SPLIT pipe

Documentation of SplitPipe:

models/

This directory contains the model implementation. Each sub-directory of models/ correspond a model's implementation.

To add a new model, you should at least have the following API for splitPipe:

vector<sequential> model_split(vector split_points, …)
vector<sequential> model_part(int start, int end)

The first function defines the layers of the model and returns a vector of sequential object with consecutive atomic unit-block. With the split_points vector one can define splits, in this case each index of the vector is one model part defined by the split_points. If split_points empty then the model is split in each atomic unit block.
The function model_part returns only the part that belongs to start until the end atomic unit block.

models/vgg/

VGG models (VGG11, VG13, VGG16, VGG19) following the implementation from: PyTorch: Source Code For torchvision.models.VGG

To select the VGG model for profiling, use function:
train_vgg(dataset dataset_option, vgg_model model_option, bool split, vector<int> split_points)

- dataset_option: select dataset
- model_option: value from vgg::vgg_model enum to select the vgg model: v11, v11_bn, v13, v13_bn, v16, v16_bn, v19, v19_bn.
- split: *true* if you want to profile upon the split mode.
- split_points: optional parameter (no impact if split == *false*). It is the vector with cut layers.

models/resnet/

ResNet models (resnet18, resnet34, resnet50, resnet101, resenet152) following the implementation from: Github: ResNet_PyTorch.ipynb.

To select the ResNet model for profiling, use function:
train_resnet(dataset dataset_option, resnet_model model_option, bool split, int batch_size = 64, const std::vector<int>& split_points, bool test)

- dataset_option: select dataset

- model_option: value from resnet_model enum to select the ResNet model: resnet18, resnet34, resnet50, resnet101, resenet152.
- split: *true* if you want to profile upon the split mode.
- split_points: optional parameter (no impact if split == *false*). It is the vector with cut layers.
- test: this is a flag, when it is true it will fully train the model otherwise will return profiling results.

main.cpp

Main code that is used to profile the model. It contains two modes:

(i) conventional training: in which we run the model as one (split flag equal to false),

(ii) Split Learning: the model is split into parts and is trained in a way to simulate a split learning environment. We use this mode for a per-layer analysis of the model (split flag equal to true).

utils/

pipeline_simulation/

```
pipeline_simulation/
        aggregator.cpp
        compute_node.cpp
        data_owner.cpp
        Message.h
        network_layer.cpp
        network_layer.h
        State.h
        systemAPI.cpp
        systemAPI.h
        State.h
        Task.h
        /profiling
                data_owner_simulated.cpp
                rpi_stats.h
```