

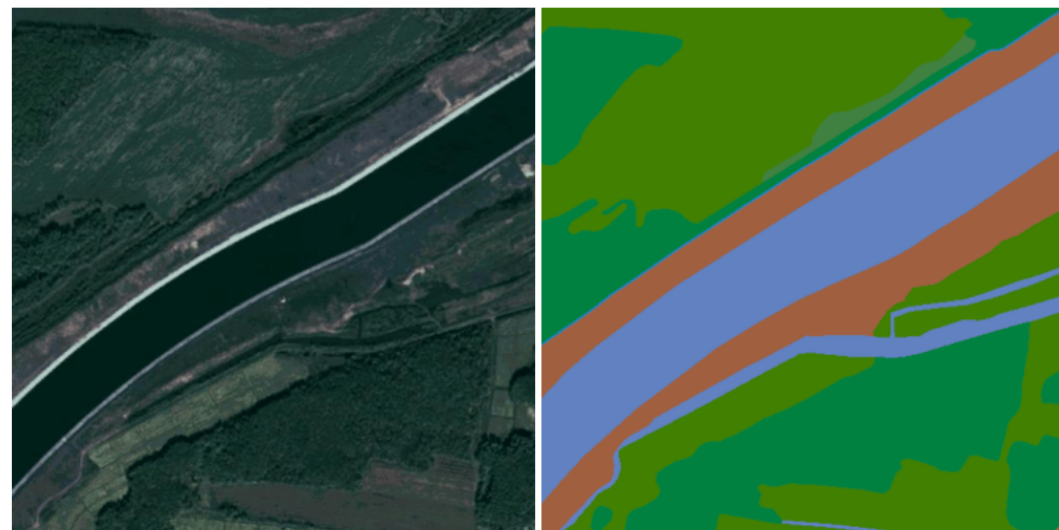
2020 CCF BDCI: 遥感影像地块分割

兰冬雷

2020E8015382008

- 使用百度飞桨（**PaddlePaddle**）搭建深度学习模型，对遥感影像数据进行地块分割。
- 遥感图像具体包括建筑、耕地、林地、水体、道路、草地和其他等 7 个类别。
- 提交的预测结果为 zip 压缩包（**ccf_baidu_remote_sense.zip**），压缩包中包含与测试集（**img_testA**）中的文件名相同的单通道 PNG 图片。

- 关注如何将图像分割成属于不同语义类别的区域，这些语义区域的标注和预测都是像素级的。
- 与目标检测相比，语义分割标注的像素级的边框显然更加精细。
- 语义分割只能判断类别，无法区分个体。





img_train

1.16 GB, 145,981个项目

上次修改时间 2020 年 10 月 10 日 下午 1:57:38



img_testA

80.8 MB, 10,000个项目

上次修改时间 2020 年 10 月 10 日 下午 1:57:41

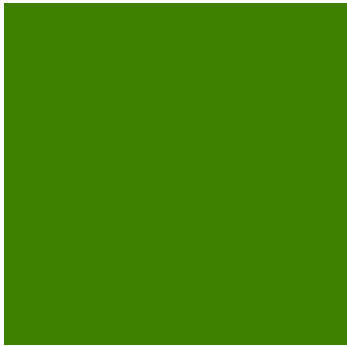
Train Set: 145,981

Test Set: 10,000

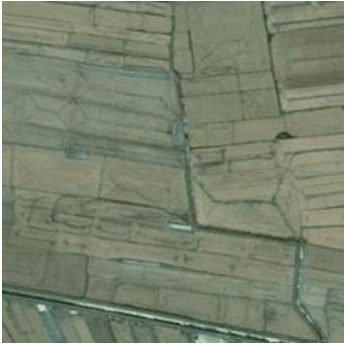
类别	像素值
建筑	0
耕地	1
林地	2
水体	3
道路	4
草地	5
其他	6
未标注数据	255



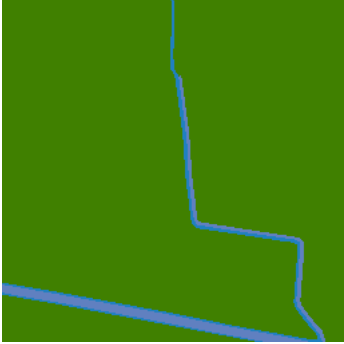
T072991.jpg



T072991.png



T000003.png



T000003.jpg

训练样本及其标注

创建数据集

数据集名称

为您的数据集起个名字，最多100个字

上传数据集

上传文件

1.最多上传10个文件

2. 总体积不超过50GB，文件名不超过50字

简介摘要

此数据集包含什么内容，可做什么用，最多200个字

确定

取消

1. 上传数据集

CCF-Demo	
文件夹	数据集
	ddd
数据集	img_testA.zip
	img_train.zip
环境	lab_train.zip

2. 数据集上传完成

Notebook 终端-1 X

```
aistudio@jupyter-536964-1153502:~$ cd data/data57012/ && ls
img_testA.zip  img_train.zip  lab_train.zip
aistudio@jupyter-536964-1153502:~/data/data57012$
```

3. 在终端使用命令操作数据集

- FCN : Fully Convolutional Networks 全卷积神经网络：从抽象的特征中恢复出每个像素所属的类别，即从图像级别的分进一步延伸到像素级别分类。
- Heatmap 热图
- 反卷积 Transposed Convolution、upsampling

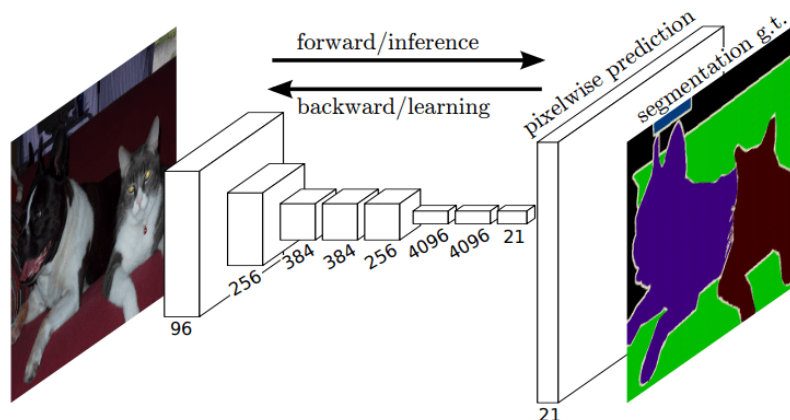


Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.

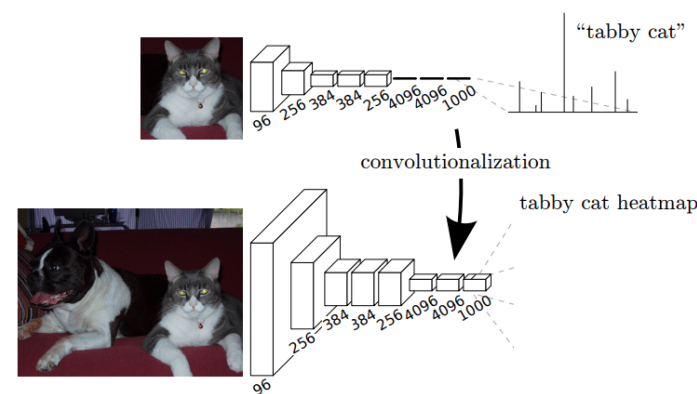
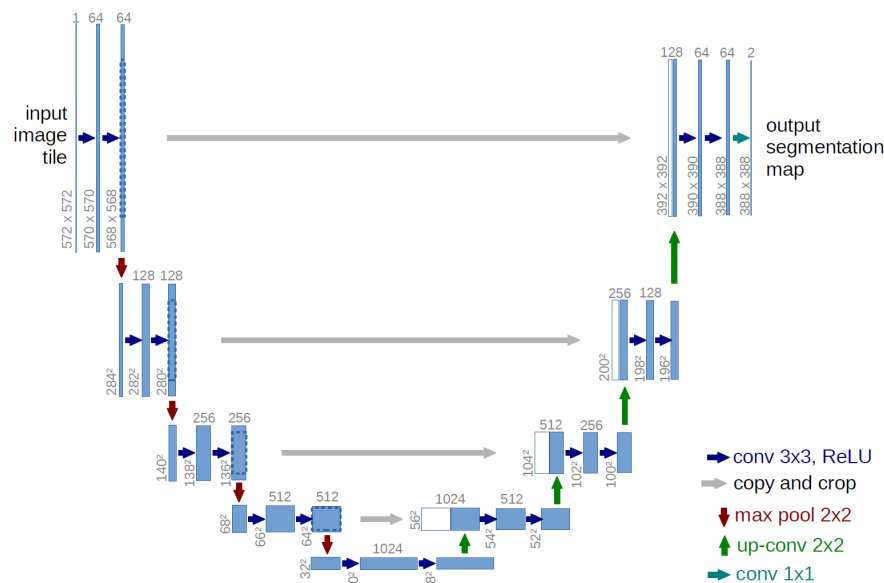


Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.

- U-Net 是 FCN 的一个改进。
- U-Net 最大的特点是 “U ” 型对称结构和 skip connection。
- U-Net 的 encoder 下采样 4 次，一共下采样 16 倍。对称地，其 decoder 也相应上采样 4 次，将 encoder 得到的高级语义特征图恢复到原图片的分辨率。
- U-Net 在医学图像分割上表现的特别好，成为大多做医疗影像语义分割任务的 baseline。



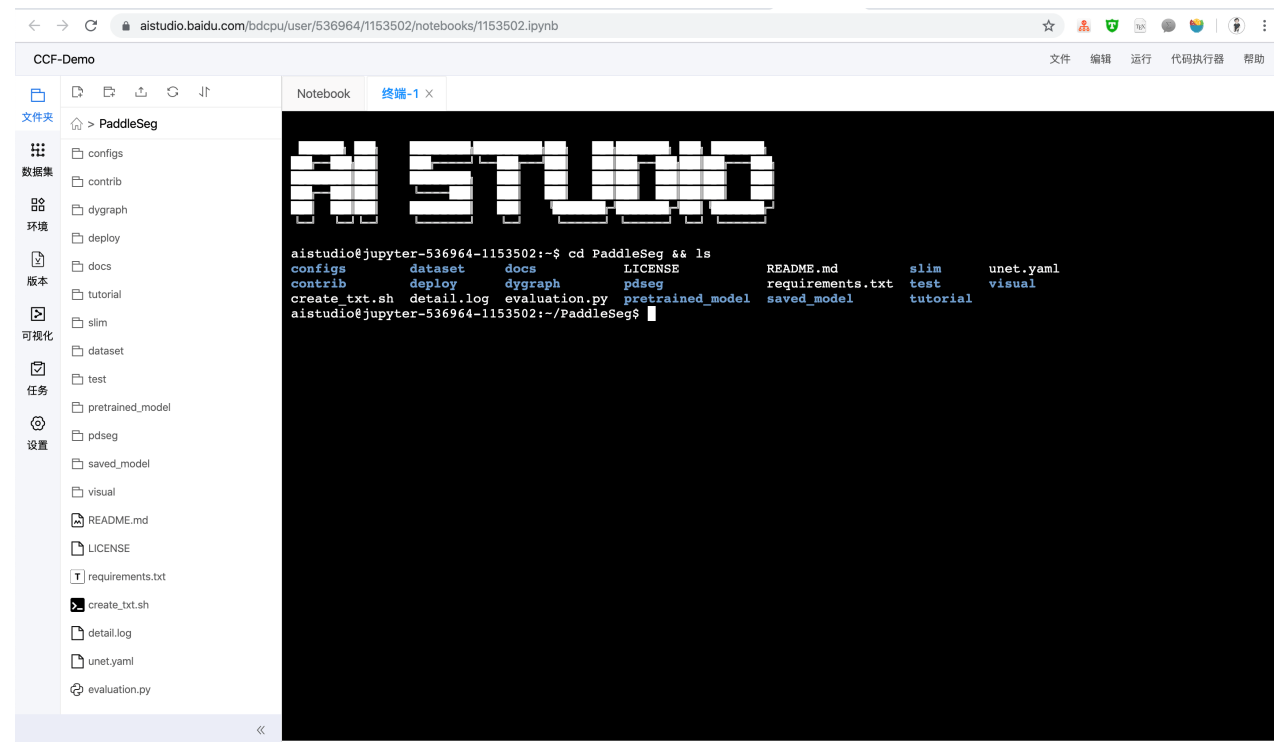
- 一些常见的图像分割网络包括：SegNet、RefineNet、PSPNet、DeepLab、Mask-RCNN、DenseNet 等。
- 在遥感图像分割比赛中，提供一些预训练好的分割网络，包括：deeplabv3、unet、pspnet、hrnet、fast_scnn、ocrnet。
- 图像分割常用的数据集

Pascal VOC / MS COCO / City Scapes

- 飞桨（[PaddlePaddle](https://www.paddlepaddle.org.cn/)）是百度自主研发、功能完备、开源开放的产业级深度学习平台，集深度学习核心训练和推理框架、基础模型库、端到端开发套件和丰富的工具组件于一体。与 PyTorch、TensorFlow 一样，是一个深度学习框架，基于这些深度学习框架，开发者可以快速的搭建深度学习模型。
- 百度 AI Studio 提供了免费的计算资源，以及一些基本的训练方法和比赛的评测脚本可以加快进展。



- 这里我将工程名称命名为「CCF-Demo」。
- 百度 AI Studio 提供了免费的计算资源
(CPU=2/RAM=8GB/磁盘=100GB)。还有大量算力卡，可以使用 Tesla V100 GPU、16GB Video Mem、4 Core CPU 及 32GB RAM 加快模型的训练。
- PaddleSeg 里面提供的一些脚本 (`train.py`、`eval.py` 等) 能提供帮助。因此，可以让我们更加专注于模型的调优上。



```
git clone https://github.com/PaddlePaddle/PaddleSeg
```

- 从大赛官网下载数据集后，进入「个人中心」，「创建数据集」。
- 将下载好的数据集上传到 **AI Studio**，就可以在项目工程中使用 **cp** 命令拷贝数据集到任意指定路径下。

创建数据集

数据集名称

为您的数据集起个名字，最多100个字

上传数据集

上传文件

1.最多上传10个文件
2. 总体积不超过50GB，文件名不超过50字

简介摘要

此数据集包含什么内容，可做什么用，最多200个字

确定

取消

1. 上传数据集

Notebook 终端-1 ×

```
aistudio@jupyter-536964-1153502:~$ cd data/data57012/ && ls
img_testA.zip  img_train.zip  lab_train.zip
aistudio@jupyter-536964-1153502:~/data/data57012$
```

3. 在终端使用命令操作数据集

CCF-Demo	
数据集	
文件夹	ddd
数据集	img_testA.zip
	img_train.zip
环境	lab_train.zip

2. 数据集上传完成

- 使用 `find`、管道 `|` 和重定向命令 `>` 创建 `txt`。
- 将训练集中 `1/50` 数据划分为验证集，用于超参数的调优。

```
#!/bin/bash
find img_testA -type f | sort > testA_list.txt
echo "Create text_list.txt"

find img_train -type f | sort > train.ccf.tmp
find lab_train -type f | sort > train.lab.ccf.tmp
paste -d " " train.ccf.tmp train.lab.ccf.tmp > all.ccf.tmp

awk '{if (NR % 50 != 0) print $0}' all.ccf.tmp > train_list.txt
awk '{if (NR % 50 == 0) print $0}' all.ccf.tmp > val_list.txt

rm *.ccf.tmp
echo "Create train_list.txt and val_list.txt."
```

```
1  # pretrained_model/download_model.py
2  import sys
3  import os
4
5  LOCAL_PATH = os.path.dirname(os.path.abspath(__file__))
6  TEST_PATH = os.path.join(LOCAL_PATH, "..", "test")
7  sys.path.append(TEST_PATH)
8  from test_utils import download_file_and_uncompress
9
10 model_urls = {
11     # ImageNet Pretrained
12     # ...
13
14     # COCO pretrained
15     "unet_bn_coco":
16     "https://paddleseg.bj.bcebos.com/models/unet_coco_v3.tgz",
17
18     # Cityscapes pretrained
19     #...
20 }
```

```
22 model_name = sys.argv[1]
23 url = model_urls[model_name]
24 download_file_and_uncompress(
25     url=url,
26     savepath=LOCAL_PATH,
27     extrapath=LOCAL_PATH,
28     extraname=model_name)
```

```
1 # 数据集配置
2 DATASET:
3     DATA_DIR: "./dataset/rs_data"
4     NUM_CLASSES: 7
5     TRAIN_FILE_LIST: "./dataset/rs_data/train_list.txt" # 训练集
6     VAL_FILE_LIST: "./dataset/rs_data/val_list.txt" # 验证集
7     TEST_FILE_LIST: "./dataset/rs_data/testA_list.txt" # 测试集
8     VIS_FILE_LIST: "./dataset/rs_data/testA_list.txt"
```

```
10 # 预训练模型配置
```

```
11 MODEL:
12     MODEL_NAME: "unet"
13     DEFAULT_NORM_TYPE: "bn"
```

```
15 # 其他配置
```

```
16 TRAIN_CROP_SIZE: (256, 256)
17 EVAL_CROP_SIZE: (256, 256)
18 AUG:
19     AUG_METHOD: "unpadding"
20     FIX_RESIZE_SIZE: (256, 256)
21 BATCH_SIZE: 32
22 TRAIN:
23     PRETRAINED_MODEL_DIR: "./pretrained_model/unet_bn_coco/" # 预训练模型
24     MODEL_SAVE_DIR: "./saved_model/unet/"
25     SNAPSHOT_EPOCH: 1
26 TEST:
27     TEST_MODEL: "./saved_model/unet/final"
```

```
28 # 超参数配置值
```

```
29 SOLVER:
```

```
30     NUM_EPOCHS: 100 # 训练 100 个 EPOCH
```

```
31     LR: 0.01
```

```
32     LR_POLICY: "poly" # 学习率衰减策略
```

```
33     OPTIMIZER: "sgd" # 优化方法采用随机梯度下降
```

- 官方提供的基础代码库 PaddleSeg，里面提供的一些脚本（train.py、eval.py 等）能提供帮助。
- 因此，可以让我们更加专注于模型的调优上。

```
1 # 指定 GPU 卡号
2 aistudio@jupyter-536964-1153502:~/PaddleSeg$ export CUDA_VISIBLE_DEVICES=0
3 # 开始训练
4 aistudio@jupyter-536964-1153502:~/PaddleSeg$ python pdseg/train.py --use_gpu --cfg
  ./unet.yaml
```

- 1 个 EPOCH 耗时 \approx 30 分钟，此时的学习率从 0.01 衰减到了 0.00982，**loss = 0.3859**。

```
epoch=1 step=4470 lr=0.00982 loss=0.3859 step/sec=2.264 | ETA 26:52:23  
Save model checkpoint to ./saved_model/unet/1
```

```
epoch=2 step=8940 lr=0.00964 loss=0.3193 step/sec=2.259 | ETA 26:23:12  
Save model checkpoint to ./saved_model/unet/2
```

```
epoch=3 step=13400 lr=0.00946 loss=0.3018 step/sec=2.262 | ETA 25:48:07  
epoch=3 step=13410 lr=0.00946 loss=0.2968 step/sec=2.262 | ETA 25:47:44  
Save model checkpoint to ./saved_model/unet/3
```

```
epoch=4 step=17880 lr=0.00928 loss=0.2857 step/sec=2.261 | ETA 25:15:35  
Save model checkpoint to ./saved_model/unet/4
```

```
epoch=5 step=22350 lr=0.00910 loss=0.3391 step/sec=2.262 | ETA 24:42:08  
Save model checkpoint to ./saved_model/unet/5
```

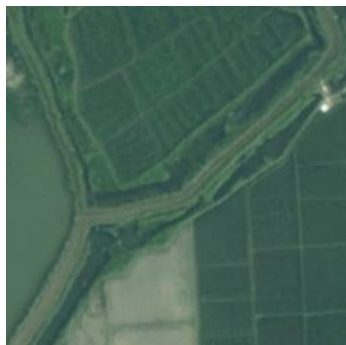

- 指标平均交并比 $MIoU$, 记真实值为 i , 预测为 p_{ij} 的像素数量为 :

- $$MIoU = \frac{1}{K+1} \sum_{i=0}^k \frac{p_{ii}}{\sum_{j=0}^k p_{ij} + \sum_{j=0}^k p_{ji} - p_{ii}}$$

2020 CCF BDCI : 遥感影像地块分割-初赛A榜

团队成员用户名	score	mIoU	提交状态	提交时间
Landoon	58.34668	0.58347	完成	2020-10-26 11:26

mIoU = 0.58374 为只训练了 1 个 EPOCH 的成绩。（后面的我忘记提交了）



A146006.jpg



A146006.png



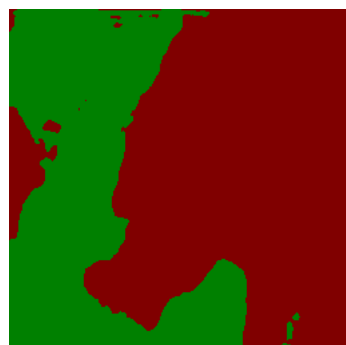
A146037.jpg



A146037.png



A146040.jpg



A146040.png



A146144.jpg



A146144.png

THANKS

兰冬雷

2020E8015382008