**Swish Plugin Documentation**

**Description:**

We implemented the SWISH activation function which is currently not supported in the TensorRT.

Swish is a self-gated activation function discovered by researchers at Google. According to their [paper](https://arxiv.org/abs/1710.05941v1), it performs better than ReLU with a similar level of computational efficiency. The activation function is shown in Figure 1.

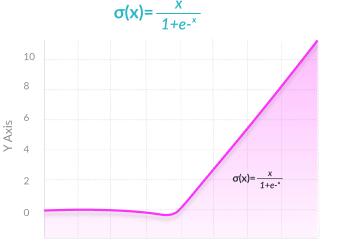


Figure 1. SWISH activation function

**Directory Details:**

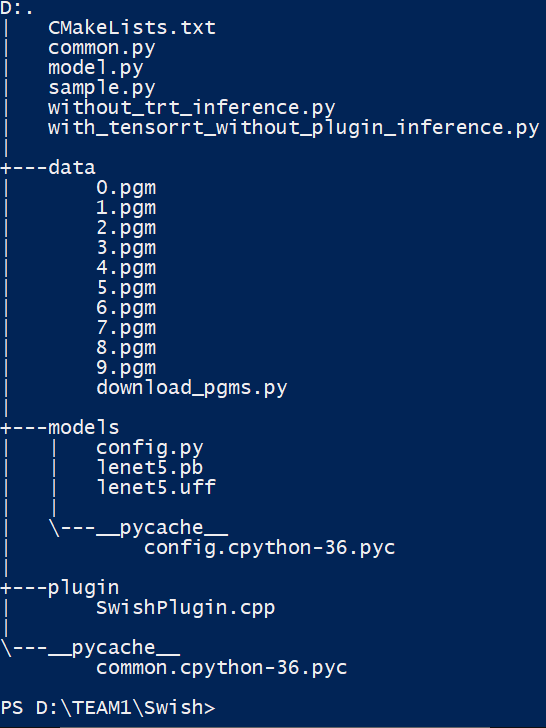
We used NATIVE way to implement the SWISH activation functional in the TensorRT. We worked with different files, the details of which are listed below:

* **SwishPlugin.cpp** - We implemented the swish activation function in enqueue() function. We
  + looped over the total elements (in code, it is mTotalElements),
  + performed simple arithmetic operations,
  + changed value in the output pointer,
  + and increased input and output pointers by 1.

We are assuming that activation functions apply to single layer (Example- model.add(Dense(64, activation='tanh')) ) and therefore, we don’t require outer loop for each input.

* **CMakeLists.txt -** We use this file to build our plugin. **Inference files -** We have included multiple inference files (“sample.py”, “without\_trt\_inference.py”, “with\_tensorrt\_without\_plugin\_inference.py”) for inference with Plugin, without TensorRT, with TensorRT but without plugin respectively.
* **Model.py -** We have edited a Lenet5 model by including Swish activation in it. The model is present in the ‘./models’ directory.
* **Config.py -** This file uses graphsurgeon to replace the nodes which are part of the ‘activation’ node to be replaced by TensorRT node.
* **Dataset -** We have used the MNIST dataset to train and test the model. The images are present in the data directory.

**Note:** Instructions on how to use the above files are provided in the next section.



**Steps:**

The following instructions cover all the steps required to run the inference files successfully:

* Move inside Swish directory - “cd ./Swish”
* Train and save the edited lenet5 model using - “python model.py”
* Move inside models directory - “cd ./models”
* Convert pb file to uff - “convert-to-uff –p ./config.py”
* Move back to Swish directory - “cd ..”
* Building Swish plugin:
  + “sh
  + mkdir –p build
  + cd build
  + cmake ..
  + make
  + exit"
* Now we can run any provided inference files to test our plugin:
  + “python name\_of\_inference\_file\_given\_in\_previous\_section.py”

**Prerequisites:**

In order to test our implemented plugin, we use following TensorRT container on NVIDIA DGX-1 V100 supercomputer:

* TensorRT container used: **nvcr.io/nvidia/tensorrt 19.12-py3**

**Results:**

We test out model with and without using the custom created SWISH plugin. Below are the results per test example in seconds(s):

**Result without using TensorRT:**



**Result without using the custom created plugin:**



**Result with custom created plugin:**



**NOTE:** It can be noted from the above screenshot that the inference speed increases with custom created Swish plugin marginally as Swish is relatively a simple function to compute and lenet5 is a simple model with activation function used only once.