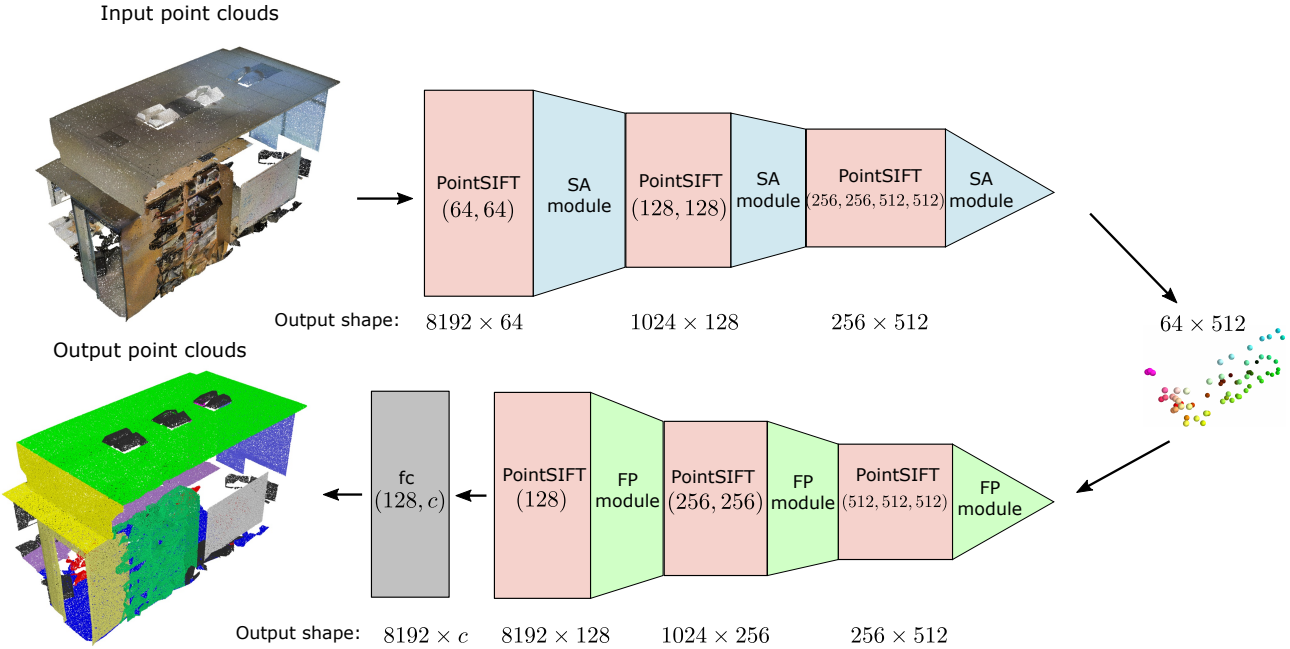
本赛题代码由相同方向的论文PointSIFT项目借鉴修改而成，该论文源代码可参考<https://github.com/MVIG-SJTU/pointSIFT>（PointSIFT: A SIFT-like Network Module for 3D Point Cloud Semantic SegmentationCreated by Mingyang Jiang, Yiran Wu, Cewu Lu (corresponding author).）

原论文数据集是室内场景scannet <https://arxiv.org/pdf/1702.04405.pdf>

Pointsift源代码运行流程可参考<https://blog.csdn.net/Felaim/article/details/81088936>



本次赛题针对数据集的差异，做了一些修改，使之适应比赛数据。下面叙述一下给方面的具体细节：

1. **运行环境：**Tensorflow , CUDA , python的 numpy等工具包
2. **运行流程：**第一次需要编译cuda代码 tf\_sampling\_compile.sh(程序文件中已编译好)

进行训练时如果有多个 GPU可以用如下命令

**CUDA\_VISIBLE\_DEVICES=0,1,2,3 python train\_and\_eval\_scannet.py --gpu\_num=4 --batch\_size 16 --data\_path ./**

训练好的结果(评测得分高的)模型会保存在model\_param文件夹中。

进行预测时(单GPU)使用如下命令

**CUDA\_VISIBLE\_DEVICES=0 python predict.py --gpu\_num=1 --batch\_size 8 --data\_path ./**

通过修改代码中结果最好一次结果的文件名称，读入模型并进行预测。结果输出在predictvalue文件夹中，每个场景输出一个文件(如x.npy)。

再执行 python submit.py 把预测出的值转换成比赛要求的文件格式和文件名。

1. **程序各文件解释**

**scannet\_dataset.py** 训练集、测试集、预测集数据的预处理

**predict.py** 加载训练好的模型和参数，读入预测数据集，进行预测后输出每个场景的预测值

**train\_and\_eval\_scannet.py** 数据、模型的加载 训练参数，流程控制

**submit.py** 把预测后的每个场景的值根据比赛格式要求转换成 .csv文件

**filenames.npy** 预测数据集中每个场景对应的文件名，用来给输出结果重命名，一维

**filenums.npy** 预测数据集中每个场景中点的个数，用来截断输出结果符合实际点的数量，一维

**predictvalue文件夹** 存放预测过程中生成的第n个场景的预测结果，如 123.npy

**predictdata文件夹** 存放需要被预测的数据集，如scannet\_predict2.pickle

**models文件夹** pointSIFT\_pointnet.py用来搭建模型

**model\_param文件夹** 存放训练过程中当前评估结果最好的模型

**submit文件夹**  预测后的值按照比赛要求的格式修改成xxx.csv存放在这

**tf\_utils文件夹** 包含模型的各个基本模块

**log文件夹** 存放训练过程中的信息，可以用tensorboard查看训练的阶段状态

**csvToPickle.py predictCsvToPickle.py transer.py** 为比赛数据预处理，统一格式为pickle

1. **数据预处理和细节说明**

训练时只用到了前两万个场景，并把源csv格式数据转换成pickle格式文件，再通过修改训练集的场景大小，缩小x,y,z范围，去除坐标原点周围的点，然后再随机抽取出8192个点放到网络中训练。模型中的radius值从原先的0.1等根据scannet室内和比赛场景的不同，统一放大到原来的8倍。预测时也同样缩小了预测范围，为了不打乱点的顺序，不在范围内的点的坐标值会被修改到原点，最后网络的输出即为当前场景的预测值。根据每个场景的点数量不同，和模型的输入大小，统一成8192个点输入，每个场景会被按照文件xyz中的顺序分成8份，每份8192个点，不足的部分通过在后部分填充点(0,0,0)来满足。预测集因为文件大小缘故被平分成7份分别进行。