TopK 层

- 初始示例代码
- op
- k
- axes

初始示例代码

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

np.random.seed(97)
nIn, cIn, hIn, wIn = 1, 3, 4, 5  # 输入张量 NCHW
data = np.random.permutation(np.arange(nIn * cIn * hIn * wIn, 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))
#---------------------------------------------------------- --------------------# 替换部分
topKLayer = network.add_topk(inputT0, trt.TopKOperation.MAX, 2, 1 << 1)
#---------------------------------------------------------- --------------------# 替换部分
network.mark_output(topKLayer.get_output(0))
network.mark_output(topKLayer.get_output(1))
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)))
outputH1 = np.empty(context.get_binding_shape(2), dtype=trt.nptype(engine.get_binding_dtype(2)))
_, inputD0 = cudart.cudaMallocAsync(inputH0.nbytes, stream)
_, outputD0 = cudart.cudaMallocAsync(outputH0.nbytes, stream)
_, outputD1 = cudart.cudaMallocAsync(outputH1.nbytes, stream)

cudart.cudaMemcpyAsync(inputD0, inputH0.ctypes.data, inputH0.nbytes,
cudart.cudaMemcpyKind.cudaMemcpyHostToDevice, stream)
context.execute_async_v2([int(inputD0), int(outputD0), int(outputD1)], stream)
cudart.cudaMemcpyAsync(outputH0.ctypes.data, outputD0, outputH0.nbytes,
cudart.cudaMemcpyKind.cudaMemcpyDeviceToHost, stream)
cudart.cudaMemcpyAsync(outputH1.ctypes.data, outputD1, outputH1.nbytes,
cudart.cudaMemcpyKind.cudaMemcpyDeviceToHost, stream)
cudart.cudaStreamSynchronize(stream)

print("inputH0 :", data.shape)
print(data)
print("outputH0:", outputH0.shape)
print(outputH0)
print("outputH1:", outputH1.shape)
print(outputH1)

cudart.cudaStreamDestroy(stream)
cudart.cudaFree(inputD0)
cudart.cudaFree(outputD0)
cudart.cudaFree(outputD1)
```

- 输入张量形状 (1,3,4,5)

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

- 输出张量 0 形状 (1,2,4,5)，对次高维上相同 HW 位置的元素取降序前 2 名

$$\left[\left[\begin{bmatrix} 34. & 52. & 58. & 54. & 50. \\ 48. & 59. & 57. & 30. & 18. \\ 45. & 38. & 56. & 36. & 37. \\ 53. & 51. & 41. & 55. & 47. \end{bmatrix} \begin{bmatrix} 13. & 15. & 32. & 46. & 49. \\ 10. & 44. & 29. & 20. & 12. \\ 19. & 33. & 40. & 31. & 25. \\ 23. & 42. & 24. & 35. & 26. \end{bmatrix}\right]\right]$$

- 输出张量 1 形状 (1,2,4,5)，表示输出张量 0 中各元素在输入张量中的通道号
- $output0\left[output1\left[c,h,w\right],h,w\right] = input\left[c,h,w\right]$

$$\left[\begin{bmatrix} 1 & 0 & 2 & 1 & 2 \\ 2 & 0 & 1 & 1 & 2 \\ 0 & 1 & 2 & 1 & 0 \\ 2 & 2 & 2 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 2 & 0 \\ 1 & 2 & 2 & 2 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 2 & 2 \end{bmatrix}\right]$$

---

op

```
topKLayer = network.add_topk(inputT0, trt.TopKOperation.MIN, 2, 1 << 1)
topKLayer.op = trt.TopKOperation.MAX  # 重设 topk 极取值方向
```

- 指定 op=trt.TopKOperation.MAX，输出张量 0/1 形状 (1,2,4,5)，结果与初始示例代码相同
- 指定 op=trt.TopKOperation.MIN，输出张量 0 形状 (1,2,4,5)，对次高维上相同 HW 位置的元素取升序前 2 名

$$\left[\left[\begin{bmatrix} 9. & 14. & 2. & 27. & 39. \\ 0. & 43. & 22. & 6. & 11. \\ 4. & 5. & 8. & 28. & 7. \\ 3. & 21. & 1. & 16. & 17. \end{bmatrix} \begin{bmatrix} 13. & 15. & 32. & 46. & 49. \\ 10. & 44. & 29. & 20. & 12. \\ 19. & 33. & 40. & 31. & 25. \\ 23. & 42. & 24. & 35. & 26. \end{bmatrix}\right]\right]$$

- 指定 op=trt.TopKOperation.MIN，输出张量 0 形状 (1,2,4,5)，表示输出张量 0 中各元素在输入张量中的通道号

$$\left[\begin{bmatrix} 0 & 2 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 2 & 2 & 0 & 2 & 2 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 2 & 0 \\ 1 & 2 & 2 & 2 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 2 & 2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 & 1 & 2 \\ 2 & 0 & 1 & 1 & 2 \\ 0 & 1 & 2 & 1 & 0 \\ 2 & 2 & 2 & 0 & 1 \end{bmatrix}\right]$$

- 可用的选项

trt.TopKOperation 名	说明
MAX	从最大值开始取
MIN	从最小值开始取

---

k

```
topKLayer = network.add_topk(inputT0, trt.TopKOperation.MAX, 3, 1 << 1)
topKLayer.k = 2  # 重设取极值个数
```

- 指定 k=2，输出张量 0/1 形状 (1,2,4,5)，结果与初始示例代码相同
- TensorRT6、7 和 8 上规定了最大 k 为 3840，超出后报错

```
[TRT] [E] 3: [layers.h::setK::1268] Error Code 3: API Usage Error (Parameter check failed at:
/_src/build/cuda-11.4/8.2/x86_64/release/optimizer/api/layers.h::setK::1268, condition: k > 0 && k <=
MAX_TOPK_K
)
```

---

axes

```
topKLayer = network.add_topk(inputT0, trt.TopKOperation.MAX, 2, 1<<2)
topKLayer.axes = 1 << 1  # 重设取极值的维度
```

- 指定 axes=1<<0，因为输入张量该维上宽度不足 2，报错：

```
[TRT] [E] 4: (Unnamed Layer* 0) [TopK]: K not consistent with dimensions
[TRT] [E] 4: (Unnamed Layer* 0) [TopK]: K not consistent with dimensions
[TRT] [E] 4: (Unnamed Layer* 0) [TopK]: K not consistent with dimensions
[TRT] [E] 4: [network.cpp::validate::2871] Error Code 4: Internal Error (Layer (Unnamed Layer* 0) [TopK]
failed validation)
```

- 指定 axes=1<<1，输出张量 0/1 形状 (1,2,4,5)，结果与初始示例代码相同
- 指定 axes=1<<2，输出张量 0/1 形状 (1,3,2,5)，对季高维上相同 CW 位置的元素取降序前 2 名

$$\left[\left[\begin{bmatrix} 45. & 59. & 22. & 55. & 49. \\ 23. & 52. & 8. & 31. & 37. \end{bmatrix}\begin{bmatrix} 34. & 43. & 57. & 54. & 47. \\ 19. & 42. & 40. & 36. & 39. \end{bmatrix}\begin{bmatrix} 53. & 51. & 58. & 46. & 50. \\ 48. & 44. & 56. & 35. & 26. \end{bmatrix}\right]\right]$$
$$\left[\left[\begin{bmatrix} 2 & 1 & 1 & 3 & 0 \\ 3 & 0 & 2 & 2 & 2 \end{bmatrix}\begin{bmatrix} 0 & 1 & 1 & 0 & 3 \\ 2 & 3 & 2 & 2 & 0 \end{bmatrix}\begin{bmatrix} 3 & 3 & 0 & 0 & 0 \\ 1 & 1 & 2 & 3 & 3 \end{bmatrix}\right]\right]$$

- 指定 axes=1<<3，输出张量 0/1 形状 (1,3,4,2)，对叔高维上相同 CH 位置的元素取降序前 2 名

$$\left[\left[\begin{bmatrix} 52. & 49. \\ 59. & 22. \\ 45. & 37. \\ 55. & 23. \end{bmatrix}\begin{bmatrix} 54. & 39. \\ 57. & 43. \\ 40. & 38. \\ 47. & 42. \end{bmatrix}\begin{bmatrix} 58. & 50. \\ 48. & 44. \\ 56. & 28. \\ 53. & 51. \end{bmatrix}\right]\right]$$

- 不能同时指定两个及以上的 axes，如指定 axes=(1<<2)+(1<<3)，会收到报错：

```
[TRT] [E] 4: [graphShapeAnalyzer.cpp::computeOutputExtents::1026] Error Code 4: Miscellaneous ((Unnamed
Layer* 0) [TopK]: error while computing output extent)
[TRT] [E] 4: [graphShapeAnalyzer.cpp::computeOutputExtents::1026] Error Code 4: Miscellaneous ((Unnamed
Layer* 0) [TopK]: error while computing output extent)
[TRT] [E] 3: (Unnamed Layer* 0) [TopK]: reduceAxes must specify exactly one dimension
[TRT] [E] 4: [network.cpp::validate::2871] Error Code 4: Internal Error (Layer (Unnamed Layer* 0) [TopK]
failed validation)
```