Quantize + Dequantize 层

- 初始示例代码
- axis
- set_input 与 zeroPt

初始示例代码

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

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

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.flags = 1 << int(trt.BuilderFlag.INT8)  # 需要打开 int8 模式
config.max_workspace_size = 1 << 30
inputT0 = network.add_input('inputT0', trt.DataType.FLOAT, (nIn, cIn, hIn, wIn))
#------------------------------------------------------------ --------------------# 替换部分
constantLayer0 = network.add_constant([], np.array([60 / 127], dtype=np.float32))  # 目前只支持 build 期常量
constantLayer1 = network.add_constant([], np.array([1], dtype=np.float32))

quantizeLayer = network.add_quantize(inputT0, constantLayer0.get_output(0))  # 目前只支持 float32 的量化
quantizeLayer.axis = 0  # 指定量化轴
dequantizeLayer = network.add_dequantize(quantizeLayer.get_output(0), constantLayer1.get_output(0))
dequantizeLayer.axis = 0
#------------------------------------------------------------ --------------------# 替换部分
network.mark_output(dequantizeLayer.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 (1,3,4,5)

$$\begin{bmatrix}\begin{bmatrix}\begin{bmatrix} 0. & 1. & 2. & 3. & 4. \\ 5. & 6. & 7. & 8. & 9. \\ 10. & 11. & 12. & 13. & 14. \\ 15. & 16. & 17. & 18. & 19. \end{bmatrix} \begin{bmatrix} 20. & 21. & 22. & 23. & 24. \\ 25. & 26. & 27. & 28. & 29. \\ 30. & 31. & 32. & 33. & 34. \\ 35. & 36. & 37. & 38. & 39. \end{bmatrix} \begin{bmatrix} 40. & 41. & 42. & 43. & 44. \\ 45. & 46. & 47. & 48. & 49. \\ 50. & 51. & 52. & 53. & 54. \\ 55. & 56. & 57. & 58. & 59. \end{bmatrix}\end{bmatrix}\end{bmatrix}$$

- 输出张量形状 (1,3,4,5)

$$\begin{bmatrix}\begin{bmatrix}\begin{bmatrix} 0. & 2. & 4. & 6. & 8. \\ 11. & 13. & 15. & 17. & 19. \\ 21. & 23. & 25. & 28. & 30. \\ 32. & 34. & 36. & 38. & 40. \end{bmatrix} \begin{bmatrix} 42. & 44. & 47. & 49. & 51. \\ 53. & 55. & 57. & 59. & 61. \\ 63. & 66. & 68. & 70. & 72. \\ 74. & 76. & 78. & 80. & 83. \end{bmatrix} \begin{bmatrix} 85. & 87. & 89. & 91. & 93. \\ 95. & 97. & 99. & 102. & 104. \\ 106. & 108. & 110. & 112. & 114. \\ 116. & 119. & 121. & 123. & 125. \end{bmatrix}\end{bmatrix}\end{bmatrix}$$

- 计算过程:

$$\begin{aligned} Quantize: output &= \mathbf{clamp}\left(\mathbf{round}\left(\frac{input}{scale}\right) + zeroPt\right) \\ &= \mathbf{clamp}\left(\mathbf{round}\left([0.,1.,2.,\ldots,59.]/\frac{60}{127}\right) + 0\right) \\ &= \mathbf{clamp}\left([0,2,4,\ldots,125] + 0\right) \\ &= [0,2,4,\ldots,125] \\ Dequantize: output &= (input - zeroPt) * scale \\ &= ([0,2,4,\ldots,125] - 0) * 1. \\ &= [0.,2.,4.,\ldots,125.] \end{aligned}$$

- 必须指定量化轴，否则报错:

```
[TensorRT] ERROR: 2: [scaleNode.cpp::getChannelAxis::20] Error Code 2: Internal Error ((Unnamed Layer*
2) [Quantize]: unexpected negative axis)
[TensorRT] ERROR: 2: [scaleNode.cpp::getChannelAxis::20] Error Code 2: Internal Error ((Unnamed Layer*
3) [Dequantize]: unexpected negative axis)
```

---

axis

```
constantLayer0 = network.add_constant([3], np.array([60 / 127, 120 / 127, 240 / 127], dtype=np.float32))
constantLayer1 = network.add_constant([], np.array([1], dtype=np.float32))

quantizeLayer = network.add_quantize(inputT0, constantLayer0.get_output(0))
quantizeLayer.axis = 1
dequantizeLayer = network.add_dequantize(quantizeLayer.get_output(0), constantLayer1.get_output(0))
dequantizeLayer.axis = 0
```

- 输出张量形状 (1,3,4,5)，三个通道分别把 [0,60]，[0,120]，[0,240] 映射为 [0,127]（分别大约是除以二、不变、乘以二）

$$\left[\left[\left[\begin{matrix}0. & 2. & 4. & 6. & 8.\\ 11. & 13. & 15. & 17. & 19.\\ 21. & 23. & 25. & 28. & 30.\\ 32. & 34. & 36. & 38. & 40.\end{matrix}\right]\left[\begin{matrix}21. & 22. & 23. & 24. & 25.\\ 26. & 28. & 29. & 30. & 31.\\ 32. & 33. & 34. & 35. & 36.\\ 37. & 38. & 39. & 40. & 41.\end{matrix}\right]\left[\begin{matrix}21. & 22. & 22. & 23. & 23.\\ 24. & 24. & 25. & 25. & 26.\\ 26. & 27. & 28. & 28. & 29.\\ 29. & 30. & 30. & 31. & 31.\end{matrix}\right]\right]\right]$$

---

set_input 与 zeroPt

```
constantLayer0 = network.add_constant([3],np.array([20/127,40/127,60/127],dtype=np.float32))
constantLayer1 = network.add_constant([],np.array([1],dtype=np.float32))
constantLayer2 = network.add_constant([3],np.array([-60,-96,-106],dtype=np.int32))
zeroPointLayer = network.add_identity(constantLayer2.get_output(0))
zeroPointLayer.get_output(0).dtype = trt.DataType.INT8

quantizeLayer = network.add_quantize(inputT0,constantLayer0.get_output(0))
quantizeLayer.axis = 1
quantizeLayer.set_input(0,inputT0)                                                  # 第
0 输入是被量化的张量
quantizeLayer.set_input(1,constantLayer0.get_output(0))                             # 第
1 输入是 scale 张量
quantizeLayer.set_input(2,zeroPointLayer.get_output(0))                             # 第
2 输入是 zeroPoint 张量（TensorRT<=8.2 尚不可用）
dequantizeLayer = network.add_dequantize(quantizeLayer.get_output(0),constantLayer1.get_output(0))
dequantizeLayer.axis = 0
```

- 输出张量形状 (1,3,4,5)

$$\left[\left[\left[\begin{matrix}0. & 6. & 13. & 19. & 25.\\ 32. & 38. & 44. & 51. & 57.\\ 64. & 70. & 76. & 83. & 89.\\ 95. & 102. & 108. & 114. & 121.\end{matrix}\right]\left[\begin{matrix}64. & 67. & 70. & 73. & 76.\\ 79. & 83. & 86. & 89. & 92.\\ 95. & 98. & 102. & 105. & 108.\\ 111. & 114. & 117. & 121. & 124.\end{matrix}\right]\left[\begin{matrix}85. & 87. & 89. & 91. & 93.\\ 95. & 97. & 99. & 102. & 104.\\ 106. & 108. & 110. & 112. & 114.\\ 116. & 119. & 121. & 123. & 125.\end{matrix}\right]\right]\right]$$