

# 基于 SAM 大模型的遥感影像分析

## 一、实验背景

在现代军事行动中，高效、精确的情报获取是制胜的关键。遥感和无人机影像作为战场侦察和目标监控的重要手段，为指挥官提供了高分辨率的情报数据。然而，大量的影像数据需要高效地进行分析和分类，以实现战场实时感知。为了应对这一挑战，本实验旨在采用先进的 SAM（Segment Anything Model）算法，对遥感或无人机影像进行精确分割与分类，并实现对目标数目的自动统计，从而提升战场信息的获取速度与精度。

## 二、实验目的

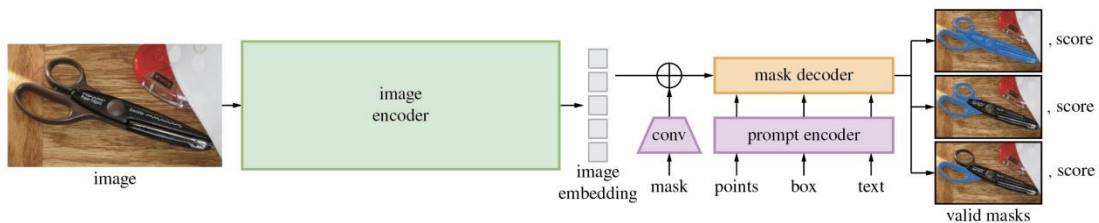
- 1、掌握 SAM 分割算法的基本原理及其在影像分类中的应用；
- 2、提高对遥感和无人机影像的解读与分析能力，能够精准锁定军事目标；
- 3、强化对先进信息技术在军事侦察与情报处理中的实际应用技能。

## 三、实验原理

### 3.1、SAM 模型的工作原理

Segment Anything Model (SAM) 是一种基于视觉变换器（Vision Transformer, ViT）的深度学习模型。其主要原理包括以下几个方面：

- （1）视觉特征提取：SAM 模型通过预训练好的 ViT 对影像中的像素信息进行深度提取。该过程能够生成影像的高维度特征表达，确保对复杂环境和目标细节有足够的敏感度。
- （2）注意力机制分割：SAM 利用多头注意力机制（Multi-head Self-Attention, MHSA）将每个像素点与其他像素点关联起来，实现对影像不同区域的分割。该机制能够针对影像中的军事目标和地理特征进行精确的区域识别。
- （3）前景目标标注：SAM 模型根据特定的分割策略对影像进行前景目标标注，通过对军事目标（如车辆、建筑物、地形特征）的高效分割，实现精准分类。



SAM 分割的演示 demo: <https://segment-anything.com/demo#>

SAM 项目链接: <https://github.com/facebookresearch/segment-anything>

### 3.2、遥感图像分类的工作原理

在遥感图像分类中，使用机器学习方法来实现分类。首先，从遥感图像中提取特征，例如颜色直方图、纹理特征等，作为每个像素的描述。在此过程中，通过解析 VOC 格式（或者其他格式）的标注文件，可以为图像中的不同区域分配正确的标签。随后，使用这些特征和标签来训练模型（例如随机森林、支持向量机等）。

## 四、实验任务说明

### 1、实验内容

- （1）参考提供的代码框架，学习如何调用 SAM 的预训练模型对遥感图像进行分割（根据个人电脑情况，选择 gpu 或者 cpu）；

(2) 基于 SAM 模型的分割结果, 设计一个 ui, 来实现对遥感图像的标注 (标注出每一块 sam 分割得到的区域是什么类型, 比如建筑、稻田、广场等), 尽量合理设计 ui 减少人工工作量;

(3) 基于上一步标注的数据集, 使用机器学习方法训练遥感影像分割后每一块区域的分类标注, 能够用不同颜色的 mask 表示不同区域。将自己标注的数据分为训练集和测试集, 在测试集计算分类的准确率 (Overall Accuracy, OA) 和 Kappa 系数;

(4) 在分类基础上, 实现每个类别计数。这一步可以使用图像形态学中的连通区域统计方法或者其他方法。

(5) 开放题: 尝试将遥感图像进行放大 (面积放大到 4 倍、9 倍, 即将原图切分为 4 块、9 块) 再进行上述的分割、分类、统计流程, 分析下方法能否鲁棒, 如果不能, 利用数据增强等方式来提升算法对输入尺度的鲁棒性。

## 2、实验数据

```
--asset
--- sam_vit_b_01ec64.pth (默认使用的 SAM 模型)

--dataset
----example (任务 1-调用 SAM 进行自动化分割的测试图)
----image (提供的用于半自动化标注数据)
----label (自行标注的数据结果)

--utils
----tif_processing.py
----others.py

--class_reference.py (用于分类的参考代码)
```

其中, 实验的遥感数据来源于 <https://zenodo.org/records/5706578> 中的部分。如需扩充, 可自行选择。

## 五、实验步骤

### 3、学习如何调用 SAM 分割模型进行单图像的分割

直接加载 sam\_vit\_b\_01ec64.pth 的预训练模型进行分割。设置可以选择自己电脑的 GPU 或者 CPU。根据自己电脑配置自由选择

另外, 对于 gpu 显存不足的同学, 可以尝试修改 sam\_kwargs 的参数 (可能导致分割效果下降)。另外也可以探索减少输入图像分辨率等方法降低显存消耗等等。参考代码:

```
def sam_segmentation(image_path, segmentation_path, str_device):
    checkpoint = r'asset\sam_vit_b_01ec64.pth'
    if str_device == "GPU":
        device = 'cuda' if torch.cuda.is_available() else 'cpu'
    else:
        device = 'cpu'
    sam_kwargs = {
        "points_per_side": 16, # 降低密度
        "crop_n_layers": 0, # 关闭多层裁剪
        "crop_overlap_ratio": 0.2, # 降低裁剪区域重叠}
```

```

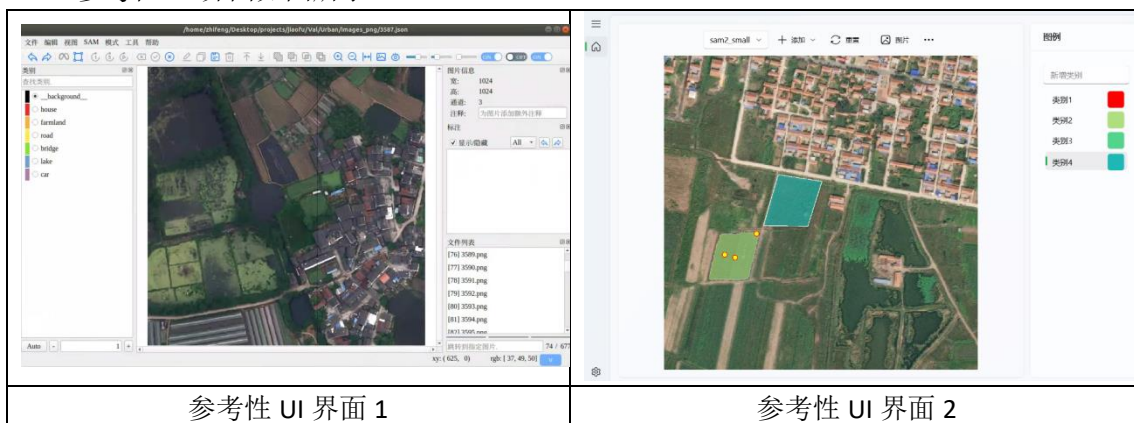
sam = SamGeo(
    checkpoint=checkpoint,
    model_type='vit_b',
    device=device,
    # sam_kwargs=None,
    sam_kwargs=sam_kwargs,)
with torch.no_grad():
    sam.generate(image_path, segmentation_path)

```

#### 4、基于 SAM 完成遥感数据的半自动化标注

调用 SAM 模型写一个半自动化标注 UI 界面，进行辅助性标注。标注数据的存储格式自由选择，可以存储为 json 格式或者 txt 格式或者遥感图像的.shp 格式。需要保证后续分类读取能顺利读取该标注格式即可。

参考性 UI 界面如图所示：



存储数据 json 格式参考：

```

1  {
2      "info": {
3          "description": "ISAT",
4          "folder": "/home/zhifeng/Desktop/projects/jiaofu/ISAT_with_segment_anything-master/example/images",
5          "name": "00000000113.jpg",
6          "width": 416,
7          "height": 640,
8          "depth": 3,
9      },
10     "objects": [
11         {
12             "category": "house",
13             "group": 1,
14             "segmentation": [
15                 [
16                     36.0,
17                     514.0
18                 ],
19                 [
20                     35.0,
21                     516.0
22                 ],
23                 [
24                     34.0,
25                     521.0
26                 ],
27             ]
14

```

#### 5、分类和计数

采用随机森林的方法进行分类，并对分类结果 `predicted` 里面的类别数和每个类别里面的个数进行统计。参考代码：

```

def main_classification(img_path, segmentation_path, samples_path, classification_path,
train_radio, classification_method):
    #-----read file-----
    print("Starting classification")
    im_data, im_geotrans, im_proj, im_width, im_height = tif_processing.read_tif(img_path)
    segmentation, _, _, _ = tif_processing.read_tif(segmentation_path)

    im_data = im_data[0:3]
    temp = im_data.transpose((2, 1, 0))
    segmentation = segmentation.transpose((1, 0))
    #-----split train,test-----
    # Need to fill in the code
    #-----build feature-----
    segment_ids = np.unique(segmentation)
    objects = []
    object_ids = []
    for i in segment_ids:
        segment_pixels = temp[segmentation == i]
        object_features = other.segment_features(segment_pixels)
        objects.append(object_features)
        object_ids.append(i) # 修复 ID 添加
    #-----get train/test objects using train/test samples-----
    # Need to fill in the code
    #-----classify-----
    if classification_method == "DT":
        classifier = DecisionTreeClassifier(random_state=0)
        classifier.fit(training_objects, training_labels)
        predicted = classifier.predict(objects)
        # 打印类别数和统计信息
        # Need to fill in the code
    #-----output-----
    clf = segmentation.copy()
    for segment_id, klass in zip(segment_ids, predicted):
        clf[clf == segment_id] = klass
    mask = np.sum(temp, axis=2)
    mask[mask > 0.0] = 1.0
    mask[mask == 0.0] = -1.0
    clf = np.multiply(clf, mask)
    clf[clf < 0] = 0
    clf[clf > 255] = 255
    clf = clf.transpose((1, 0))
    driverTiff = gdal.GetDriverByName("GTiff")
    clfds = driverTiff.Create(classification_path, im_width, im_height, 1, gdal.GDT_Byte)
    clfds.SetGeoTransform(im_geotrans)

```




```

clfds.SetProjection(im_proj)
clfds.GetRasterBand(1).SetNoDataValue(255)
clfds.GetRasterBand(1).WriteArray(clf)
clfds = None
# 准确度评估
# Need to fill in the code
return {
    # Need to fill in the code
}

```

## 六、实验效果

需要输出分类的类别数和每个类别下的物体个数。以及分类的准确率等。

输入图	分割图	分类图
		

## 七、得分点

- 1、学习如何调用 SAM 分割模型，用于图片的自动化分割（10 分）；
- 2、基于 SAM 模型，写一个半自动化的数据标注工具。要求，在该 UI 界面上，可以通过点击物体就能通过 SAM 模型框选整个物体，并标注上相应的类别（25 分）；
- 3、基于各自标注的数据。采用随机森林等方法训练一个分类算法，用于遥感影像的分类训练与测试（30 分）；
- 4、在分类基础上完成分类后的类别计数（15 分）。

开放题（20 分）：

尝试将遥感图像进行放大或者缩小（面积放大到 4 倍、9 倍，即将原图切分为 4 块、9 块，缩小面积，可以复制原图 4 次再缩放成原图大小，即缩小面积 4 倍）再进行上述的分割、分类、统计流程，分析下方法能否鲁棒，如果不能，利用数据增强等方式来提升算法对输入尺度的鲁棒性。

（如优化标注算法，不能简单换用其他的预训练模型，这种提升效果的方式将不给分，建议利用数据增强等方式，增强数据集进行分类训练）。