1. 对预处理进行优化

预处理阶段有两个阶段，分别是取值操作和标准化操作。

* 1. 取值操作(1-9)

对于高光谱图像重构，我们采用方法是9个像素值重构成一条光谱曲线，而在实际的高光谱拍摄过程中，输入进重构网络中的是一个光强矩阵(如1024\*1024)。为了不损失高光谱图像重构中太多的像素值，采用一个类似3\*3大小，步长为1的卷积核，不过操作不是相乘，而是直接取值，把光强值矩阵变成输入矩阵(1024\*1024\*9)。

采用的优化方法是使用Pycuda库，把取值操作分配到每个线程中，每个线程取9个值；又根据取值操作的规律，可以找到一组对应的映射规律：

当x, y = 0时，基于当前线程坐标加值顺序为(0,1,2)

当x, y = 1时，基于当前线程坐标加值顺序为(2,0,1)

当x, y = 2时，基于当前线程坐标加值顺序为(1,2,0)

其中，x, y的值为当前线程坐标对3取余，当前线程坐标记为(tx,ty)，即x, y = (tx, ty) % 3。

* 1. 标准化操作

对数据进行标准化，有利于提高算法精度和加速算法的收敛速度。



* 1. Cuda核函数实现取值操作(1-9)和标准化操作



* 1. 取值操作(1-16)

上述取值操作是1024\*1024 变成 1024\*1024\*9，此取值操作是1024\*1024 变成 1024\*1024\*16。映射关系和上述取值操作(1-9)原理一样，映射关系为：

当x, y = 0时，基于当前线程坐标加值顺序为(0,1,2,3)

当x, y = 1时，基于当前线程坐标加值顺序为(3,0,1,2)

当x, y = 2时，基于当前线程坐标加值顺序为(2,3,0,1)

当x, y = 3时，基于当前线程坐标加值顺序为(1,2,3,0)

其中，x, y的值为当前线程坐标对4取余，当前线程坐标记为(tx,ty)，即x, y = (tx, ty) % 4。

1. 对神经网络推理优化

2.1 TensorRT技术

TensorRT是一个SDK，用于优化经过训练的深度学习模型，以实现高性能推理。 TensorRT包含一个用于训练的深度学习模型的深度学习推理优化器，以及一个用于执行的运行时。在Pytorch框架中训练高光谱重构模型之后，使用TensorRT以更高的吞吐量和更低的延迟运行重构模型。

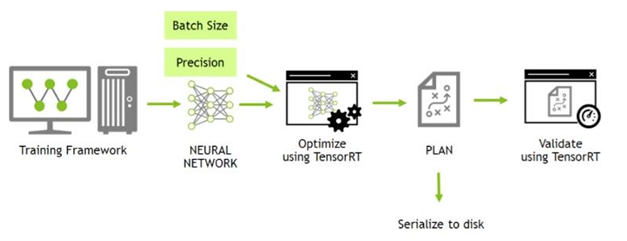
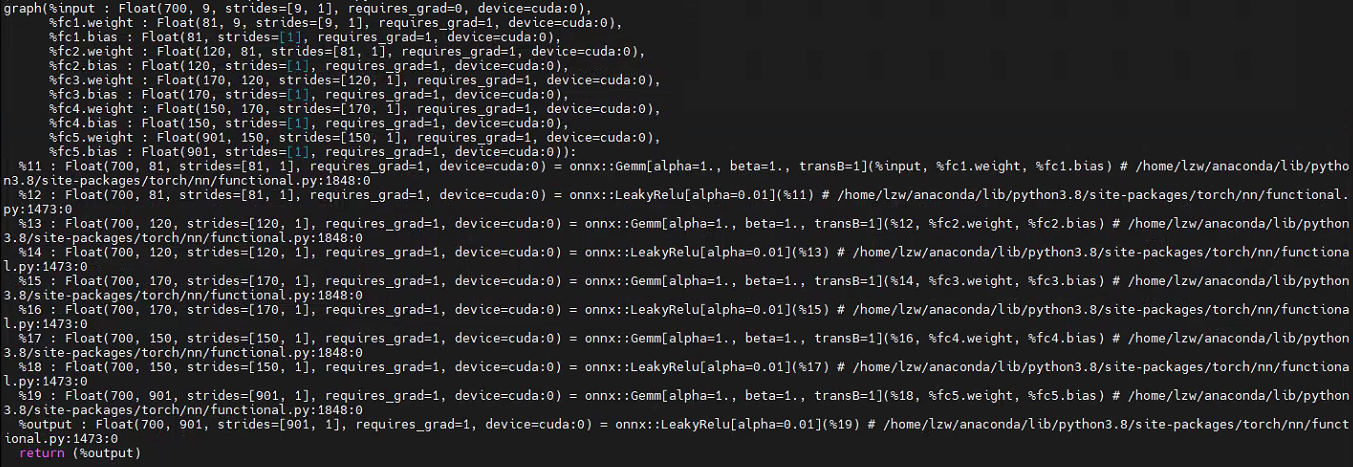


图1：使用TensorRT的典型深度学习开发周期

2.2 对神经网络推理优化加速步骤

第一步，把训练好的神经网络模型转化为.onnx

使用Pytorch中的torch.onnx.export，把神经网络模型导出成为onnx格式，如下图为导出输入大小为(700,9)的DNN的graph。



第二步，调用tensorRt的API，把.onnx转化为.trt



第三步，使用tensorRt进推理加速：

1. 获取engine



1. 设置GPU中数据缓存地址、CUDA流



1. 进行tensorRt推理



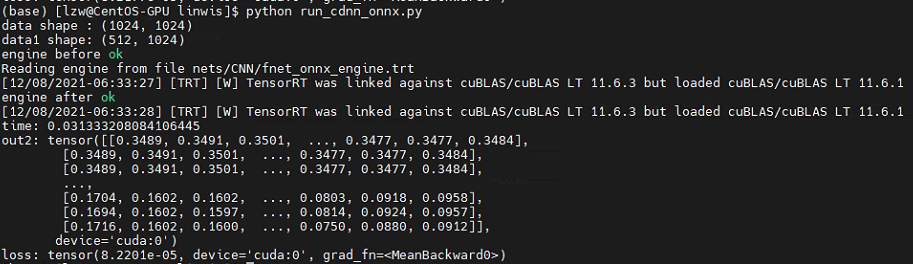
1. 对预处理和神经网络的推理结合优化

3.1 对DNN优化的结果

DNN的神经网络结构：



对于1024\*1024的光强矩阵，预处理加推理的时间为0.031s左右，与原模型的RMSE为8.2201e-05。原模型推理1024\*1024需要0.065S左右，且不包括预处理时间。



3.2 对CNN优化的结果

CNN神经网络结构：



由于我们目前的卷积重构网络层数过多，参数过于庞大，因此无法推理1024\*1024的光强矩阵。卷积重构网络用pytorch在GPU上推理的最大高光谱图片为900\*900，时间为1.25s左右。用tensorrt推理的最大高光谱图片为700\*700，时间为0.17s左右，同数据规模的pytorch推理需要0.77s，上述实验都是在v100 32G环境下测试的。

