page-id` includes the collection name, year, month, day, and page number separated by underscores. The `box-id` is randomly assigned to all tokens that appear in the same box as indicated by their coordinates on the page, and the zero-based `token-offset` is the offset of the token within the box reading from left to right or top to bottom depending on the orientation of the box. Token offsets are assigned left to right even if the tokens in the box would be read right to left.

The fourth column in the file records the ID of the previous token. This explicitly encodes the reading order even though the collection itself is presented in reading order. A token that begins an article has `None` as the previous token ID.

The final four columns of the file are the coordinates for the bounding box in which that token occurs. In particular the fifth and sixth columns are the $x$ and $y$ coordinates of the upper left corner of the bounding box, while the seventh and eight columns are the $x$ and $y$ coordinates of the lower right corner of the bounding box.

## 3. The Renmin OCR/NER Collection

We used the above methodology to create a reusable OCR-NER collection over Chinese text documents. The Renmin OCR-NER collection consists of the June 1-4, 2018 editions of the Renmin Ribao Newspaper,[3] comprising seventy-two pages. There are a total of 427,885 tokens

[1] Version 0.62.0

[2] A `dpi` of 216 is used in this collection.

[3] `http://paper.people.com.cn/`

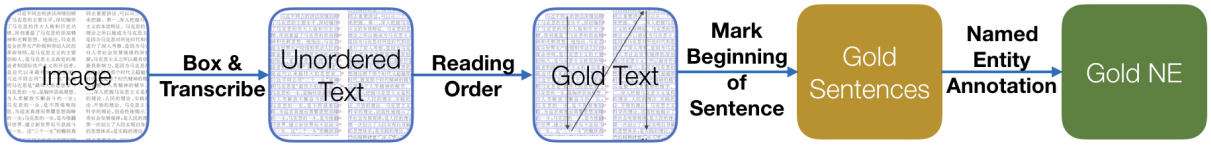


Figure 1: Annotation Steps Recommended by Annotation Methodology

(separate Unicode characters, mostly Chinese). The collection is annotated at the character level to avoid the imprecision introduced by word segmentation in this unspaced script, where even human annotators may disagree on precisely which characters make up an individual word. The collection consists of 10,364 sentences and 16,065 entities. The entities are distributed over the types as is shown in Figure 2. As can be seen, over half of the names are either person or GPE. All entities are distinct; no overlapping or nested entities were allowed. In cases of nested entities, the entity with the greatest extent was preferred.

The collection has been split by article based on number of entities, reserving 80% of the entities for training, 10% for development, and 10% for evaluation. There are some rare types such as AIR and GOVT that only occur in the training set. Other types including COMP and TIME have one example in the test partition, but no examples in the development partition. We retained these types for consistency with future collections that use the same tag set. However, types that are insufficiently attested could be changed to MISC rather than being kept as an independent type. The effect of such a modification on scores is not reported in this paper. The dataset can be obtained from `https://github.com/hltcoe/cmn-renmin-ocr-ner-dataset`.

The description provided in Section 2 represents the process we developed as the best way to produce clean OCR-NER collections while minimizing annotation effort. The annotation of this particular Renmin corpus deviated from the basic process in several ways. This section outlines these deviations and reports the number of annotation hours required for each of the steps.

We initially developed three annotation tasks: reading order, sentence boundary detection, and named entity tagging. These three tasks were done in order as is shown in Figure 3. To collect the reading order, annotators were shown an image of a newspaper page with blue boxes around the text and box numbers in red that coincided with the output of `pdftotext`. Figure 4 shows a portion of one page. This figure includes a few of the artifacts found in the `pdftotext` output. For instance, the top bounding box is not closed because the box includes the title of the article to the right of the one displayed. The third line of the article interferes with proper ordering of the two columns. Several boxes contain overlapping characters; box numbers are indicated by two integers separated by a dash in these cases. Finally this is an article that continues on a different page; the continuation instructions appear in Box 258.

Using a form-fill-style interface, annotators were asked to indicate the number of the box that precedes the box being annotated. Annotators were ask to select "None" if the box represented the beginning of an article, beginning header information, or a caption for a picture that did not clearly belong with an article. If another box on the current page was the correct preceding box, than the annotator selected that option and entered the number. If the preceding box occurred on a prior page, a "different page" option was selected and the prior page number was noted. A comments box allowed annotators to alert the collection designer to such things as a box spanning multiple columns, as seen in Figure 4. To break up the work, each task asked annotators to label about ten boxes at a time in a locally installed Mechanical Turk-like service known as Turkle.[4] Because it was rare for an article to span multiple pages, no new interface was developed to address this issue. Instead, annotators used a Google Doc to record which box on a different page was the correct preceding box.

Because a linear reading order is needed, errors in reading order were obvious to the collection builders. Most of the errors centered around headlines, especially vertical headlines that commonly occur in Chinese newspapers. These headlines were frequently not clustered numerically, so different annotators annotated different parts of the headline. In addition, headlines may consist of multiple parts that can be read in any order. For these reasons a single annotator determines the reading order for each page.

These inconsistencies led us to seek an alternative to the approach to annotating reading order described in Section 2. We developed a second interface in which an annotator simply mouses over each of the boxes in reading order to assign the reading order to the boxes on a page. Using a single annotator for an entire page eliminates the reading order inconsistencies observed in our first approach. In total the annotators spent approximately 88 hours annotating reading order.

For sentence boundary identification, annotators were shown six boxes at a time and asked to identify each character that begins a sentence. This task required a total of 145 hours to complete. No quality control was performed, although a second pass could be used to correct problems where opening punctuation is associated with the prior sentence.

Finally, annotators were asked to annotate name mentions. We use the Dragonfly annotation tool (Lin et al., 2018) for this purpose. Annotators are provided a set of named entity tags from which to choose. The annotation tool allows each token sequence to be assigned a type, with 'O' being assigned to each token that is not part of a named entity mention. Initial annotations were done at the box level. To

[4] `https://github.com/hltcoe/turkle`

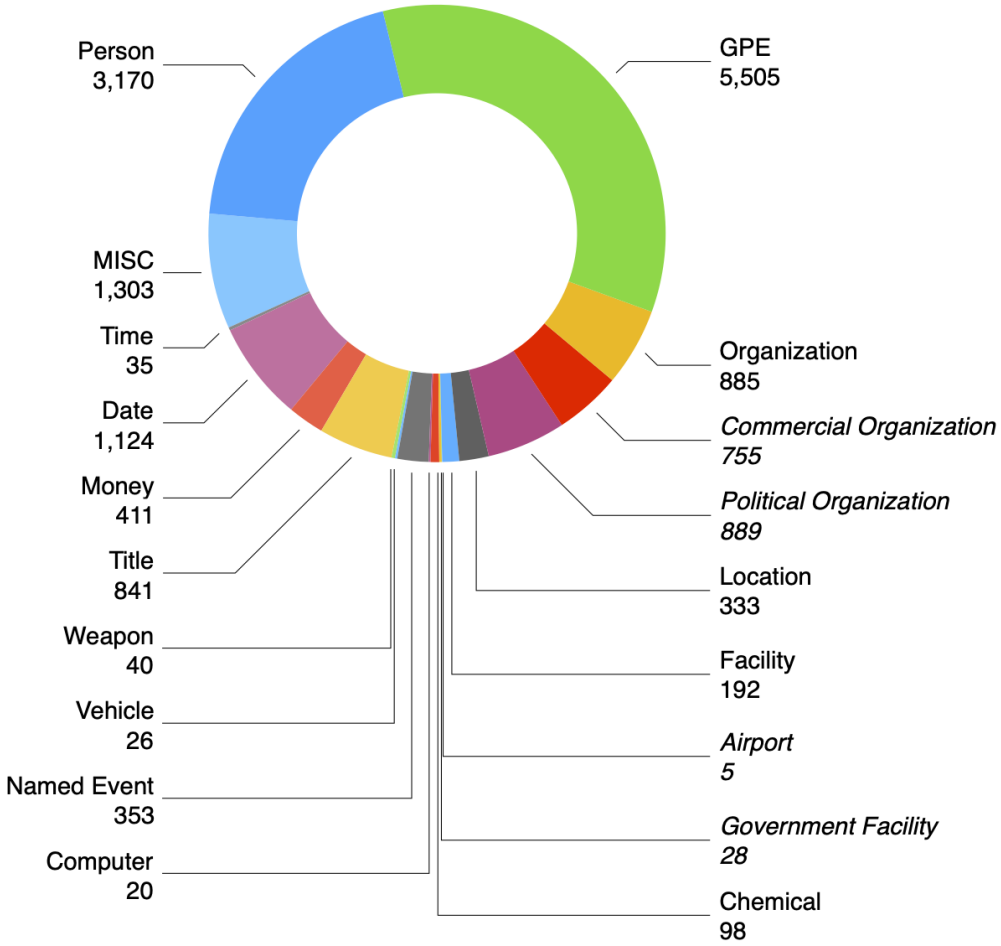


Figure 2: NER types and number of instances

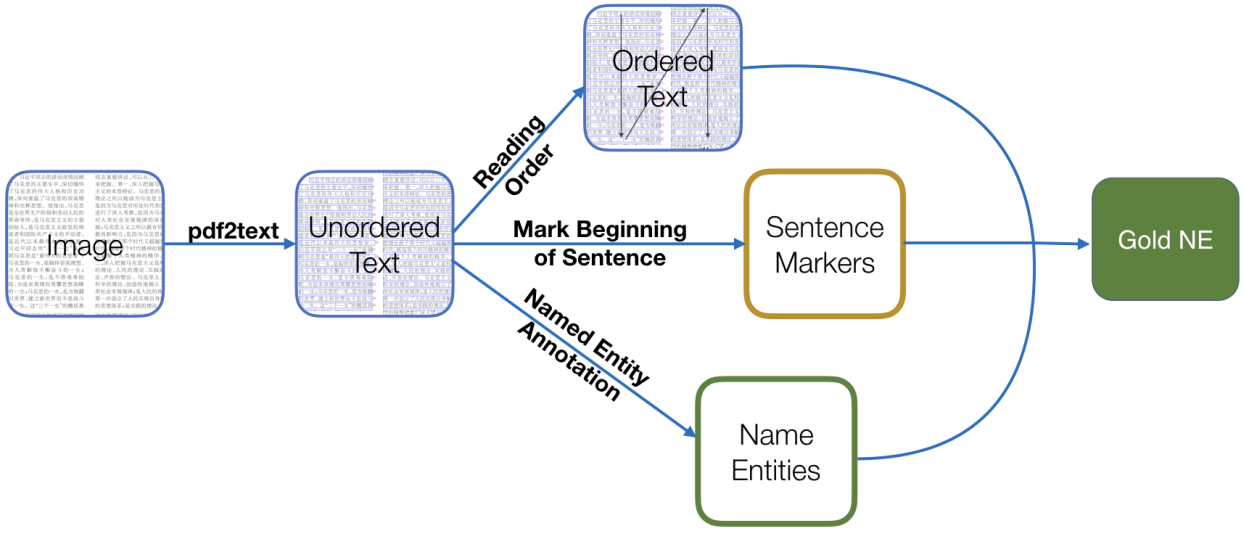


Figure 3: Instantiation of Recommended Annotation Process Used to Produce Renmin Collection

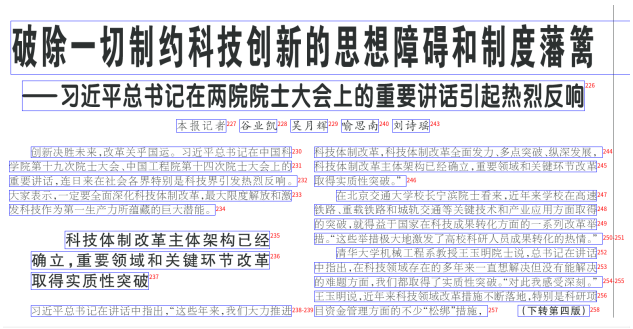


Figure 4: Newspaper page with overlaid boxes

handle cases where the starting token in a box should be labeled with an I-tag vecause the name began in a previous box, a pseudo-token was added at the start of each box. This artificial token was selected as the first token of names when the beginning of the name began in the prior box. After annotation the artificial token was stripped. With reading order annotations, the name would be appended to the prior box. Unfortunately, sometimes the annotator of the prior box failed to mark the beginning of the name or assigned a different type leading to an inconsistent labeling. To account for this, a second annotator reviewed the inconsistent annotations after the other annotation tasks were complete.

Named entity annotation followed two procedures. Part of the collection (comprising 40% of the tokens) was annotated by a single annotator and then put through a quality control process after reading order and beginning of sentence annotation was complete. The second annotator reviewed the names. The remainder of the collection was initially annotated by two annotators so that inner-annotator agreement could be calculated. Overall annotators agreed on the label for 94% of tokens; however, when considering only tokens where at least one annotator marked the token as part of a name, annotators agreed on the label of a token for 53% of the tokens. The overall Cohen's Kappa statistic, a common measure of inner-annotator agreement, is 0.696. The doubly annotated sentences were only reviewed when annotators disagreed, or there was an inconsistency in the names such as a name lacking a begin token. The total annotation effort for this task, including reviewing, took 1,444 hours.

## 4. Evaluating NER over OCR

The collection can be used in at least three ways. First, for OCR evaluation, the box coordinates can be used to determine where text occurs and what characters should be found in a box. Given that the distributed collection is organized in a standard NER format, and that the text is divided into articles, there are a few instances where a single box is split across articles because it contains text from each of the articles. Thus, use of the data for evaluation of OCR systems will require the researcher to assemble the contents of each box, possibly from more than one location in the collection. These instances can be identified by looking at the box offset and character position within the box, both of which are encoded in the `token-id`.

Second, for NER evaluation, columns after the first two columns can be ignored. We used the `Conlleval` tool from the CoNLL 2003 evaluation (Tjong Kim Sang and De Meulder, 2003) to score all runs. `Conlleval` calculates precision, recall, and $F_1$-score across all entities and across each entity class.

Third, to support evaluation of NER over the output of OCR, additional steps are required. The remainder of this section describes how this is supported. Like most NER scorers, `Conlleval` assumes that the system output and ground truth files contain exactly the same tokens, and that the tokens in the two files appear in the same order. Neither of these assumptions holds for NER over OCR. First, an image has no inherent reading order; tokens in the ground truth may be in a different order than that of the OCR output. To cope with this, the OCR output and the ground truth must be aligned prior to scoring, using the location of the text in the image. Second, OCR may insert or delete tokens. This means the number of tokens in the ground truth may not match the number of tokens the NER system has labeled. To cope with this, new token-aligned versions of the gold (ground truth) and NER (OCR system output) files must be produced that do contain the same number of tokens.

The production of an NER score for an image is shown in Figure 5. NER-labeled OCR output is produced from an image by running OCR over it and then using an NER system to label the OCR output.

The first step to scoring that output is to create a modified gold file that is re-ordered to match the order found in the text ingested by the NER system. To do this, the OCR system must identify which ground truth location corresponds to each OCR emission. Then a modified gold file is produced that matches the ordering of the text.

Once the ground truth file reflects the received reading order, the two sequences are aligned using Levenshtein distance. This algorithm produces a minimal list of edit operations that will transform the OCR output into the reference. We allow the standard three operations for this transformation: `substitute`, `insert`, and `delete`. `Substitute` has no impact on the one-to-one correspondence of tokens required for NER evaluation; in contrast, `insert` and `delete` break the alignment. To restore the alignment, each `insert` into the OCR output is aligned to a null token added to the gold file. Likewise, each `delete` from the OCR output is aligned to the corresponding ground truth token by inserting a null token into the NER over OCR output at the corresponding position. All null tokens that appear in the middle of a named entity are labeled with the type of that entity. All other null tokens are labeled as non-entities (commonly represented by an '`O`' tag). This process generates new gold and OCR files that can be scored using standard methods.

## 5. Baseline Results

We exercise our collection in the three scenarios that the collection supports and report baseline results as references for future research.

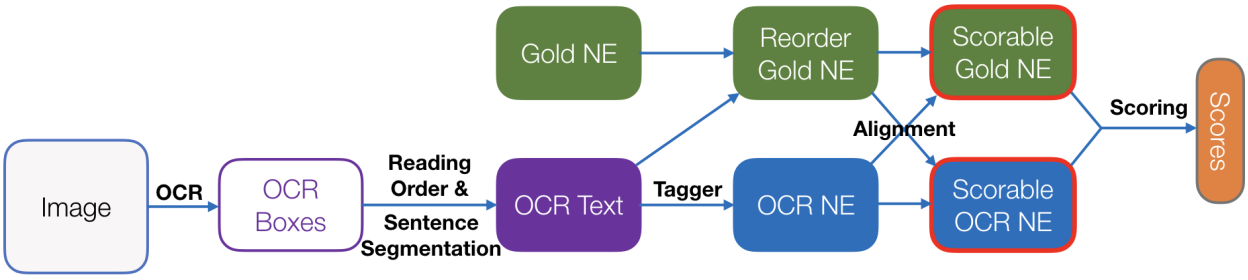


Figure 5: Experimental Process

### 5.1. OCR System

The OCR system used in our experiments is an end-to-end neural model (Rawls et al., 2017) that combines a convolutional neural network (CNN) with a long-short-term memory (LSTM) recurrent network. Input line images to the CNN are resized to 30 pixels high with a variable width that maintains the original aspect ratio. The network is based on the VGG architecture (Simonyan and Zisserman, 2014) and consists of seven convolution layers, where each 2D-convolution uses a 3x3 filter kernel, followed by BatchNorm and RELU. We apply fractional max-pooling (Graham, 2014) after layers 2 and 4 using a ratio of .5 for height and .7 for width. This pooling ratio allows us to keep more of the features in the width dimension. A fully-connected bridge layer joins the output of the CNN to the 3-layer bidirectional LSTM. A final fully-connected layer maps the LSTM output to the size of the training character set. The sequence-to-sequence problem is trained using connectionist temporal classification (CTC) loss (Graves et al., 2006), which allows segmentation-free training. We use an Adam optimizer and set an initial learning rate of 1e-3, which is reduced by a factor of ten as the loss plateaus.

Our Chinese model is trained using data gathered and transcribed by Yet One More Deep Learning Enterprise (YOMDLE).[5] This data set includes document images gathered from multiple domains and unconstrained settings. Unconstrained document images are taken from challenging settings that often include complex backgrounds, multiple fonts, lighting changes, and occlusions. Examples include images of newspapers, magazines, Web pages, and maps. The Chinese training set includes approximately 1,000 document images with over 13,000 transcribed line images. The average line height is 52 pixels and the average width is 503 pixels. At training time we apply a random set of image augmentations that include blur, noise, sharpen, emboss, pixel dropout, channel inversion, brightness, hue, saturation, contrast, and gray-scale. The model was evaluated on the SLAM data set (Etter et al., 2019), which includes 500 documents and over 10,000 line images, drawn from a similar domain as the YOMDLE set. Our system is evaluated using a character error rate (CER), scoring 13.3 CER on the evaluation set. On the Renmin Collection, our system scored 3.0 CER on the training set, 2.4 CER on the development set, and 2.3 CER on the test set.

[5] http://yomdle.com/

| Hyperparamaters | |
|---|---|
| BiLSTM layers | 1 |
| BiLSTM hidden size | 256 |
| BiLSTM dropout | .5 |
| Optimizer | adafactor |
| Gradient clipping | 1.0 |
| Learning rate scheduler | cosine decay |
| BERT layers used | $-4, -3, -2, -1$ |
| Weight decay | .005 |
| Mini batch size | 8 |

Table 1: Default hyperparameters in the baseline NER model

### 5.2. NER System

The baseline results come from a Neural NER architecture with the following features. It is a common Bi-LSTM-CRF model like many sequence-to-sequence NER systems (Huang et al., 2015), which includes a stacked bi-directional recurrent neural network with long short-term memory units and a conditional random field decoder (similar to Chiu and Nichols (2016) without the character-level CNN). We combine this system with BERT (Devlin et al., 2018), which is a stack of bi-directional transformer encoders. As is done in Devlin et al. (2018), we use the representation of the first sub-token as the input to the token-level classifier over the NER label set. We keep the BERT frozen during training and testing, feeding the text into BERT and concatenating its final four layers as an input to our Bi-LSTM-CRF. Table 1 shows the hyperparameters used for our experiments. We did not perform a hyperparameter search. We use Google's Chinese BERT, which has the following properties: 12-layer, 768-hidden, 12-heads, and 110M parameters.[6]

Table 2 reports the overall $F_1$-score for the development data for both the transcriptions and the OCR output, while Table 3 contains the overall $F_1$-score for the test data. Training was done on the transcribed data. A random partition was used for validation during training.

Table 4 presents performance by entity type for NER over OCR boxes. In general the system performs better for the

[6] https://storage.googleapis.com/bert_models/2018_11_03/chinese_L-12_H-768_A-12.zip

| | OCR Boxes | Ground Truth Sentence |
|---|---|---|
| Transcriptions | | 70.92 |
| OCR ouput | 66.54 | 69.70 |

Table 2: Development set $F_1$-scores averaged over all types

| | OCR Boxes | Ground Truth Sentence |
|---|---|---|
| Transcriptions | | 68.51 |
| OCR ouput | 64.47 | 67.48 |

Table 3: Test set $F_1$-scores averaged over all types

types with greater training data such as person and GPE. The system also performs well on numerical types such as dates, time, money, due to their relative lack of variation. The system struggles with general organizations more than the subtypes of organization, both commercial and political. The system also performed less well on types with fewer training examples.

## 6. Synthetic Training and Test Sets

Given the resources required to create collections that support NER over OCR research, we designed a methodology to use an existing NER collection to create synthetic images that can be then be processed by the full OCR/NER stack. Text from an existing NER collection is laid out as an image reflecting the desired document configuration (*e.g.,* break the text into same-length sequences and stack them, as if they were a newspaper article). In this way the position(s) on the page and entity label of each character is known. This approach requires no new annotation.

When the document source is a labeled NER collection, we generate a synthetic document containing that text. Synthetic data generation provides an opportunity to build an optical character recognition system without the cost of annotation. This process can be used to generate both line and document level images. The advantage of generating document level images is that we can build templates that mirror the complex layouts of unconstrained images. As an example, we can generate synthetic newspapers that include multiple columns, fonts, styles, and even embedded images.

At image generation time, we render a seed text drawn from an existing NER collection using a random selection from over 60 Chinese fonts. Style attributes such as font size, font color, and background color are then applied to each image. Text attributes such as rotation and random cropping provide artifacts that often are found in scans of complex document images. Finally, the image can be degraded using Gaussian noise and pixel dropout.

| Entity Type | Precision | Recall | $F_1$-score | Instances Found |
|---|---|---|---|---|
| Overall | 69.94% | 59.79% | 64.47 | 1,384 |
| CHEM | 50.00% | 9.09% | 15.38 | 2 |
| COMM | 36.36% | 54.55% | 43.64 | 33 |
| COMP | 0.00% | 0.00% | 0.00 | 0 |
| DATE | 83.76% | 75.38% | 79.35 | 117 |
| EVNT | 33.33% | 33.33% | 33.33 | 36 |
| FAC | 0.00% | 0.00% | 0.00 | 12 |
| GPE | 74.77% | 70.52% | 72.58 | 547 |
| LOC | 53.33% | 42.11% | 47.06 | 15 |
| MISC | 75.00% | 38.63% | 50.99 | 120 |
| MONEY | 83.72% | 78.26% | 80.90 | 43 |
| ORG | 35.71% | 24.39% | 28.99 | 56 |
| PER | 70.57% | 77.73% | 73.98 | 282 |
| POL | 63.95% | 45.83% | 53.40 | 86 |
| TIME | 100.00% | 100.00% | 100.00 | 1 |
| TITLE | 78.79% | 40.00% | 53.06 | 33 |
| VEH | 100.00% | 16.67% | 28.57 | 1 |

Table 4: Precision, Recall, and $F_1$-score averaged over all types and for each category in the test set for NER over OCR output boxes.

## 7. Example Use Case

In this section we demonstrate that the scoring approach laid out in this paper can be used to evaluate NER when the OCR system changes, the NER system changes, or both systems change. Rather than acquiring multiple OCR engines, we use our ability to degrade images with Gaussian noise to produce varied OCR output. This noise increased the character error rate from 2.3 to 12.4. This OCR output is different both from the transcripts that have the NER annotations and from the output reported on in Section 5.2. For an alternative NER system, we simply retrained our NER system on different data. This new NER system scored an $F_1$ of 69.60 on the transcriptions as opposed the one described in Section 5.2, which score 68.51. The new system had much higher recall, but lower precision. This version appears to have learned a better representation of chemicals (CHEM) but failed to find any vehicles (VEH).

The new NER system was then used to tag the degraded OCR output. Table 5 reports the results. Unsurprisingly, overall NER performance suffered because of the increased character error rate. This was particularly true for people (PER) where precision decreased dramatically. Performance has not universally worse. For instance, the new system performed better on natural locations (LOC). Given that the training objective is maximization of performance over all entity types, this difference could simply be due to the natural variance in performance across individual types seen when training a neural system several times on the same data. However, the main takeaway from this experiment is that no new annotations were needed to produce these results.

## 8. Conclusions

Research to improve NER performance over digitized text must be able to support the full context of NER over OCR; to do so, new collections must be created. This paper makes several contributions. First, it lays out a methodology for building OCR-NER collections. It suggests two ways to obtain named entity annotations, one that uses human annotators, the other that injects an existing NER collection into the digitized text pipeline. Next, the paper introduces and makes available[7] the Chinese Renmin collection, which

[7]The dataset can be obtained from `https://github.com/hltcoe/cmn-renmin-ocr-ner-dataset`.

| Entity Type | Precision | Recall | $F_1$-score | Instances Found |
|---|---|---|---|---|
| Overall | 61.67% | 55.38% | 58.36 | 906 |
| CHEM | 25.00% | 8.33% | 12.50 | 4 |
| COMM | 35.29% | 27.27% | 30.77 | 17 |
| COMP | 0.00% | 0.00% | 0.00 | 0 |
| DATE | 79.69% | 77.86% | 78.76 | 128 |
| EVNT | 38.89% | 37.84% | 38.36 | 36 |
| FAC | 0.00% | 0.00% | 0.00 | 22 |
| GPE | 70.40% | 63.64% | 66.85 | 527 |
| LOC | 62.50% | 50.00% | 55.56 | 16 |
| MISC | 70.83% | 28.45% | 40.60 | 96 |
| MONEY | 74.51% | 82.61% | 78.35 | 51 |
| ORG | 25.00% | 6.10% | 9.80 | 20 |
| PER | 48.63% | 75.58% | 59.18 | 401 |
| POL | 64.77% | 47.11% | 54.55 | 88 |
| TIME | 100.00% | 100.00% | 100.00 | 1 |
| TITLE | 61.29% | 57.58% | 59.38 | 62 |
| VEH | 0.00% | 0.00% | 0.00 | 0 |

Table 5: Precision, Recall, and $F_1$-score averaged over all types and for each category in the test set for NER over degraded OCR output boxes.

contains 16K entities over 10K sentences in 4 days of newspaper articles from the *Renmin Ribao* newspaper. Finally, it provides baseline OCR and NER performance numbers over the collection to help calibrate subsequent research using the collection. We plan to reuse this collection creation methodology to produce new collections in languages such as Arabic, English, Korean, and Russian.

## 9. Bibliographical References


Chiu, J. P. and Nichols, E. (2016). Named entity recognition with bidirectional lstm-cnns. *Trans. of the ACL*, 4:357–370.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Etter, D., Rawls, S., Carpenter, C., and Sell, G. (2019). A synthetic recipe for ocr. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 864–869.

Galibert, O., Rosset, S., Grouin, C., Zweigenbaum, P., and Quintard, L. (2012). Extended named entity annotation on OCRed documents: From corpus constitution to evaluation campaign. In *Proceedings of the Eighth conference on International Language Resources and Evaluation (LREC'12), Istanbul, Turkey, may*. Citeseer.

Graham, B. (2014). Fractional max-pooling. *arXiv preprint arXiv:1412.6071*.

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM.

Huang, Z., Xu, W., and Yu, K. (2015). Bidirectional lstm-crf models for sequence tagging. *CoRR*, abs/1508.01991.

Kettunen, K. and Ruokolainen, T. (2017). Names, right or wrong: Named entities in an OCRed historical Finnish newspaper collection. In *Proceedings of the 2Nd International Conference on Digital Access to Textual Cultural Heritage*, DATeCH2017, pages 181–186, New York, NY, USA. ACM.

Lin, Y., Costello, C., Zhang, B., Lu, D., Ji, H., Mayfield, J., and McNamee, P. (2018). Platforms for non-speakers annotating names in any language. In *Proceedings of ACL 2018, System Demonstrations*, pages 1–6. Association for Computational Linguistics.

Miller, D., Boisen, S., Schwartz, R., Stone, R., and Weischedel, R. (2000). Named entity extraction from noisy input: Speech and OCR. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLC '00, pages 316–324, Stroudsburg, PA, USA. Association for Computational Linguistics.

Neudecker, C. (2016). An open corpus for named entity recognition in historic newspapers. In *LREC*.

Packer, T. L., Lutes, J. F., Stewart, A. P., Embley, D. W., Ringger, E. K., Seppi, K. D., and Jensen, L. S. (2010). Extracting person names from diverse and noisy OCR text. In *Proceedings of the fourth workshop on Analytics for noisy unstructured text data*, pages 19–26. ACM.

Rawls, S., Cao, H., Kumar, S., and Natarjan, P. (2017). Combining convolutional neural networks and lstms for segmentation free OCR. In *Proc. ICDAR*.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556.

Tjong Kim Sang, E. F. and De Meulder, F. (2003). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142–147, Stroudsburg, PA, USA. Association for Computational Linguistics.