Unary 层

- 初始示例代码
- op

初始示例代码

```
import numpy as np
from cuda import cudart
import tensorrt as trt

nIn, cIn, hIn, wIn = 1, 1, 3, 3  # 输入张量 NCHW
data = np.arange(-4, 5, dtype=np.float32).reshape(nIn, cIn, hIn, wIn)  # 输入数据

np.set_printoptions(precision=8, linewidth=200, suppress=True)
cudart.cudaDeviceSynchronize()

logger = trt.Logger(trt.Logger.ERROR)
builder = trt.Builder(logger)
network = builder.create_network(1 << int(trt.NetworkDefinitionCreationFlag.EXPLICIT_BATCH))
config = builder.create_builder_config()
config.max_workspace_size = 1 << 30
inputT0 = network.add_input('inputT0', trt.DataType.FLOAT, (nIn, cIn, hIn, wIn))  # 单输入示例代码
#------------------------------------------------------------ --------------------# 替换部分
unaryLayer = network.add_unary(inputT0, trt.UnaryOperation.ABS)
#------------------------------------------------------------ --------------------# 替换部分
network.mark_output(unaryLayer.get_output(0))
engineString = builder.build_serialized_network(network, config)
engine = trt.Runtime(logger).deserialize_cuda_engine(engineString)
context = engine.create_execution_context()
_, stream = cudart.cudaStreamCreate()

inputH0 = np.ascontiguousarray(data.reshape(-1))
outputH0 = np.empty(context.get_binding_shape(1), dtype=trt.nptype(engine.get_binding_dtype(1)))
_, inputD0 = cudart.cudaMallocAsync(inputH0.nbytes, stream)
_, outputD0 = cudart.cudaMallocAsync(outputH0.nbytes, stream)

cudart.cudaMemcpyAsync(inputD0, inputH0.ctypes.data, inputH0.nbytes,
cudart.cudaMemcpyKind.cudaMemcpyHostToDevice, stream)
context.execute_async_v2([int(inputD0), int(outputD0)], stream)
cudart.cudaMemcpyAsync(outputH0.ctypes.data, outputD0, outputH0.nbytes,
cudart.cudaMemcpyKind.cudaMemcpyDeviceToHost, stream)
cudart.cudaStreamSynchronize(stream)

print("inputH0 :", data.shape)
print(data)
print("outputH0:", outputH0.shape)
print(outputH0)

cudart.cudaStreamDestroy(stream)
cudart.cudaFree(inputD0)
cudart.cudaFree(outputD0)
```

- 输入张量形状 (1,1,3,3)

$$\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}-4. & -3. & -2.\\ -1. & 0. & 1.\\ 2. & 3. & 4.\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

- 输出张量形状 (1,1, 3, 3)

$$\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}4. & 3. & 2.\\ 1. & 0. & 1.\\ 2. & 3. & 4.\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

---

op

```
unaryLayer = network.add_unary(inputT0, trt.UnaryOperation.NEG)
unaryLayer.op = trt.UnaryOperation.ABS                                    # 重
设使用的一元函数
```

- 输出张量形状 (1,1,3,3)，结果与初始示例代码相同
- 可用的一元函数

trt.UnaryOperation 名	函数
NEG	$-x$
NOT	$not\,(x)$
ABS	$\lvert x\rvert$
FLOOR	$\lfloor x\rfloor$
CEIL	$\lceil x\rceil$
RECIP	$\frac{1}{x}$
SQRT	$\sqrt{x}$
EXP	$\exp(x)$
LOG	$\log(x)$（以 e 为底）
ERF	$erf\,(x)=\int_0^x \exp\left(-t^2\right)dt$
SIN	$\sin(x)$
COS	$\cos(x)$
TAN	$\tan(x)$
ASIN	$\sin^{-1}(x)$
ACOS	$\cos^{-1}(x)$
ATAN	$\tan^{-1}(x)$
SINH	$\sinh(x)$
COSH	$\cosh(x)$
没有TANH	tanh 作为 activation 层的函数
ASINH	$\sinh^{-1}(x)$
ACOSH	$\cosh^{-1}(x)$
ATANH	$\tanh^{-1}(x)$