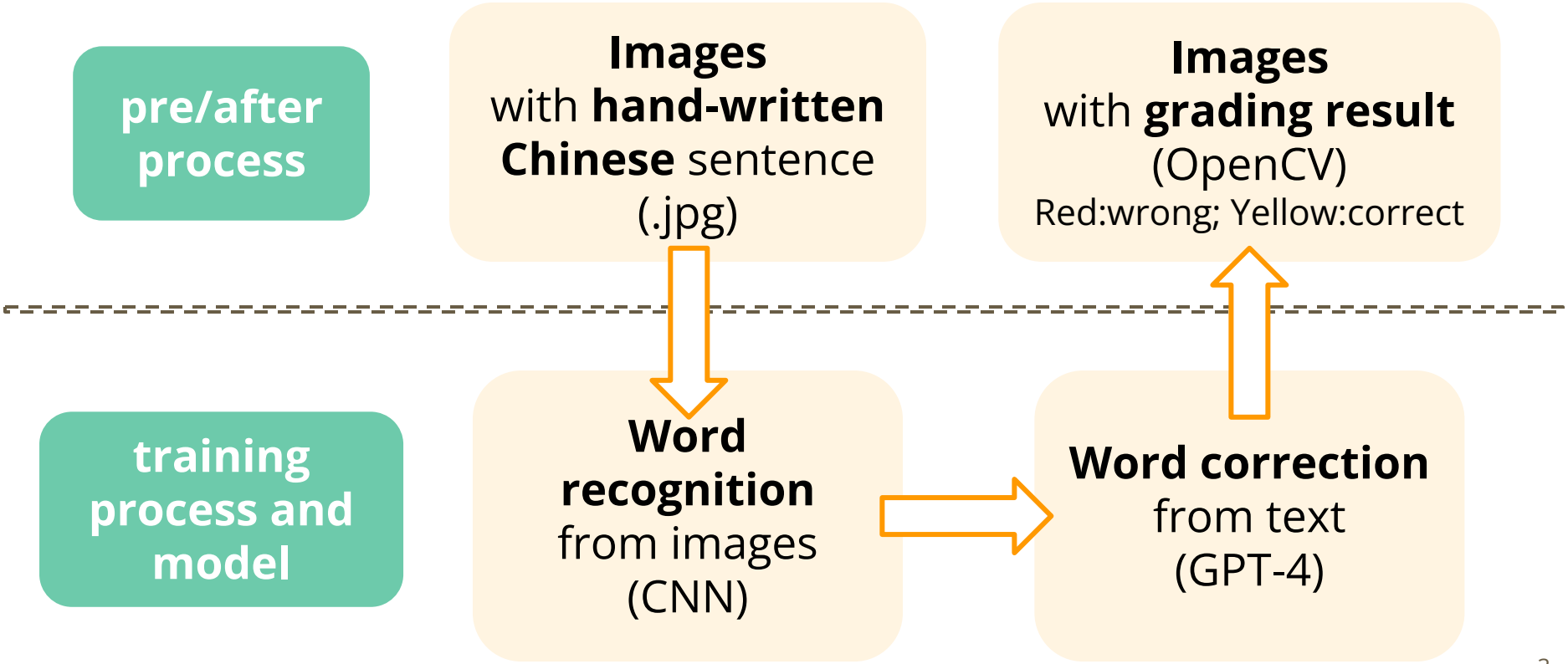
Hand-written Article Correction

Group name: Group one
Members:
111550022 施采緹 111550090 劉思予
111550131 張芷瑜 110550047 巫廷翰

Introduction

"How can we accurately digitalize and correct hand-written Chinese articles to ensure they are coherent and error-free?"

Pipeline


pre/after process
Images with hand-written Chinese sentence (.jpg)
Images with grading result (OpenCV)
Red:wrong; Yellow:correct
training process and model
Word recognition from images (CNN)
Word correction from text (GPT-4)

Related Work

Open Source

- [ocrcn-tf2] https://github.com/jjcheer/ocrcn_tf2
- [Tesseract OCR] https://github.com/tesseract-ocr/tesseract
- [CnOCR] https://github.com/breezedeus/CnOCR

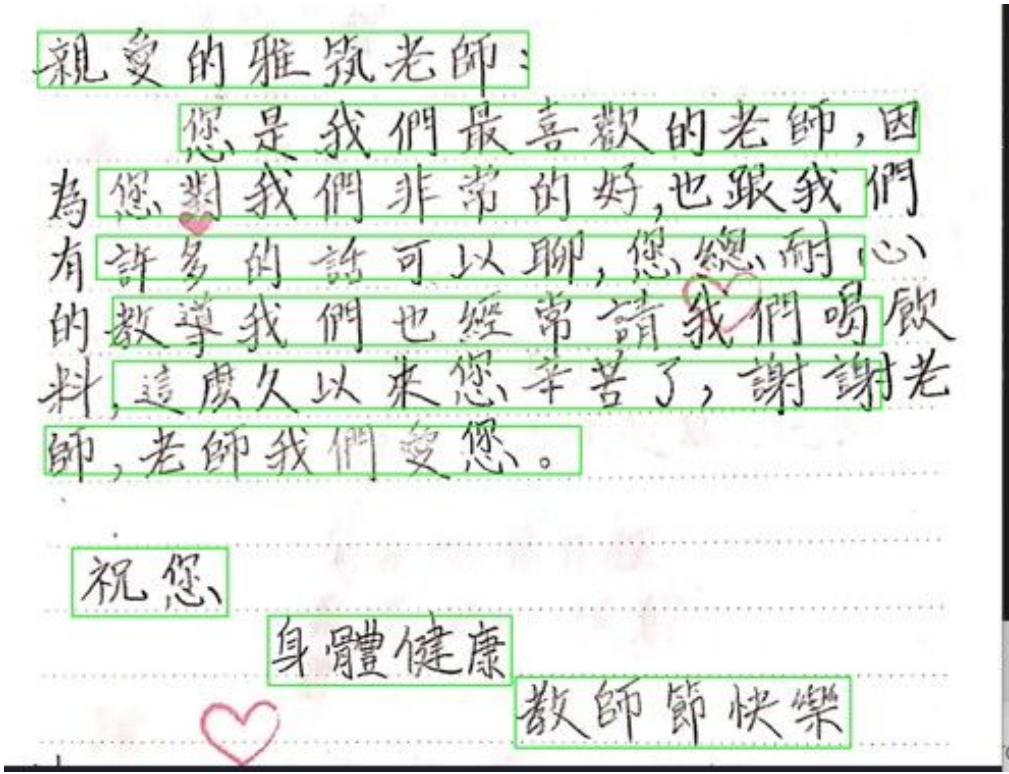
Others

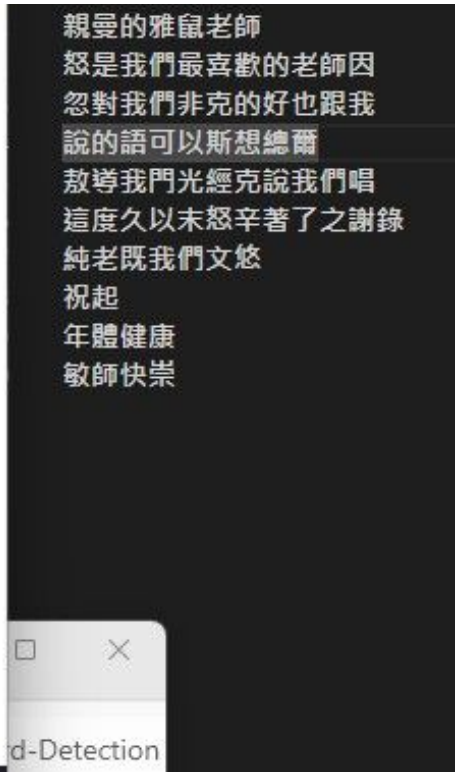
- [Google Cloud Vision API] Detect text in images | Cloud Vision API
- [Adobe Acrobat] 掃描與OCR | Adobe Acrobat

Compare with other methods (open source)

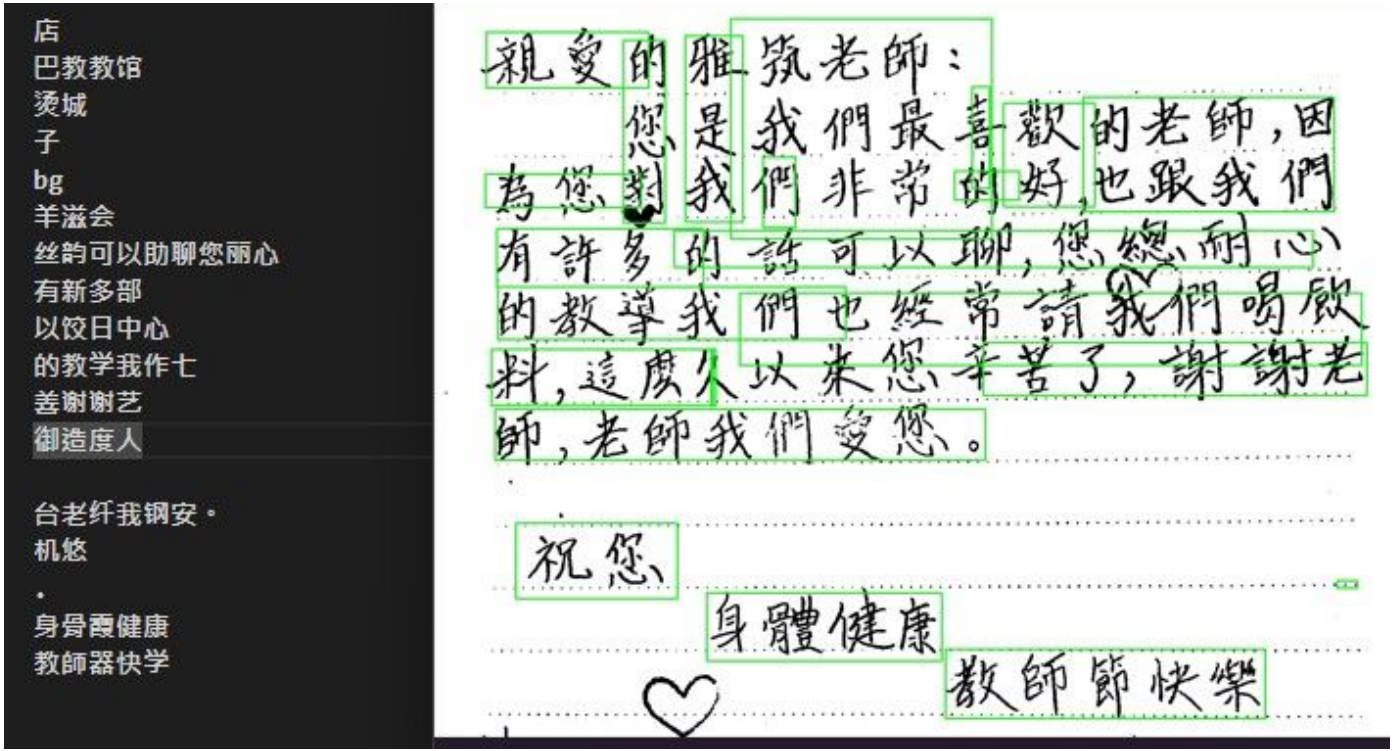
ocrcn_tf2	CnOcr (Traditional)	Tesseract	CnOcr (Simplified)
Handwriting Dataset	pretrained	chi_tra	pretrained
OCR	Conv + RNN + CTC	OCR	Conv + RNN + CTC

Other methods - CnOcr (Traditional)

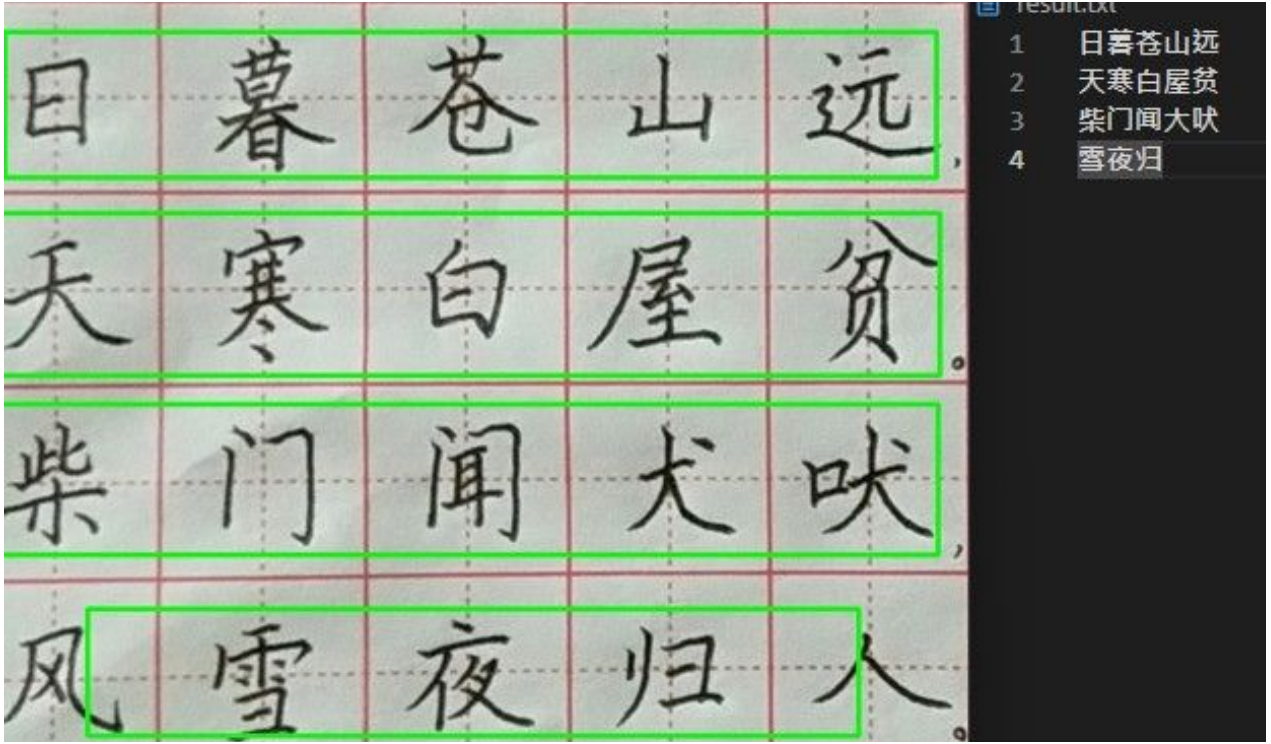




Other methods - Tessertact



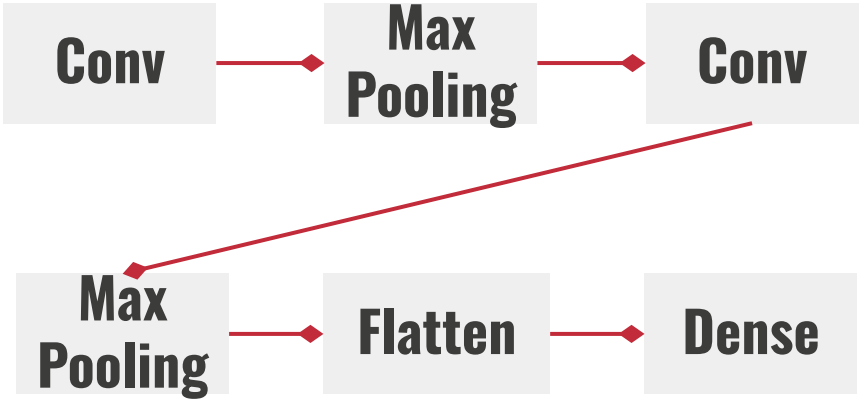
Other methods - CnOcr (Simplified)



Dataset

- Use Traditional Chinese Handwriting Dataset (common words dataset) to train the model
- 4,803 characters (classes)
- 250, 712 images

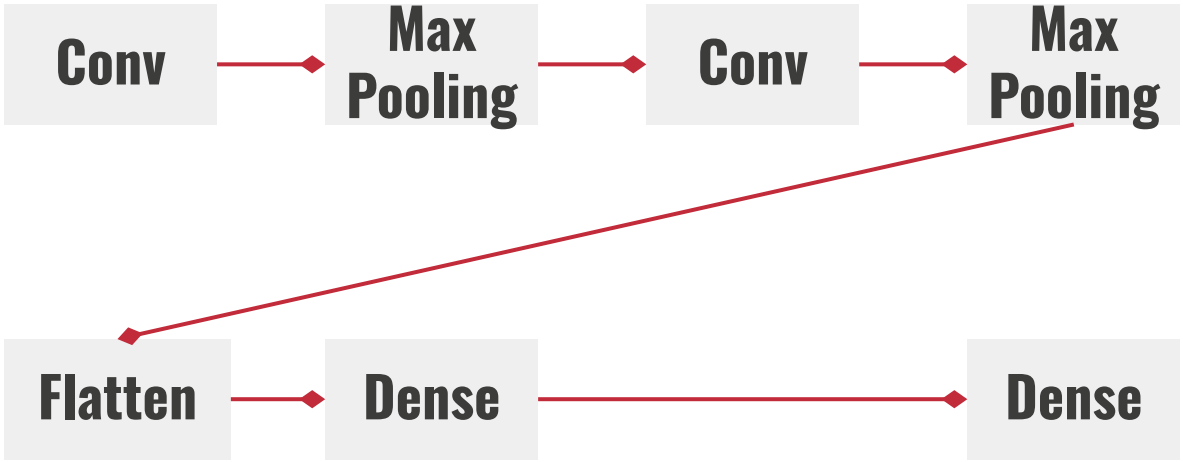
Baseline - a simple CNN Model




```python
def build_net_3(input_shape, num_classes):
    model = Sequential()

    model.add(Conv2D(input_shape = input_shape, filters = 32,
                     kernel_size = (3, 3), strides = (1, 1),
                     padding = 'same', activation = 'relu'))
    model.add(MaxPooling2D(pool_size = (2, 2)), padding = 'same')

    model.add(Conv2D(filters = 64, kernel_size = (3, 3), padding = 'same'))
    model.add(MaxPooling2D(pool_size = (2, 2), padding = 'same'))

    model.add(Flatten())

    model.add(Dense(num_classes, activation = 'softmax'))

    return model
```

- 2 Convolution layers
- 2 MaxPooling layers
- 1 Flatten Layer
- 1 Dense Layer

Main Approach - CNN Model



- Add a dense layer to extract more characters
- Use relu function to increase unlinearity of model

Main Approach - Training

STEP 1

Preprocess data

Use ImageDataGenerator to load data and enhance images

- 80% Training
- 20% Validation

STEP 2

Initialize model

Import CNN model and set the parameters

STEP 3

Compile model

- Optimizer: Adam
- Loss: SparseCategoricalCrossentropy
- Metrics: accuracy

STEP 4

Training

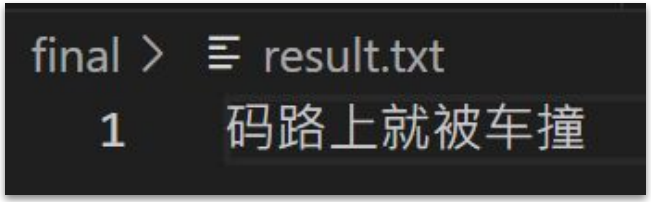
- batch = 32
- epoch = 1000
- steps_per_epoch = 1024

Main Approach - Detection & Correction of Wrong Characters

STEP 1

Transform handwritten article to .txt file

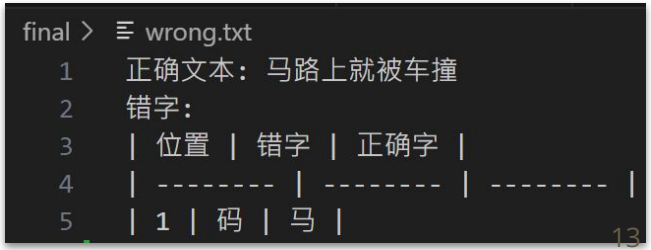
Input handwritten article image to the model, and get the output .txt file



STEP 2

Detecting & Correcting wrong characters

Detect and correct wrong characters in the text file by integrating ChatGPT



Main Approach - Marking Errors in the Article

STEP 1

Given the .jpg file

Modify the original image:

1. **resize** the image
2. convert image intto **grayscale**

STEP 2

Horizontal / Vertical projection

Count the number of white pixels in:

1. every rows -> horizontal projection (line seperation)
2. every columns -> vertical projection (word separation)

STEP 3

Mark each words

Mark every single word:

1. **True** -> yellow
2. **False** -> red

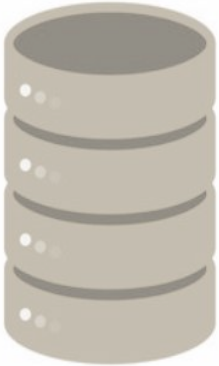
Limitations

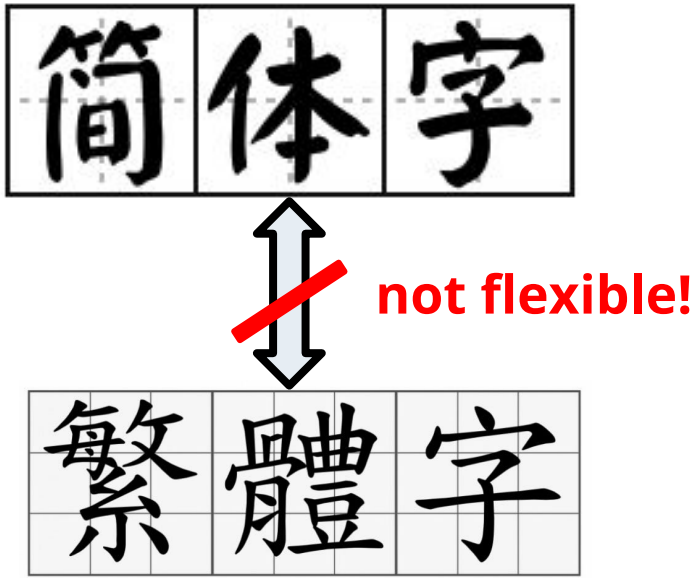
OpenCV

vertical projection

Limitations

Traditional / Simplified Chinese





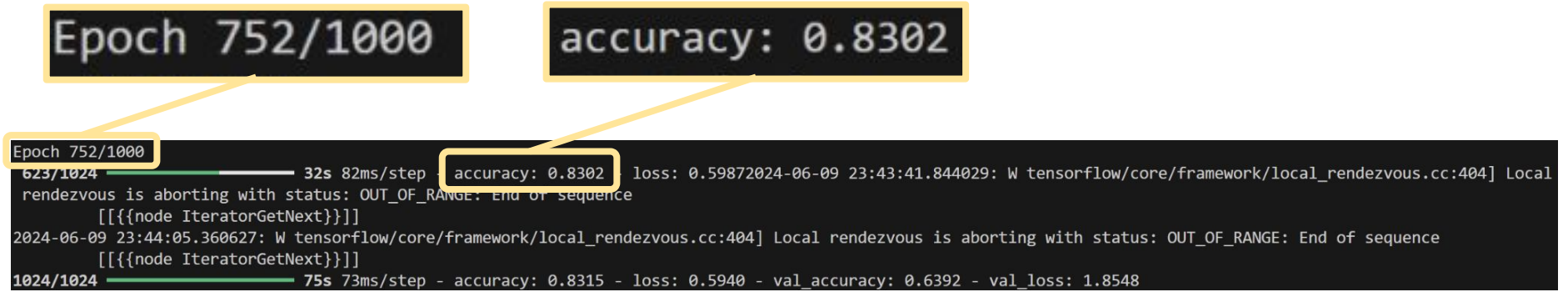
Detect and Correct Error

input: 在路上背车撞
expect:在路上被车撞

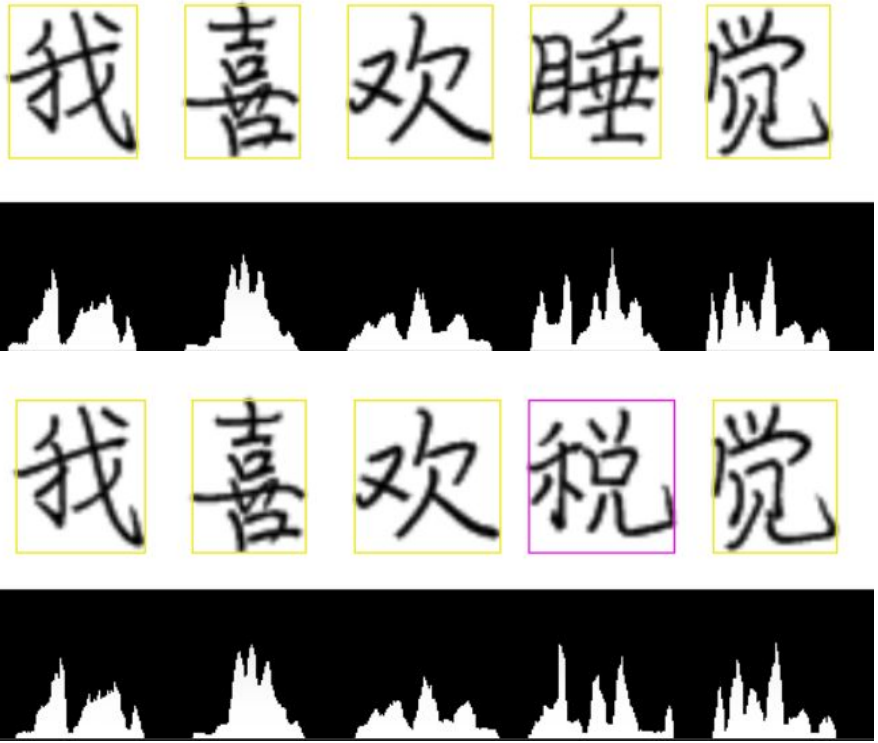
detect: 在路上背华拉
correct: 在路上背画拉

If the detected result is poor, we can't provide a good correction!

Result & Analysis



Result & Analysis - Success




```
錯誤文本：我喜欢睡觉
你提供的文本"我喜欢睡觉"没有错字，
所以无法按照你的要求提供错误位置和正确字。
All correct!
```

```
錯誤文本：我喜欢税觉
正确文本：我喜欢睡觉
错字：
| 位置 | 错字 | 正确字 |
| -------- | -------- | -------- |
| 4 | 税 | 睡 |
```

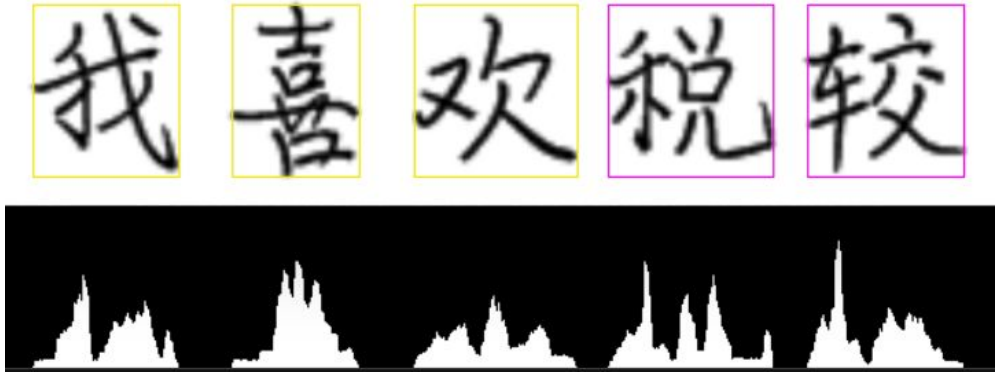
Result & Analysis - Fail

input: 我喜歡稅較

expect: 我喜歡睡覺

detect: 找喜歡稅較

correct: 找喜歡的書籍



錯誤文本：找喜欢税较
正确文本：找喜欢的书籍
错字：

位置	错字	正确字
4	税	的
5	较	书
6	空	籍

Reference

- ocrcn_tf2: https://github.com/jjcheer/ocrcn_tf2
- CnOCR: https://github.com/breezedeus/CnOCR
- Tesseract: https://github.com/tesseract-ocr/tesseract
- Traditional Chinese Handwriting text dataset: https://github.com/chenkenanalytic/handwritting_data_all
- Google Cloud Vision API: https://cloud.google.com/vision/docs/ocr
- Adobe Acrobat: https://experienceleague.adobe.com/zh-hant/docs/document-cloud-learn/acrobat-learning/getting-started/scan-and-ocr
- Word spliting method: https://www.cnblogs.com/zxy-joy/p/10687152.html

THANKS FOR LISTENING