APSS Training Guide

指导用户如何构建APSS项目的训练过程。

本项目支持的后端设备为：

- Ascend(Help)
- GPU
- CPU

本指导文件以GPU环境为例。

目录

- 项目清单
- 环境构建
    - [Method 1: With Mindspore's Official Image and Build from Source](#Method 1: With Mindspore's Official Image and Build from Source)
    - [Method 2: With Our Docker Image](#Method 2: With Our Docker Image)
- 程序运行
- 训练原理

项目清单

- 源代码 (https://github.com/Cheny1m/APSS)
    - apss为项目源代码部分，inference表示推理部分代码，nets表示模型，problems表示我们抽象出来的拟训练问题，training表示训练部分代码，utils是一些工具类。
    - apss.egg-info为打包后测试使用pip安装apss包后的元数据信息。
    - docs是一些说明文档
    - resource是数据包链接
    - scripts是一些自动化处理的脚本
    - config.json包含一些全局的配置。
    - dockerfile为训练环境镜像构建文件。
    - pyproject.toml为配置依赖启动文件。
- 数据包
    - 单独开辟了占用空间较大的数据存储，并在构建程序运行环境时，分别将源代码文件和数据包内容同时映射或放入运行环境中，数据包由代码文件中的/resource目录进行映射。即 APSS/resource -->data
- Docker环境镜像 (https://hub.docker.com/repository/docker/cheny1m/apss-mindspore-gpu-cuda11.1/general)
    - 使用Mindspore官方镜像后，使用pip进行安装。
    - 使用dockerfile构建我们已经打包好的容器或者从docker hub上拉取。

环境构建(GPU)

Requirements:

- Python >= 3.7
- Mindspore >= 2.2.0 (Help)

Method 1: With Mindspore's Official Image and Build from Source

启动容器：将源代码目录APSS（本例中为/home/upa1/cym/MindSpore/APSS）和数据包目录data（本例中为/home/upa1/cym/MindSpore/data）分别映射到容器内部的APSS目录（本例中为/root/APSS）和APSS/resource目录（本例中为/root/APSS/resource） ***注意：如果数据包的容器映射目录不为默认的resource，请在config.json中修改RESOURCE_DIR的value为您定义的目录。***

```
docker run -itd -v /dec/shm:/dev/shm -v /home/upa1/cym/MindSpore/APSS:/root/APSS -v
/home/upa1/cym/MindSpore/data:/root/APSS/resource --name apss --runtime=nvidia swr.cn-
south-1.myhuaweicloud.com/mindspore/mindspore-gpu-cuda11.1:2.2.0 /bin/bash

docker exec -it apss /bin/bash
```

从源码构建：

```
cd ~/APSS
pip install -e .
```

Method 2: With Our Docker Image

[可选1]拉取镜像

```
docker push cheny1m/apss-mindspore-gpu-cuda11.1:1.0
```

[可选2]或者通过dockerfile构建镜像

```
docker build -t apss-mindspore-gpu-cuda11.1:1.0 .
```

获得镜像后启动容器：

```
docker run -itd -v /dev/shm:/dev/shm -v /home/upa1/cym/MindSpore/APSS:/root/APSS -v
/home/upa1/cym/MindSpore/data:/root/APSS/resource --name apss --runtime=nvidia
cheny1m/apss-mindspore-gpu-cuda11.1:1.0 /bin/bash

docker exec -it apss /bin/bash
cd ~/APSS
```

环境构建(Ascend)

Requirements:

- Python >= 3.7
- Mindspore >= 2.2.0 (Help)

164:source /home/kkr/env.sh

程序运行

设置运行环境的context

本步骤主要设置运行时的目标设备和模式，默认目标设备为GPU，默认运行模式为PYNATIVE_MODE。如需查看详情和修改目标设备及运行模式，请在config.json中修改。

- "DEVICE_TARGET": 设置运行设备。支持[Ascend],[GPU],[CPU].
- "CONTEXT_MODE": 设置运行环境context的mode,在[0]：(GRAPH_MODE)和[1]：(PYNATIVE_MODE)中选择。

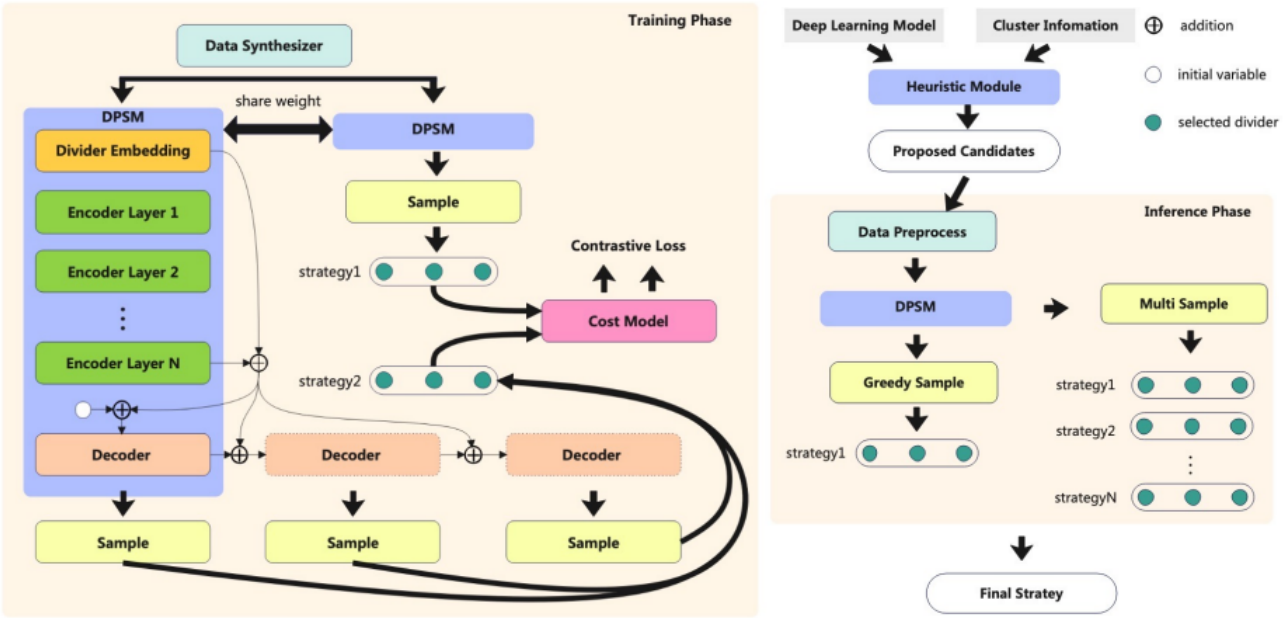
一步执行训练

```
python -m apss.training.apss_run --graph_size 8 --num_split 3 --rebuild_data
```

- graph_size , num_split 分别代表了问题的层数大小和需要执行pipeline划分的数量，两个命令行参数共同描述了所训练问题的大小，可根据需求动态调整。目前graph_size取值范围为[8,18,25,30,42,54,102]，num_split取值范围为[1,3,7,15,31,63]。
- rebuild_data 表示是否在执行训练前，从Data Synthesizer中生成训练数据，默认建议开启。如果需要从.ckpt中接续训练或无需改变之前生成的训练数据直接禁用--rebuild_data参数即可。生成的训练数据可在数据包的/data目录下找到。
- 已经完成过执行训练后，本次运行的参数文件及.ckpt文件将保存在数据包的/output文件夹下，日志保存在数据包的/log文件夹下，可以通过tensorboard_logger在浏览器中实时查看训练过程及其数据。

执行上述代码会执行apss的训练，所有在num_split取值范围中且小于设定的num_split的模型都将被训练。每个模型训练默认训练100个epoch，每个epoch训练1,280,000条数据，batch_size为512。

训练原理


Training Phase
Data Synthesizer
share weight
DPSM
Divider Embedding
Encoder Layer 1
Encoder Layer 2
Encoder Layer N
Decoder
Decoder
Decoder
DPSM
Sample
strategy1
Contrastive Loss
Cost Model
strategy2
Sample
Sample
Sample
Deep Learning Model
Cluster Infomation
addition
initial variable
selected divider
Heuristic Module
Proposed Candidates
Inference Phase
Data Preprocess
DPSM
Multi Sample
Greedy Sample
strategy1
strategy1
strategy2
strategyN
Final Stratey