



PaddleOCR的java高性能部署

分享嘉宾：Calvin

DJL Contributor

AIAS Founder



目录

CONTENT

项目背景

实现机制与核心代码

性能调优与提高识别率的办法

AIAS开源项目介绍

- **业务痛点**
- 人工录入时间长，出错率高，没有自动化机制
- 随着业务增长，文档数量爆发式增长，需要不断增加人力满足数据结构化的业务需求
- 客户业务办理操作冗烦，客户体验差
- 员工重复性操作，效率低下



OCR能力分类

类型	模型应用
证件类	身份证
	营业执照
	银行卡
	驾驶证
	户口本
	房产证
	...
票据类	发票
	火车票
	机票
	...
表格类	企业报表
	...
其它	通用文字识别
	...

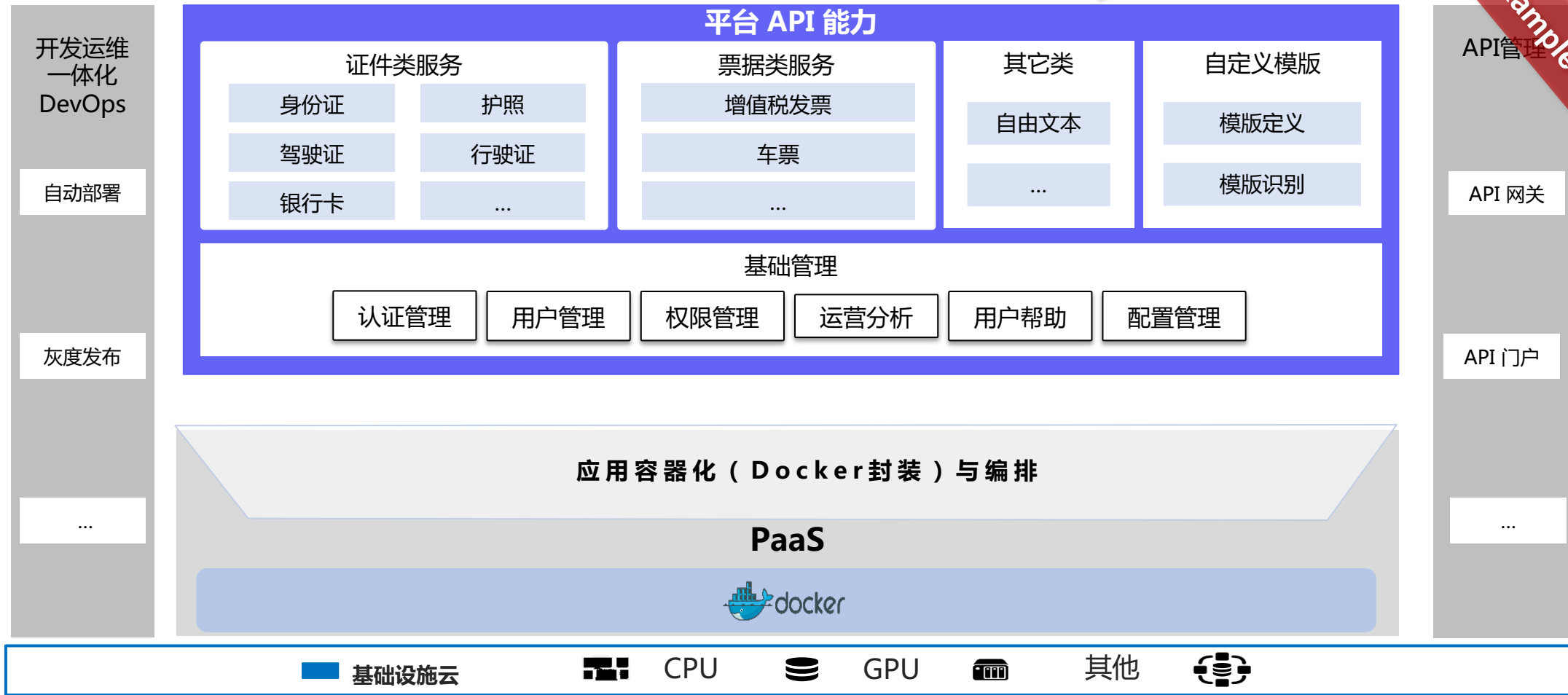
ToB - OCR能力平台

飞桨

企业内部各个业务领域对人工智能的需求比较旺盛，为更好赋能业务，参照成熟架构模型，借鉴人工智能应用理念，构建快速化、智能化AI架构。

企业内部IT部门，如何满足业务方的AI需求？

Example



AI技术选型：PaddleOCR

PP-OCR系列高质量预训练模型，准确的识别效果

- 超轻量PP-OCRv2系列：检测（3.1M）+ 方向分类器（1.4M）+ 识别（8.5M）= 13.0M
- 超轻量PP-OCR mobile移动端系列：检测（3.0M）+ 方向分类器（1.4M）+ 识别（5.0M）= 9.4M
- 通用PPOCR server系列：检测（47.1M）+ 方向分类器（1.4M）+ 识别（94.9M）= 143.4M

PP-Structure文档结构化系统

- 支持版面分析与表格识别（含Excel导出）
- 支持关键信息提取任务
- 支持DocVQA任务

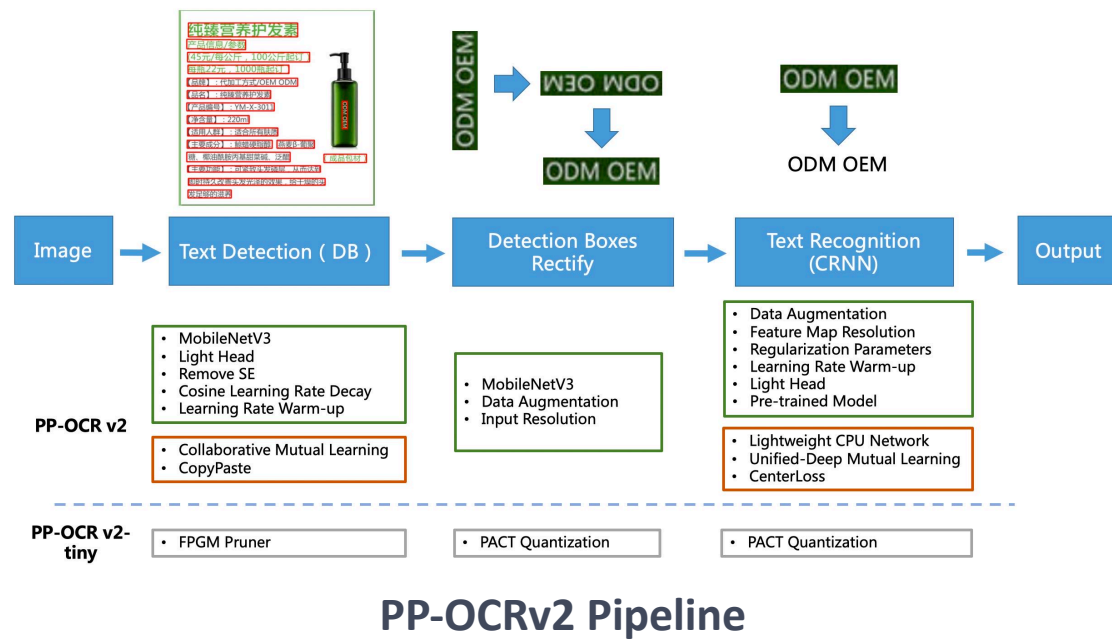
丰富易用的OCR相关工具组件

- 半自动数据标注工具PPOCRLabel：支持快速高效的数据标注
- 数据合成工具Style-Text：批量合成大量与目标场景类似的图像

支持用户自定义训练，提供丰富的预测推理部署方案

支持PIP快速安装使用

可运行于Linux、Windows、MacOS等多种系统



目录

CONTENT

项目背景

实现机制与核心代码

性能调优与提高识别率的办法

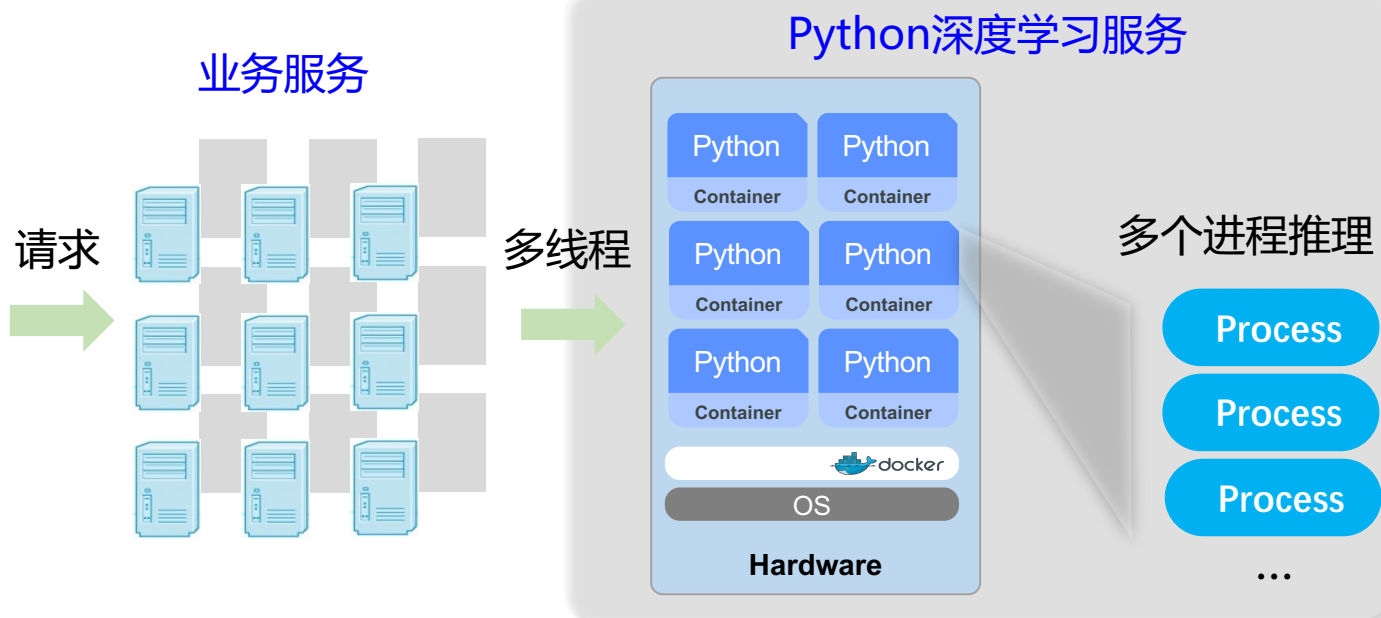
AIAS开源项目介绍

客户痛点案例 – 中国某知名电商，Netflix

原做法：搭建Python server然后来进行深度学习推理

缺点：

- 多了一层网络传输的开销
- Python使用多进程：内存/显存开销大
- 多维护一个推理服务
- 跨语言维护(Python, Java)



新方案：部署模型到Java服务上

优点：

- 去除Python服务：减少延迟
- 原生支持多线程：内存开销减少
- DJL内建内存管理系统(NDManager)：长时间推理也不会内存泄漏



多线程使用时不会额外复制模型，相当于是就算开100个线程，内存里也只有一个模型的占用，GPU也是同理。
一个多线程推理吊打python的多进程推理的核心优势。

DJL简介 DeepJavaLibrary

飞桨

DJL官网:

<https://djl.ai>

<https://github.com/deepjavalibrary/djl>



客户成功案例：

- Amazon广告开发团队，零售团队等。
- TalkingData
- Netflix
- 中国知名电商

特性：

- 支持广泛，构建在多个深度学习框架之上 (PaddlePaddle, TensorFlow, PyTorch, MXNet, etc)
- 性能优异，多线程支持和内存管理，广泛用于搜广推任务
- 配置简单，可以轻松使用DJL部署你的模型
- 运行稳定，长时间运行性能及内存/显存开销非常稳定
- 社区活跃，AWS积极为开源社区贡献



aws

Lanking - AWS AI算法工程师

<https://www.zhihu.com/people/lanking-42>

往期培训：

Java全方位、高适配推理
部署方案强势来袭！



DJL 支持的深度学习框架及系统

No.	系统	MXNet	PyTorch	Tensorflow	TFLite	ONNXRuntime	PaddlePaddle
1	Windows	✓	✓	✓	×	✓	✓
2	Win GPU	✓	✓	×	×	✓	×
3	MAC x86	✓	✓	✓	✓	✓	✓
4	Cent OS 7	×	✓	✓	×	✓	×
5	Ubuntu	✓	✓	✓	✓	✓	✓
6	Linux GPU	✓	✓	✓	×	✓	✓
7	Android	×	✓	×	×	×	×

DJL 架构

支持的模型:

PaddlePaddle

- Paddle 动态图模型 (2.0)
- Paddle 静态图模型 (1.x)

Apache MXNet

- Symbolic模型
- Gluon模型需要先hybridize后保存

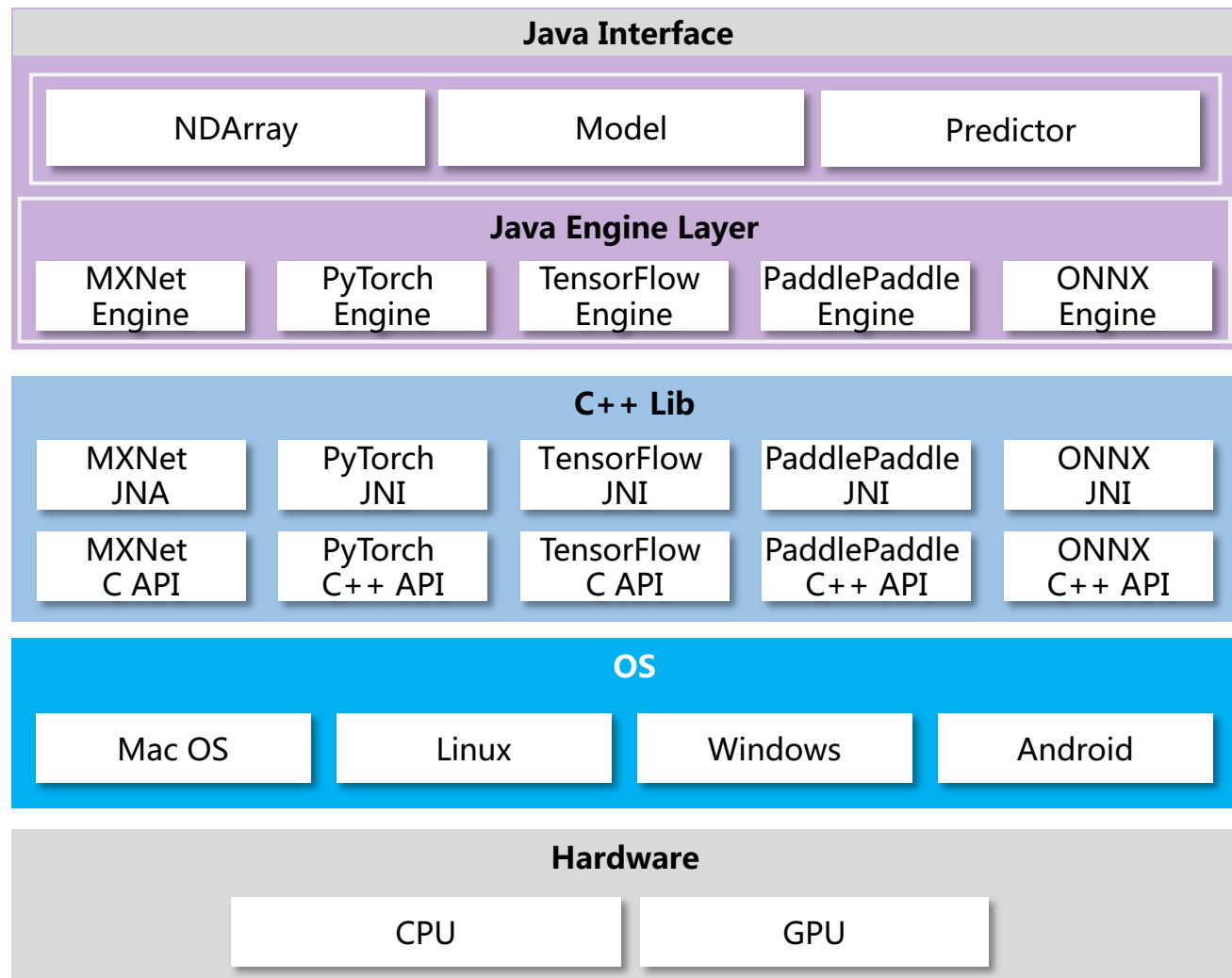
PyTorch

- TorchScript模型
- PyTorch模型可通过jt.trace或是jit.script转成TorchScript

TensorFlow

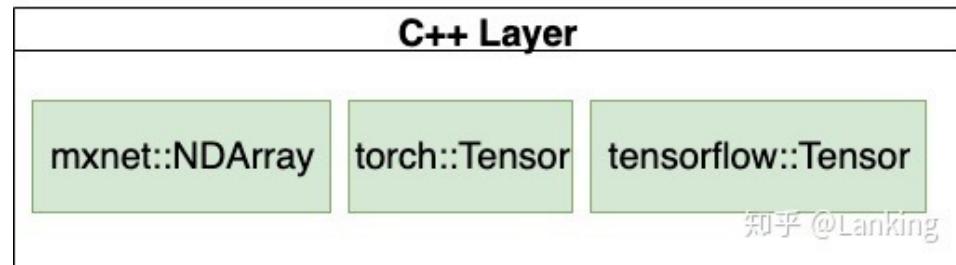
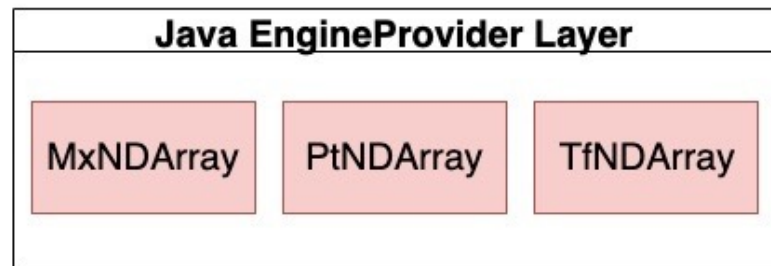
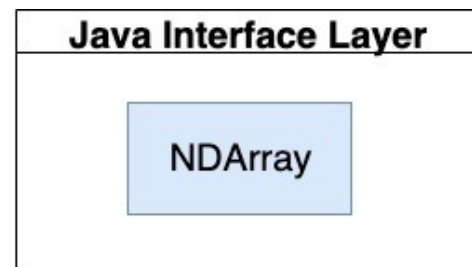
- SavedModel模型
- h5模型可通过tf.keras.models.load_model搭配tf.saved_model.save转成SavedModel

ONNX



DJL 架构 – 核心组件

- NDArrary
相当于python numpy的java实现，解决复杂的矩阵运算问题
多维数组，存在于native C++ 内存里，如此可以方便调用下面的加速库：
 - 矩阵加速库：LAPACK, BLAS
 - CPU加速库：oneDNN(MKLDNN)
 - GPU加速库：CUDA, cuDNN
- NDManager
- 管理NDArrary, 更好的内存自动回收管理
- Model
 - Trainer
 - Predictor



知乎 @Lanking

DJL 架构 – 核心组件 - NDArray

- 四则运算：add, sub, mul, div, ...
- 矩阵运算：matMul
- 比较运算：eq, gt, ...
- 归约运算：sum, max, min, ...
- 其它运算：abs, exp, ...
- 改变形状：reshape, swapAxes, ...

1	2
3	4

数组array
(2,2)

1+2	2+2
3+2	4+2

array.add(2)

1 * 2	2 * 2
3 * 2	4 * 2

array.mul(2)

1	2
3	4

1	2
3	4

array.matMul(array)

1	2
3	4

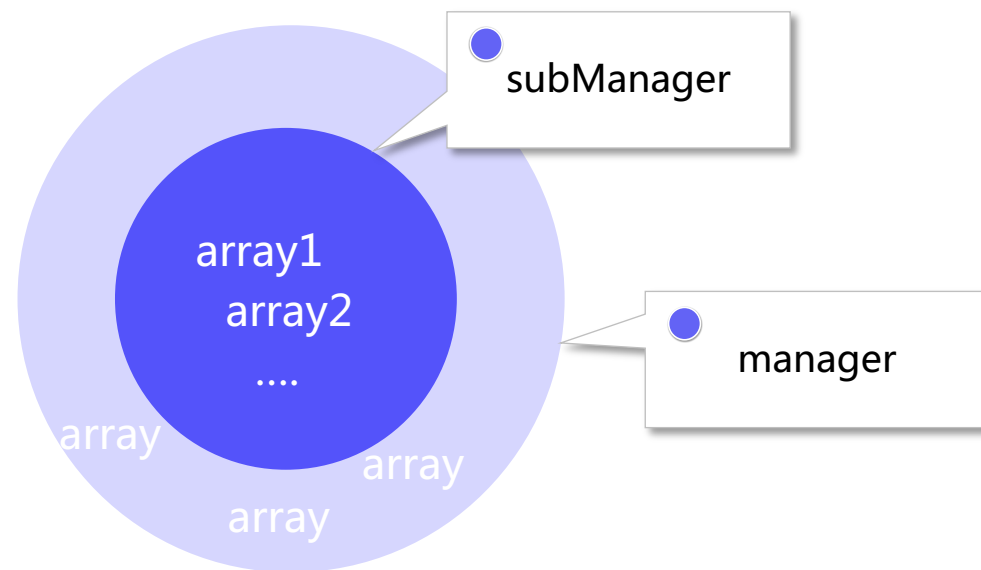
Array.max()

DJL 架构 – 核心组件 - NDManager

- 管理数据的创建及生命周期
- 创建所有NDArray都需要通过NDManager

```
manager.create(new float[] {0f, 1f, 2f, 3f}, new Shape(2, 2));
manager.zeros(new Shape(2, 2));
manager.randomUniform(0f, 1f, new Shape(2, 2));
```

```
NDManager subManager = manager.newSubManager();
array.attach(subManager);
// 调用close方法，及时回收subManager管理的内存
subManager.close();
```



推理流程



推理流程实现 - Translator



DJL 提供了高级接口 Translator

```
public NDList processInput(TranslatorContext ctx, I)
```

```
public O processOutput(TranslatorContext ctx, NDList list)
```

优点：

统一代码处理逻辑

增加代码的复用性

推理流程实现 – Criteria / Model / Predictor

模型加载方式:

- 支持本地模型加载：包括zip文件，文件夹
 - optModelPath (“path/to/file”)
- 支持从URL加载：包括 http, s3, hdfs
 - optModelUrls (“url”)

多线程关键点：

- CPU环境，model共享一个
- GPU环境，一个GPU创建加载一个
- 多线程环境下，一个线程创建一个predictor
- 避免每次predict时都创建新的predictor

// 加载模型

```
Criteria<Image, Classifications> criteria = Criteria.builder()
    .optEngine( “PaddlePaddle” )
    .setTypes(Image.class, Classifications.class)
    .optModelUrls( “/Users/calvin/Documents/ch_PP-OCRv2_det_infer” )
    .optTranslator(new PpWordDetectionTranslator ())
    .optProgress(new ProgressBar())
    .build();
```

```
try ( ZooModel<Image, Classifications> model = criteria.loadModel()){
    Classifications predictResult = null;
```

// 运行推理

```
try ( Predictor< Image, Classifications> predictor = model.newPredictor()){
    predictResult = predictor.predict(list);
    ...
}
```

推理流程实现 – 例子 – 方向检测

文字方向检测：

// 加载模型

```
Criteria<Image, Classifications> criteria = Criteria.builder()
    .optEngine( "PaddlePaddle" )
    .setTypes(Image.class, Classifications.class)
    .optModelPath( "/Users/calvin/Documents/ch_ppocr_mobile_v2.0_cls_infer.zip" )
    .optTranslator(new PpWordRotateTranslator ())
    .optProgress(new ProgressBar())
    .build();
```

```
ZooModel< Image, Classifications> model = criteria.loadModel();
```

// 运行推理

```
Predictor< Image, Classifications> predictor = model.newPredictor();
predictor.predict(list);
```

```
public class PpWordRotateTranslator implements Translator<Image,
Classifications> {
    List<String> classes = Arrays.asList("No Rotate", "Rotate");
```

```
public PpWordRotateTranslator() {}
```

```
public Classifications processOutput(TranslatorContext ctx, NDList list) {
    NDArray prob = list.singletonOrThrow();
    return new Classifications(this.classes, prob);
}
```

```
public NDList processInput(TranslatorContext ctx, Image input){
    NDArray img = input.toNDArray(ctx.getNDManager());
    img = NDImageUtils.resize(img, 192, 48);
    img = NDImageUtils.toTensor(img).sub(0.5F).div(0.5F);
    img = img.expandDims(0);
    return new NDList(new NDArray[]{img});
}
```

```
public Batchifier getBatchifier() {
    return null;
}
}
```

推理流程实现 – 例子 – 文字检测

// 加载模型

```
Criteria<Image, DetectedObjects> criteria = Criteria.builder()
    .optEngine( "PaddlePaddle" )
    .setTypes(Image.class, DetectedObjects.class)
    .optModelPath( "/Users/calvin/Documents/ch_PP-OCRv2_det_infer" )
    .optTranslator(new PpWordDetectionTranslator ())
    .optProgress(new ProgressBar())
    .build();
ZooModel<Image, DetectedObjects> model = criteria.loadModel();
```

// 运行推理

```
Predictor< Image, DetectedObjects> predictor = model.newPredictor();
predictor.predict(list);
```

// 前处理

```
public NDList processInput(TranslatorContext ctx, Image input) {
    NDArrary img = input.toNDArrary(ctx.getNDManager());
    int h = input.getHeight();
    int w = input.getWidth();
    int[] hw = this.scale(h, w, this.maxLength);
    img = NDImageUtils.resize(img, hw[1], hw[0]);
    img = NDImageUtils.toTensor(img);
    img = NDImageUtils.normalize(img, new float[]{0.485F, 0.456F, 0.406F}, new
float[]{0.229F, 0.224F, 0.225F});
    img = img.expandDims(0);
    return new NDList(new NDArrary[]{img});
}
```

// 后处理

```
public DetectedObjects processOutput(TranslatorContext ctx, NDList list) {
    NDArrary result = list.singletonOrThrow();
    result = result.squeeze().mul(255.0F).toType(DataType.UINT8, true).neq(0);
    boolean[] flattened = result.toBooleanArray();
    Shape shape = result.getShape();
    int w = (int)shape.get(0);
    int h = (int)shape.get(1);
    boolean[][] grid = new boolean[w][h];
    ...
    return new DetectedObjects(names, probs, boxes);
}
```

如何使用java部署paddleOCR模型？

<https://gitee.com/paddlepaddle/PaddleOCR>

PP-OCR系列模型列表（更新中）

模型简介	模型名称	推荐场景	检测模型	方向分类器	识别模型
中英文超轻量PP-OCRv2模型 (13.0M)	ch_PP-OCRv2_xx	移动端&服务器端	推理模型 / 训练模型	推理模型 / 预训练模型	推理模型 / 训练模型
中英文超轻量PP-OCR mobile模型 (9.4M)	ch_ppocr_mobile_v2.0_xx	移动端	训练模型	训练模型	训练模型
中英文通用PP-OCR server模型 (143.4M)	ch_ppocr_server_v2.0_xx	服务器端	推理模型 / 预训练模型	推理模型 / 预训练模型	推理模型 / 预训练模型

首先：如何下载模型？

ch_PP-OCRv2_det_infer >

inference.pdiparams
inference.pdiparams.info
inference.pdmodel

// 加载模型

```
Criteria<Image, DetectedObjects> criteria = Criteria.builder()
    .optEngine( "PaddlePaddle" )
    .setTypes(Image.class, DetectedObjects.class)
    .optModelUrls( "/Users/calvin/Documents/ch_PP-OCRv2_det_infer" )
    .optTranslator(new PpWordDetectionTranslator ())
    .optProgress(new ProgressBar())
    .build();
ZooModel<NDList, NDList> model = criteria.loadModel();
```

// 运行推理

```
Predictor<NDList, NDList> predictor = model.newPredictor();
NDManager manager = NDManager.newBaseManager()
NDList list ; // the input for your models as pure NDArray
predictor.predict(list);
```

PaddleOCR的java实现

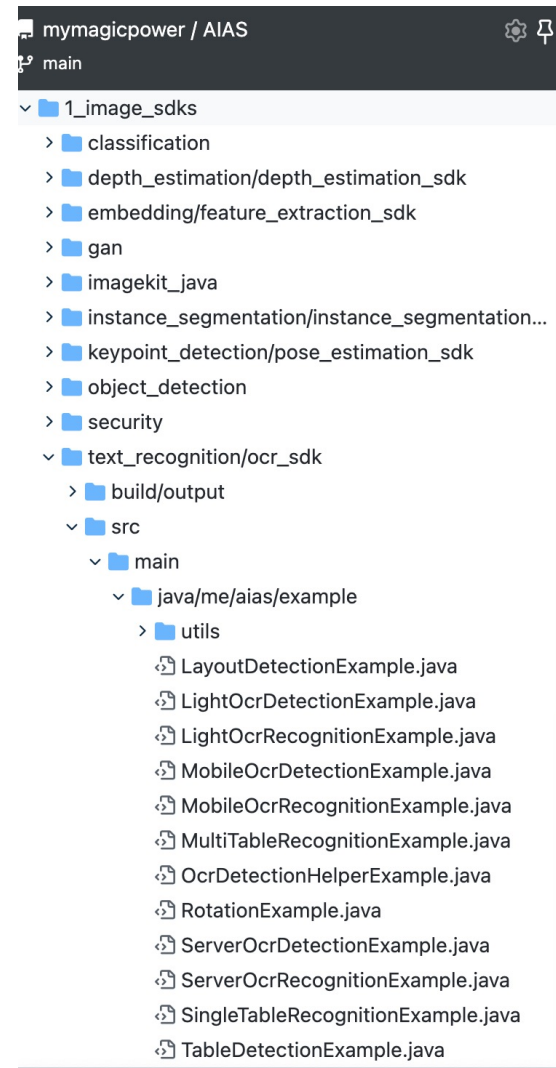
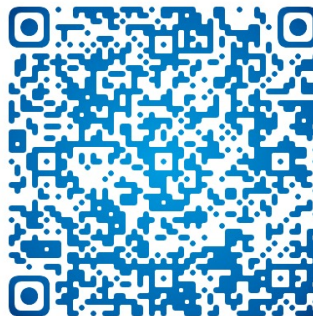
实现的功能：

- 文字方向检测
- 文字检测
- 文字识别
- 版面分析
- 表格识别

Github 项目地址



Gitee 项目地址



SDK功能 - 文字检测

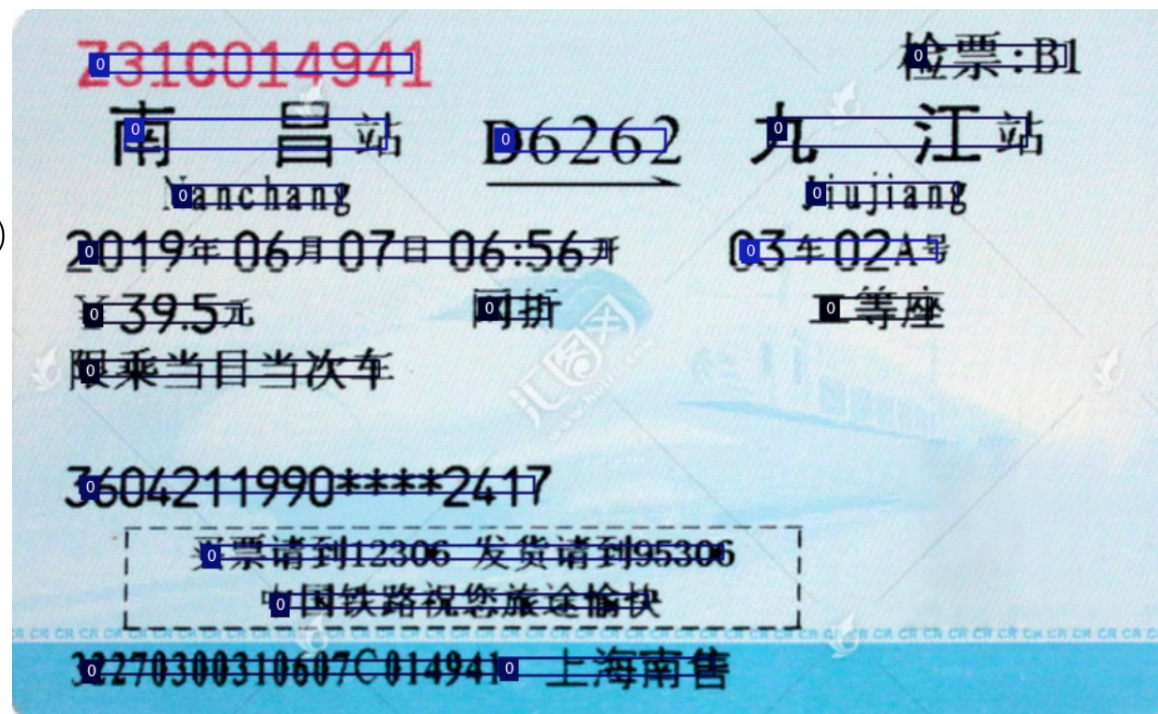
// 加载模型

```
public Criteria<Image, DetectedObjects> detectCriteria() {  
    Criteria<Image, DetectedObjects> criteria =  
        Criteria.builder()  
            .optEngine("PaddlePaddle")  
            .setTypes(Image.class, DetectedObjects.class)  
            .optModelPath("/Users/calvin/Documents/ch_PP-OCRv2_det_infer.zip")  
            .optTranslator(new PpWordDetectionTranslator(new  
                ConcurrentHashMap<String, String>()))  
            .optProgress(new ProgressBar())  
            .build();  
    ZooModel<Image, DetectedObjects> model = criteria.loadModel();  
}
```

// 运行推理

```
Predictor<Image, DetectedObjects> predictor = model.newPredictor();  
predictor.predict(img);
```

- 输出图片效果如下：



SDK功能 - 文字方向分类

文字方向识别：

0度 90度 180度 270度

// 加载模型

```
Criteria<Image, Classifications> criteria = Criteria.builder()  
    .optEngine( "PaddlePaddle" )  
    .setTypes(Image.class, Classifications.class)  
    .optModelPath( "/Users/calvin/Documents/ch_ppocr_mobile_v2.0_cls_infer.zip" )  
    .optTranslator(new PpWordRotateTranslator ())  
    .optProgress(new ProgressBar())  
    .build();
```

```
ZooModel< Image, Classifications> model = criteria.loadModel();
```

// 运行推理

```
Predictor< Image, Classifications> predictor = model.newPredictor();  
predictor.predict(list);
```

- 输出图片效果如下：



SDK功能 - 文字识别

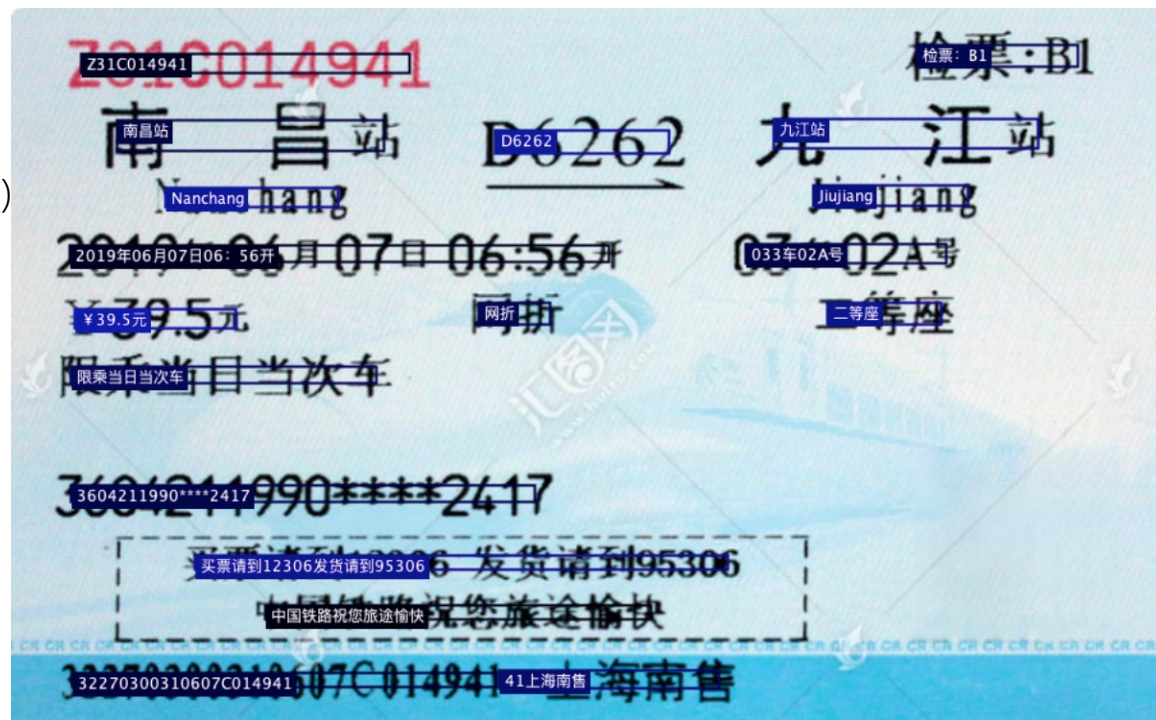
// 加载模型

```
Criteria<Image, String> criteria =  
    Criteria.builder()  
        .optEngine("PaddlePaddle")  
        .setTypes(Image.class, String.class)  
        .optModelPath("/Users/calvin/Documents/ch_PP-OCRv2_rec_infer.zip")  
        .optProgress(new ProgressBar())  
        .optTranslator(new PpWordRecognitionTranslator())  
        .build();  
ZooModel< Image, String> model = criteria.loadModel();
```

// 运行推理

```
Predictor< Image, String > predictor = model.newPredictor();  
predictor.predict(img);
```

- 输出图片效果如下：



SDK功能 - 版面分析

Text, Title, List, Table, Figure

// 加载模型

```
Criteria<Image, DetectedObjects> criteria =
```

```
Criteria.builder()
```

```
.optEngine("PaddlePaddle")
```

```
.setTypes(Image.class, DetectedObjects.class)
```

```
.optModelPath(
```

```
    "/Users/calvin/Documents/pp yolov2_r50vd_dcn_365e_publaynet_infer.zip")
```

```
.optTranslator(new LayoutDetectionTranslator())
```

```
.optProgress(new ProgressBar())
```

```
.build();
```

```
ZooModel<Image, DetectedObjects> model = criteria.loadModel();
```

// 运行推理

```
Predictor<Image, DetectedObjects> predictor = model.newPredictor();
```

```
predictor.predict(img);
```

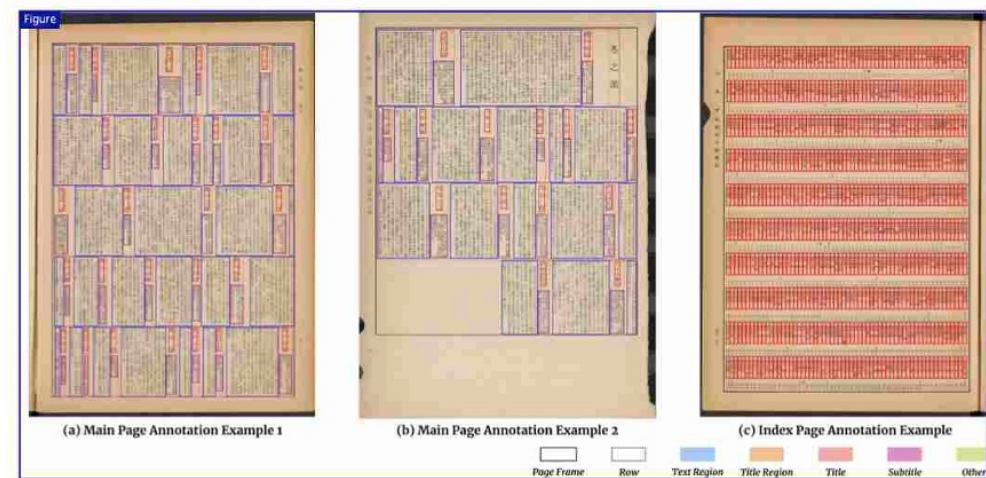


Figure 7: Annotation Examples in HJDataset. (a) and (b) show two examples for the labeling of main pages. The boxes are colored differently to reflect the layout element categories. Illustrated in (c), the items in each index page row are categorized as title blocks, and the annotations are denser.

Text The over union (IOU) level [0.50:0.95]², on the test data. In general, the high mAP values indicate accurate detection of the layout elements. The Faster R-CNN and Mask R-CNN achieve comparable results, better than RetinaNet. Noticeably, the detections for small blocks like title are less precise, and the accuracy drops sharply for the title category. In Figure 8, (a) and (b) illustrate the accurate prediction results of the Faster R-CNN model.

Pre-training for other datasets

Text We also examine how our dataset can help with a real-world document digitization application. When digitizing new publications, researchers usually do not generate large scale ground truth data to train their layout analysis models. If they are able to adapt our dataset, or models trained on our dataset, to develop models on their data, they can build their pipelines more efficiently and develop more accurate models. To this end, we conduct two experiments. First we examine how layout analysis models trained on the main pages can be used for understanding index pages. Moreover, we study how the pre-trained models perform on other historical Japanese documents.

Text Table 4 compares the performance of five Faster R-CNN models that are trained differently on index pages. If the model loads pre-trained weights from HJDataset, it includes information learned from main pages. Models trained over

Text This is a core metric developed for the COCO competition [42] for evaluating the object detection quality.

Text The training data can be viewed as the benchmarks, while training with few samples (five in this case) are considered to mimic real-world scenarios. Given different training data, models pre-trained on HJDataset perform significantly better than those initialized with COCO weights. Intuitively, models trained on more data perform better than those with fewer samples. We also directly use the model trained on main to predict index pages without fine-tuning. The low zero-shot prediction accuracy indicates the dissimilarity between index and main pages. The large increase in mAP from 0.344 to 0.471 after the model is

Text Table 3: Detection mAP @ IOU [0.50:0.95] of different models for each category on the test set. All values are given as percentages.

Category	Faster R-CNN	Mask R-CNN*	RetinaNet
Page Frame	99.046	99.097	99.038
Row	98.831	98.482	95.067
Title Region	87.571	89.483	69.593
Text Region	94.463	86.798	89.531
Title	65.908	71.517	72.566
Subtitle	84.093	84.174	85.865
Other	44.023	39.849	14.371
mAP	81.991	81.343	75.223

Text* training Mask R-CNN, the segmentation masks are the quadrilateral regions for each block. Compared to the rectangular bounding boxes, they delineate the text region more accurately.

SDK功能 – 表格识别

// 加载模型

```
Criteria<Image, TableResult> criteria =
```

```
Criteria.builder()
```

```
.optEngine("PaddlePaddle")
```

```
.setTypes(Image.class, TableResult.class)
```

```
.optModelUrls()
```

```
    "/Users/calvin/Documents/models/ocr_models/en_tab
```

```
.optOption("removePass", "repeated_fc_relu_fuse_pass")
```

```
.optTranslator(new TableStructTranslator(new ConcurrentHashMap<String, String>()))
```

```
.optProgress(new ProgressBar())
```

```
.build();
```

```
ZooModel<Image, TableResult> model = criteria.loadModel();
```

// 运行推理

```
Predictor<Image, TableResult> predictor = model.newPredictor();
```

```
predictor.predict(img);
```

Methods	R	P	F	FPS
SegLink [26]	70.0	86.0	77.0	8.9
PixelLink [4]	73.2	83.0	77.8	-
TextSnake [18]	73.9	83.2	78.3	1.1
TextField [37]	75.9	87.4	81.3	36
MSR [38]	76.7	87.4	81.7	-
FTSN [3]	77.1	87.6	82.0	-
LSE [30]	81.7	84.2	82.9	-
CRAFT [2]	78.2	88.2	82.9	8.6
MCNJ [16]	79	88	83	-
ATRR [35]	82.1	85.2	83.6	-
PAN [34]	83.8	84.4	84.1	30.2
DB [12]	79.2	91.5	84.9	32.0
DRRG [41]	82.30	88.05	85.08	-
Ours (SynText)	80.68	85.40	82.97	12.68
Ours (MLT-17)	84.54	86.62	85.57	12.31



Methods	R	P	F	FPS
SegLink[26]	70.0	86.0	77.0	8.9
PixelLink[4]	73.2	83.0	77.8	-
TextSnake[18]	73.9	83.2	78.3	1.1
TextField[37]	75.9	87.4	81.3	36
MSR38	76.7	87.4	81.7	-
FTSN3	77.1	87.6	82.0	-
LSE[30]	81.7	84.2	82.9	-
CRAFT[2]	78.2	88.2	82.9	8.6
MCNJ161	79	88	83	-
ATRR[35]	82.1	85.2	83.6	-
PAN[34]	83.8	84.4	84.1	30.2
DB[12]	79.2	91.5	84.9	32.0
DRRG141	82.30	88.05	85.08	-
Ours(SynText)	80.68	85.40	82.97	12.68
Ours (MLT-17)	84.54	86.62	85.57	12.31

	A	B	C	D	E	F
1	Methods	R	P	F	FPS	
2	SegLink[26]	70	86	77.0	8.9	
3	PixelLink[4j	73.2	83	77.8		
4	TextSnake18]	73.9	83.2	78.3	1.1	
5	TextField[37]	75.9	87.4	81.3	36	
6	MSR38]	76.7	87.4	81.7		
7	FTSN3]	77.1	87.6	82		
8	LSE[30]	81.7	84.2	82.9	—	
9	CRAFT[2]	78.2	88.2	82.9	8.6	
10	MCNJ161	79	88	83		
11	ATRR[35]	82.1	85.2	83.6		
12	PAN[34]	83.8	84.4	84.1	30.2	
13	DB[12]	79.2	91.5	84.9	32	
14	DRRG141]	82.3	88.05	85.08		
15	Ours (SynText)	80.68	85.4	82.97	12.68	
16	Ours (MLT-17)	84.54	86.62	85.57	12.31	

目录

CONTENT

项目背景

实现机制与核心代码

性能调优与提高识别率的办法

AIAS开源项目介绍

1. 多线程环境，设置多少个predictor合适？

一般单机有多个predictor，predictor数目可以和核心数相同，这样请求来了就可以打满。但是如果并发数超过单机核心数，而且没法批处理，那么这时候多机会派上用场帮忙分流...

2. 多线程环境，并行数如何设置？

答：深度学习框架默认的并行数就是核心的数目。

比如：单机两个gpu，模型需要load两份，一份load在gpu0，一份在gpu1

之后从两个model分别开多个predictor，每个predictor会绑定在最初model读到的gpu设备上。

此时因为CPU计算量变小，可以开双倍于CPU核心数的线程。

具体数字多少可以测试多线程推理，什么时候gpu或者cpu利用率打满。

<https://docs.djl.ai/extensions/benchmark/index.html>

3. 如果使用GPU，多线程会占用多份显存？

从一个model创建的多个predictor会共享一份模型内存，不会造成过多占用。所以，Model可以复用，如果是多线程下使用，建多个predictor，

每个线程一个predictor。

**这算是一个多线程推理吊打python的多进程推理的核心优势。 **

补充说明：



1. 性能调优文档

http://docs.djl.ai/docs/development/inference_performance_optimization.html

2. Java本身是多线程的，DJI其实也是调用了相关引擎的C++ API。多线程本身知识就不赘述了，我们在调用这些api上下了点功夫，保证使用过程是线程安全的。推理任务来了以后，每个线程同时调用这个C++ API。然后到底层每个cpu核心会操作一个请求，每个核心之前计算是互不影响的，之后处理完返回到java端。

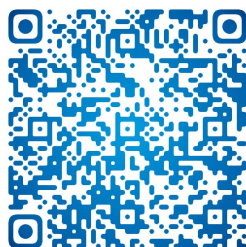
3. graph在推理时是read only的，每个核心去使用内存计算相互结果不影响，所以最后也没事。

准确率调优 - 图像预处理

Github 项目地址



Gitee 项目地址



在OCR文字识别的时候，我们得到的图像一般情况下都不是正的，多少都会有一定的倾斜。所以需要将图片转正。并且图片有可能是透视视角拍摄，需要重新矫正。

SDK功能

- 图像转正
- 图像二值化，灰度化，去噪等经典算法。



常见问题 - windows

Windows环境问题：

Windows 10 加载失败常常是因为缺少 Windows Visual C++ 相关扩展包而导致的。您可以通过扫描右侧二维码，根据 Windows 的步骤来修复系统缺失依赖项。



<http://www.dependencywalker.com/>
这个软件载入DLL看一下缺什么依赖
(或者重装了一下vc , 解决问题)

常见问题清单：

<https://docs.djl.ai/docs/development/troubleshooting.html>

常见问题 – 多线程

多线程关键点：

- 使用固定数量线程池
- CPU场景：与核心数一致
- GPU场景：此时因为CPU计算量变小，可以开双倍于CPU核心数的线程
- 每个线程实例化一个predictor
- CPU场景共享一个模型
- GPU场景，一个GPU加载一个模型

```
ExecutorService es = Executors.newFixedThreadPool(threadNum);  
long timeInferStart = System.currentTimeMillis();  
for (InferCallable callable : callables) {  
    futures.add(es.submit(callable));  
}
```

```
public static class InferCallable implements Callable<List<ImageInfo>> {  
    private Predictor<Image, DetectedObjects> predictor;  
    private DetectedObjects detections = null;  
    private ConcurrentLinkedQueue<ImageInfo> queue;  
    private List<ImageInfo> resultList = new ArrayList<>();  
  
    public InferCallable(ZooModel<Image, DetectedObjects> model, Concurrent  
        predictor = model.newPredictor();  
        this.queue = queue;  
}
```

目录

CONTENT

项目背景

实现机制与核心代码

性能调优与提高识别率的办法

AIAS开源项目介绍

AIAS - AI加速器套件

AIAS (AI Acceleration Suite) - 人工智能加速器套件。
提供: 包括SDK, 平台引擎, 场景套件在内, 合计超过100个项目组成的项目集。

- ✓ 提供 SDK, 平台引擎, 场景套件
- ✓ 加速各业务场景落地

- ✓ 为集成商赋能
- ✓ 为企业内部项目赋能

飞桨 Paddle



SDK

Image SDKs

NLP SDKs

Audio SDKs

Video SDKs

Bigdata SDKs

Biomedicine SDKs



平台引擎

API Platform

Search Engine

Training Engine

Edge Computing Engine

.....



场景套件

智慧城市

智慧交通

智慧医院

融媒体

智慧社区

智慧楼宇

智慧零售

RPA

智慧安防

金融

机器人

无人驾驶

文娱

.....

Data Hub

Model Hub

PyTorch

Pytorch Hub

mxnet

MxNet Hub

TensorFlow

TensorFlow Hub

飞桨 Paddle

Paddle Hub

Hugging Face

Huggingface Hub

进行中

TODO



AIAS涉及的范围



AIAS - AI加速器套件

Github 项目地址



Github:

<https://github.com/mymagicpower/AIAS>

Gitee:

<https://gitee.com/mymagicpower/AIAS>

Gitee 项目地址



SDK - 计算机视觉：

- 1).目标检测：目标检测、人脸检测&识别
- 2).图像分割：图像分割
- 3).行为分析：行为识别、姿态估计
- 4).GAN：超分辨率、动作驱动、去噪、风格迁移
- 5).其它类别：OCR、深度估计、图像融合、图像检索

SDK - 自然语言处理：

- 1).文本生成
- 2).词向量
- 3).机器翻译
- 4).语义模型
- 5).情感分析
- 6).句法分析
- 7).词法分析
- 8).文本审核

SDK - 语音识别：

- 1).声音克隆
- 2).语音合成
- 3).声音分类

平台引擎：

- 1).训练引擎
- 2).搜索引擎
- 3).API能力平台
- 4).边缘计算引擎

场景套件：

- 1).OCR自定义模版识别
- 2).人脸搜索
- 3).问答系统
- 4).生物医药

基于PaddlePaddle实现的sdk

PaddleDetection:

1. 人脸检测：
http://aias.top/AIAS/1_image_sdks/security/face_sdks/face_sdk.html
2. 人脸关键点：
http://aias.top/AIAS/1_image_sdks/security/face_sdks/face_landmark_sdk.html
3. 口罩检测：
http://aias.top/AIAS/1_image_sdks/security/face_sdks/mask_sdk.html
4. 人车非（人、机动车，非机动车）识别：
http://aias.top/AIAS/1_image_sdks/security/traffic_sdk.html
5. 行人检测：
http://aias.top/AIAS/1_image_sdks/security/pedestrian_sdk.html
6. 车辆检测：
http://aias.top/AIAS/1_image_sdks/security/vehicle_sdk.html

PaddleNLP:

1. 文本 - 词法分析：
http://aias.top/AIAS/2_nlp_sdks/lexical_analysis/lac_sdk.html
2. 文本 - 情感分析[中文]：
http://aias.top/AIAS/2_nlp_sdks/sentiment_analysis/senta_bilstm_sdk.html
3. 文本 - 对话情绪识别[中文]：
http://aias.top/AIAS/2_nlp_sdks/sentiment_analysis/senta_textcnn_sdk.html
4. 中文翻译为英文：
http://aias.top/AIAS/2_nlp_sdks/machine_translation/translation_zh_en_sdk.html
5. 英文翻译为德文：
http://aias.top/AIAS/2_nlp_sdks/machine_translation/translation_en_de_sdk.html
6. 词向量[中文]：
http://aias.top/AIAS/2_nlp_sdks/embedding/word_encoder_cn_sdk.html
7. 词向量[英文]：
http://aias.top/AIAS/2_nlp_sdks/embedding/word_encoder_en_sdk.html
8. 文本 - 短文本相似度[中文]：
http://aias.top/AIAS/2_nlp_sdks/language_model/semantic_simnet_bow_sdk.html

PaddleCls:

1. 动物分类识别：
http://aias.top/AIAS/1_image_sdks/classification/animal_sdk.html
2. 菜品分类识别
http://aias.top/AIAS/1_image_sdks/classification/dish_sdk.html

PaddleOCR:

1. 方向检测，文本检测，文本识别，版面分析，表格识别：
http://aias.top/AIAS/1_image_sdks/text_recognition/ocr_sdk.html
2. OCR 自定义模板识别(支持表格识别)：
http://aias.top/AIAS/8_suite_hub/iocr.html

PaddleSpeech:

1. 语音识别（ASR）【短语音】：
http://aias.top/AIAS/3_audio_sdks/asr_sdk.html
2. 语音识别（ASR）【长语音】：
http://aias.top/AIAS/3_audio_sdks/asr_long_audio_sdk.html
3. 声纹识别：
http://aias.top/AIAS/3_audio_sdks/voiceprint_sdk.html

其它：

1. 人群密度检测：
http://aias.top/AIAS/1_image_sdks/security/crowd_sdk.html
2. 单目深度估计：
http://aias.top/AIAS/1_image_sdks/depth_estimation/depth_estimation_sdk.html
3. 音频搜索 - 声纹搜索：
http://aias.top/AIAS/7_engine_hub/voiceprint_search.html

iOCR自定义模板文字识别

飞桨

基于模板文字识别

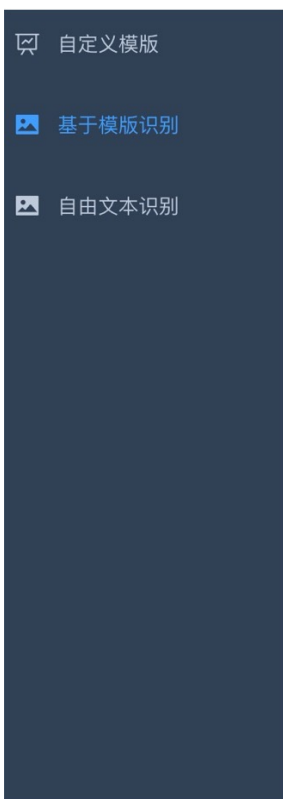
Github 项目地址



Gitee 项目地址



一般OCR的识别结果是一种按行识别的结构化输出，能够给出一行文字的检测框坐标及文字内容。但是我们更想要的是带有字段定义的结构化输出，由于表单还活着卡证的多样性，全都预定义好是不现实的。所以，设计了自定义模板的功能，能够让人设置参照锚点（通过锚点匹配定位，图片透视变换对齐），以及内容识别区来得到key-value形式的结构化数据。



iOCR自定义模板文字识别

参照锚点设置规则：

1. 建议框选4个及以上(最少3个)参照字段，并尽量分散（向四角方向）
 - 1). 如果匹配4个及以上的锚点框，则进行透视变换
 - 2). 如果匹配3个锚点框，则进行仿射变换
 - 3). 如果匹配的锚点少于三个则直接根据相对坐标计算
2. 参照锚点必须是位置固定不变，文字固定不变
3. 单个参照字段不可跨行，且中间没有大片空白
4. 参照锚点文字内容需唯一，即不会重复出现的文字

内容识别区设置规则：

1. 识别结果以<key,value>形式展示，字段名需手工设置
2. 字段名需使用有业务意义的字母数字组合，如：name，age，address
3. 字段名不能含有特殊字符及空格



Demo



谢谢观看