

# demo

---

## Tesseract

[安装](#)

[使用](#)

[脚本](#)

[命令行](#)

[特点](#)

[示例](#)

## PaddleOCR

[安装](#)

[使用](#)

[脚本](#)

[命令行](#)

[特点](#)

[示例](#)

## EasyOCR

[安装](#)

[使用](#)

[脚本](#)

[命令行](#)

[特点](#)

[示例](#)

## Kraken OCR

[安装](#)

[使用](#)

[脚本](#)

[命令行](#)

[特点](#)

# Tesseract

## 安装

最常见方式是通过 pip:

```
1 pip install pytesseract pillow
```

也可以直接安装 Tesseract 引擎:

- **Windows:** 下载 Tesseract 官方安装包
- **Linux (Ubuntu/Debian):**

```
1 sudo apt install tesseract-ocr libtesseract-dev
```

## 使用

### 脚本

```
1 import pytesseract
2 from PIL import Image
3
4 img = Image.open("test.png")
5 text = pytesseract.image_to_string(img, lang="chi_sim")
6 print(text.strip())
```

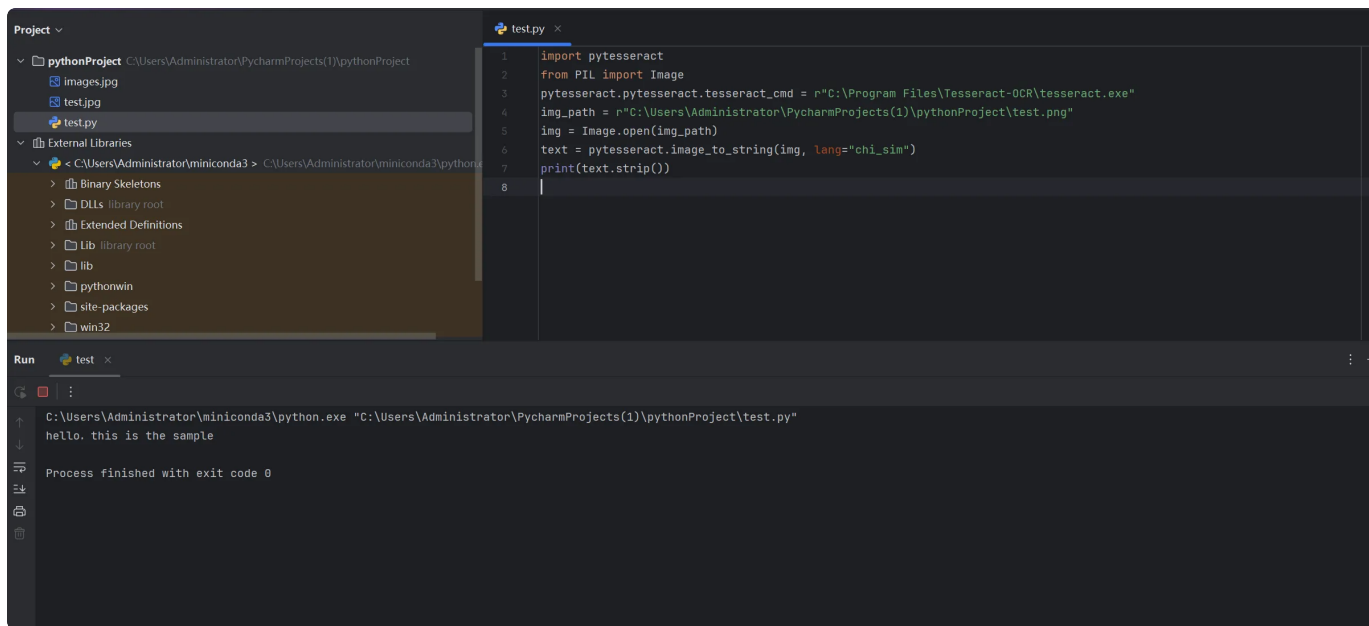
### 命令行

```
1 tesseract test.png out.txt -l chi_sim
```

## 特点

- 免费开源，Google 维护，社区大。
- 支持 100+ 语言（含中文），但对复杂背景、扭曲图片稍弱。
- 提供 `tesseract` CLI 和 Python 封装库 `pytesseract`。
- 对 CPU 友好，不依赖 GPU。

## 示例



The screenshot shows the PyCharm IDE interface. On the left, the 'Project' view displays a file structure for 'pythonProject' with files like 'images.jpg', 'test.jpg', and 'test.py'. The 'test.py' file is selected. The main editor window shows the following Python code:

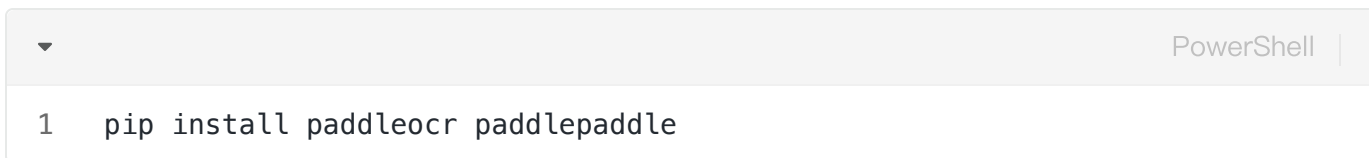
```
1 import pytesseract
2 from PIL import Image
3 pytesseract.pytesseract.tesseract_cmd = r"C:\Program Files\Tesseract-OCR\tesseract.exe"
4 img_path = r"C:\Users\Administrator\PycharmProjects(1)\pythonProject\test.png"
5 img = Image.open(img_path)
6 text = pytesseract.image_to_string(img, lang="chi_sim")
7 print(text.strip())
8
```

Below the editor, the 'Run' window shows the execution output:

```
C:\Users\Administrator\miniconda3\python.exe "C:\Users\Administrator\PycharmProjects(1)\pythonProject\test.py"
hello, this is the sample
Process finished with exit code 0
```

## PaddleOCR

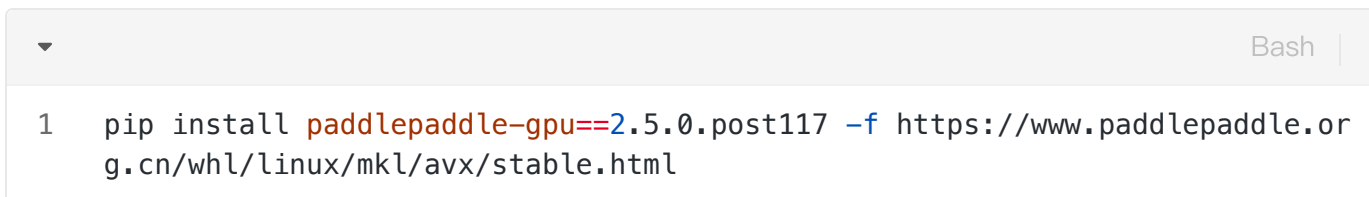
### 安装



The terminal window shows a PowerShell prompt with the following command:

```
1 pip install paddleocr paddlepaddle
```

GPU 用户建议提前安装对应版本的 PaddlePaddle:



The terminal window shows a Bash prompt with the following command:

```
1 pip install paddlepaddle-gpu==2.5.0.post117 -f https://www.paddlepaddle.org.cn/whl/linux/mkl/avx/stable.html
```

## 使用

### 脚本

```
Python |
1  from paddleocr import PaddleOCR
2
3  ocr = PaddleOCR(use_angle_cls=True, lang="ch")
4  img_path = r"C:\\Users\\yuzhe\\Desktop\\test\\test.png"
5  result = ocr.ocr(img_path, cls=True)
6
7  print("识别结果: \n")
8  for line in result[0]:
9      print(line[1][0])
```

### 命令行

```
PowerShell |
1  paddleocr --image_dir test.png --lang=ch
```

## 特点

- 中文识别准确率高，支持竖排、手写体。
- 提供快速模型与高精度模型，灵活选择。
- Python API 完善，社区活跃，文档齐全。
- 可调用 **表格识别**、**版面分析** 模块。

## 示例

```
(paddleocr.env) PS C:\Users\yuzhe\Desktop\test> python test.py
download https://paddleocr.bj.bcebos.com/PP-OCRv3/english/en_PP-OCRv3_det_infer.tar to C:\Users\yuzhe/.paddleocr\whl\det\en\en_PP-OCRv3_det_infer\en_PP-OCRv3_det_infer.tar
100%|#####| 4.00M/4.00M [00:03<00:00, 1.20MiB/s]
download https://paddleocr.bj.bcebos.com/PP-OCRv4/english/en_PP-OCRv4_rec_infer.tar to C:\Users\yuzhe/.paddleocr\whl\rec\en\en_PP-OCRv4_rec_infer\en_PP-OCRv4_rec_infer.tar
100%|#####| 10.2M/10.2M [00:08<00:00, 1.14MiB/s]
download https://paddleocr.bj.bcebos.com/dygraph_v2.0/ch/ch_ppocr_mobile_v2.0_cls_infer.tar to C:\Users\yuzhe/.paddleocr\whl\cls\ch_ppocr_mobile_v2.0_cls_infer\ch_ppocr_mobile_v2.0_cls_infer.tar
100%|#####| 2.19M/2.19M [00:01<00:00, 1.20MiB/s]
[2025/08/20 22:39:11] ppocr DEBUG: Namespace(help='==SUPPRESS==', use_gpu=False, use_xpu=False, use_npu=False, ir_optim=True, use_tensorrt=False, min_subgraph_size=15, precision='fp32', gpu_mem=500, gpu_id=0, image_dir=None, page_num=0, det_algorithm='DB', det_model_dir='C:\\Users\\yuzhe\\.paddleocr\\whl\\det\\en\\en_PP-OCRv3_det_infer', det_limit_side_len=960, det_limit_type='max', det_box_type='quad', det_db_thresh=0.3, det_db_box_thresh=0.6, det_db_unclip_ratio=1.5, max_batch_size=10, use_dilation=False, det_db_score_mode='fast', det_east_score_thresh=0.8, det_east_cover_thresh=0.1, det_east_nms_thresh=0.2, det_sast_score_thresh=0.5, det_sast_nms_thresh=0.2, det_pse_thresh=0, det_pse_box_thresh=0.85, det_pse_min_area=16, det_pse_scale=1, scales=[8, 16, 32], alpha=1.0, beta=1.0, fourier_degree=5, rec_algorithm='SVTR_LCNet', rec_model_dir='C:\\Users\\yuzhe\\.paddleocr\\whl\\rec\\en\\en_PP-OCRv4_rec_infer', rec_image_inverse=True, rec_image_shape=[3, 48, 320], rec_batch_num=6, max_text_length=25, rec_char_dict_path='C:\\Users\\yuzhe\\.anaconda3\\envs\\paddleocr\\env\\lib\\site-packages\\paddleocr\\ppocr\\utils\\en_dict.txt', use_space_char=True, vis_font_path='./doc/fonts/simfang.ttf', drop_score=0.5, e2e_algorithm='PGNet', e2e_model_dir=None, e2e_limit_side_len=768, e2e_limit_type='max', e2e_pgnet_score_thresh=0.5, e2e_char_dict_path='./ppocr/utils/ic15_dict.txt', e2e_pgnet_valid_set='totaltext', e2e_pgnet_mode='fast', use_angle_cls=True, cls_model_dir='C:\\Users\\yuzhe\\.paddleocr\\whl\\cls\\ch_ppocr_mobile_v2.0_cls_infer', cls_image_shape=[3, 48, 192], label_list=['0', '180'], cls_batch_num=6, cls_thresh=0.9, enable_mkldnn=False, cpu_threads=10, use_pdserving=False, warmup=False, sr_model_dir=None, sr_image_shape=[3, 32, 128], sr_batch_num=1, draw_img_save_dir='./inference_results', save_crop_res=False, crop_res_save_dir='./output', use_mp=False, total_process_num=1, process_id=0, benchmark=False, save_log_path='./log_output/', show_log=True, use_onnx=False, output='./output', table_max_len=488, table_algorithm='TableAttn', table_model_dir=None, merge_no_span_structure=True, table_char_dict_path=None, layout_model_dir=None, layout_dict_path=None, layout_score_threshold=0.5, layout_nms_threshold=0.5, kie_algorithm='LayoutXLM', ser_model_dir=None, re_model_dir=None, use_visual_backbone=True, ser_dict_path='./train_data/XFUND/class_list_xfun.txt', ocr_order_method=None, mode='structure', image_orientation=False, layout=True, table=True, ocr=True, recovery=False, use_pdserving=False, lang='en', det=True, rec=True, type='ocr', ocr_version='PP-OCRv4', structure_version='PP-StructureV2')
[2025/08/20 22:39:18] ppocr DEBUG: dt_boxes num : 1, elapse : 1.3946208953857422
[2025/08/20 22:39:18] ppocr DEBUG: cls num : 1, elapse : 0.17692351341247559
[2025/08/20 22:39:19] ppocr DEBUG: rec_res num : 1, elapse : 0.7296772803173828
识别结果:
hello, this is the sample
```

# EasyOCR

## 安装

PowerShell

```
1 pip install easyocr
```

需要确保 PyTorch 已安装，否则需先安装：

Bash

```
1 pip install torch torchvision torchaudio
```

## 使用

## 脚本

Python

```
1 import easyocr
2
3 reader = easyocr.Reader(['ch_sim', 'en'])
4 results = reader.readtext("test.png")
5
6 for _, text, _ in results:
7     print(text)
```

## 命令行

虽然 EasyOCR 主要提供 Python API，但也可以封装成简单脚本实现 CLI 功能。

## 特点

- 使用 PyTorch，支持 80+ 语言。
- 易上手，几行代码即可完成 OCR。
- 识别率不如 PaddleOCR，但比 Tesseract 对复杂背景友好。
- GPU 支持好，速度快。

## 示例

```
(paddleocr_env) PS C:\Users\yuzhe\Desktop\test> python test.py
Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GPU.
C:\Users\yuzhe\anaconda3\envs\paddleocr_env\lib\site-packages\torch\utils\data\dataloader.py:666: UserWarning: 'pin_memory' argument is set as true but no accelerator is found, then device pinned memory won't be used.
  warnings.warn(warn_msg)
hello ,
this is the sample
```

## Kraken OCR

### 安装

▼

PowerShell |

```
1 pip install kraken
```

源码安装：

▼

Bash |

```
1 git clone https://github.com/mittagessen/kraken.git
2 cd kraken
3 pip install -e .
```

### 使用

## 脚本

Python |

```
1  from kraken import rpred
2  from PIL import Image
3
4  im = Image.open("test.png")
5  model = rpred.load_any("en-default.mlmodel")
6  predictions = rpred.rpred(model, im)
7
8  print("识别结果: ")
9  for line in predictions:
10     print(line.text)
```

## 命令行

PowerShell |

```
1  kraken -i test.png out.txt binarize segment ocr --model en-default.mlmodel
```

## 特点

- 专注历史文献、古籍 OCR。
- 内置版面分析 (layout analysis) , 支持多栏、复杂排版。
- 支持用户自定义训练模型, 适合特殊字体和语言。
- 社区较小, 学习曲线高于 Tesseract/EasyOCR。