深蓝学院
shenlanxueyuan.com

TensorRT转换ONNX模型

课程目标

- ONNX介绍
- TensorRT 转换模型主要痛点
- onnx parser + onnx-graphsurgeon
- 转换和调试工具-polygraphy

ONNX

ONNX (Open Neural Network Exchange，开放神经网络交换)：用来表示深度学习模型的开放格式。所谓开放就是ONNX定义了一组与环境、平台均无关的标准格式，来增强各种AI模型的可交互性。

PyTorch
TensorFlow
Paddle
……

→ ONNX →

TensorRT (Nvidia)
MNN (Alibaba)
TNN (Tencent)
ONNXRuntime
……

ONNX官网：https://onnx.ai/

ONNX

ONNX模型组成：Graph，Node，Tensor

Graph 参数

- **nodes** (*Sequence[Node]*) – A list of the nodes in this graph.
- **inputs** (*Sequence[Tensor]*) – A list of graph input Tensors.
- **outputs** (*Sequence[Tensor]*) – A list of graph output Tensors.
- **name** (*str*) – The name of the graph. Defaults to “onnx_graphsurgeon_graph”.
- **doc_string** (*str*) – A doc_string for the graph. Defaults to “”.
- **opset** (*int*) – The **ONNX opset** to use when exporting this graph.
- **producer_name** (*str*) – The name of the tool used to generate the model. Defaults to “”.
- **producer_version** (*str*) – The version of the generating tool. Defaults to “”.

https://docs.nvidia.com/deeplearning/tensorrt/onnx-graphsurgeon/docs/ir/graph.html

ONNX

ONNX模型组成：Graph， Node， Tensor

Node 参数

- **op** (*str*) – The operation this node performs.
- **name** (*str*) – The name of this node.
- **attrs** (*Dict[str, object]*) – A dictionary that maps attribute names to their values.
- **inputs** (*List[Tensor]*) – A list of zero or more input Tensors.
- **outputs** (*List[Tensor]*) – A list of zero or more output Tensors.

函数

node.i(tensor_idx=0, producer_idx=0)

例子

assert node.i()==node.inputs[0].inputs[0]

assert node.i(1, 2)==node.inputs[1].inputs[2]

node.o(*consumer_idx=0*, *tensor_idx=0*)

例子

assert node.o() == node.outputs[0].outputs[0]

assert node.o (1, 2)== node.outputs[1].outputs[2]

https://docs.nvidia.com/deeplearning/tensorrt/onnx-graphsurgeon/docs/ir/graph.html

ONNX

ONNX模型组成：Graph，Node，Tensor

Tensor是个基类，有两种实现：Variable 和 Constant。

Variable：值和大小在推理之前无法确定，比如某一层的输出。

• **name** (*str*) – The name of the tensor.

• **dtype** (*numpy.dtype*) – The data type of the tensor.

• **shape** (*Sequence[Union[int, str]]*) – The shape of the tensor. This may contain strings if the model uses dimension parameters.

Constant：值和大小是确定的，比如Linear的权值。

•**name** (*str*) – The name of the tensor.

•**values** (*numpy.ndarray*) – The values in this tensor, NumPy array.

•**data_location** (*int*) – An enum value indicating the location where the tensor data is stored. Generally, this will come from onnx.TensorProto.DataLocation.

https://docs.nvidia.com/deeplearning/tensorrt/onnx-graphsurgeon/docs/ir/tensor/toc.html

背景知识点-Lower

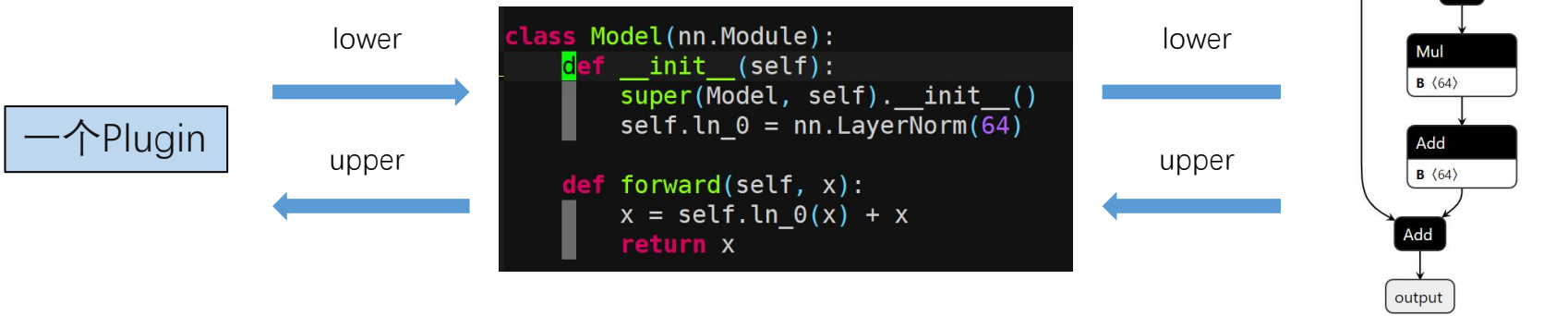
深度学习算子 Lower 概念：用一个或多个常规的算子来模拟出模型里面所不支持的、复杂的算子。反之为 Upper 。

- Lower：通过算子组合，提高训练和推理的灵活性和兼容性。比如各种训练框架和ONNX。
- Upper：通过算子合并，提高训练和推理的速度。比如各种推理框架和一些优化后的训练框架（oneflow, Megatron-LM等）
- TensorRT是Upper的典型代表。

背景知识点-Lower

深度学习算子 Lower 概念：用一个或多个常规的算子来模拟出模型里面所不支持的、复杂的算子。反之为 Upper 。

- Lower：通过算子组合，提高训练和推理的灵活性和兼容性。比如各种训练框架和ONNX。
- Upper：通过算子合并，提高训练和推理的速度。比如各种推理框架和一些优化后的训练框架（oneflow, Megatron-LM等）
- TensorRT是Upper的典型代表。



背景知识点-Myelin

Myelin

资料很少

个人理解：深度学习算子的 CUDA 代码生成器

TensorRT/lib

```
libmyelin_compiler_static.a
libmyelin_executor_static.a
libmyelin_pattern_library_static.a
libmyelin_pattern_runtime_static.a
libmyelin.so -> libmyelin.so.1
libmyelin.so.1 -> libmyelin.so.1.1.116
libmyelin.so.1.1.116
```

背景知识点-Myelin

TensorRT 筛选可以合并优化的子模型，Myelin生成该子模型的kernel。

```
def addGELU(self, x, layer_name=None, precision=None):
        POW = self.network.add_constant((1, 1, 1), trt.Weights(np.ascontiguousarray([3.0],
dtype=np.float32)))
        MULTIPLY = self.network.add_constant((1, 1, 1), trt.Weights(np.ascontiguousarray([0.044715],
dtype=np.float32)))
        SQRT = self.network.add_constant((1, 1, 1),
trt.Weights((np.ascontiguousarray([0.7978845608028653558798921198687 6], dtype=np.float32))))
        ONE = self.network.add_constant((1, 1, 1), trt.Weights((np.ascontiguousarray([1.0],
dtype=np.float32))))
        HALF = self.network.add_constant((1, 1, 1), trt.Weights((np.ascontiguousarray([0.5],
dtype=np.float32))))
        X_pow = self.network.add_elementwise(x, POW.get_output(0), trt.ElementWiseOperation.POW)
        X_pow_t = X_pow.get_output(0)
        X_mul = self.network.add_elementwise(X_pow_t, MULTIPLY.get_output(0),
trt.ElementWiseOperation.PROD)
        X_add = self.network.add_elementwise(x, X_mul.get_output(0), trt.ElementWiseOperation.SUM)
        X_sqrt = self.network.add_elementwise(X_add.get_output(0), SQRT.get_output(0),
trt.ElementWiseOperation.PROD)
        X_sqrt_tensor = X_sqrt.get_output(0)
        X_tanh = self.network.add_activation(X_sqrt_tensor, trt.ActivationType.TANH)
        X_tanh_tensor = X_tanh.get_output(0)
        X_one = self.network.add_elementwise(X_tanh_tensor, ONE.get_output(0),
trt.ElementWiseOperation.SUM)
        CDF = self.network.add_elementwise(X_one.get_output(0), HALF.get_output(0),
trt.ElementWiseOperation.PROD)
        gelu_layer = self.network.add_elementwise(CDF.get_output(0), x, trt.ElementWiseOperation.PROD)
```

myelin

自动合并

Foreign node

缺点：

1. 自动生成的kernel速度一般；
2. 自动合并的算子可能会加重fp16/int8模式下的精度损失。

TensorRT 转换模型主要痛点

PyTorch
TensorFlow
Paddle
.....
→ ONNX → TensorRT

	痛点	NVIDIA的应对方案
转换问题	API 使用难度高	ONNX Parser 几乎所有训练框架都支持转成ONNX模型
	TensorRT支持算子不全	TensorRT 8.4 GA版本开始增加支持的算子 和 onnx-graphsurgeon
优化问题	模型结构方面优化较差，基本是简单算子合并	myelin
调试问题	调试难度大	转换和调试工具-polygraphy

TRT 8.4开始，onnx parser和myelin的纯TRT方案，是TRT team发展的方向。

TRT 8.4的性能还差些，尤其有dynamic input shape时，之后的TRT 8.5和8.6会全力解决这些问题。

备注：EA version stands for early access (It is before actual release). GA stands for general availability. GA is stable version and completely tested.

onnx parser

官配 Parser

https://github.com/onnx/onnx-tensorrt

TensorRT/lib

```
libnvonnxparser.so -> libnvonnxparser.so.7
libnvonnxparser.so.7 -> libnvonnxparser.so.7.2.2
libnvonnxparser.so.7.2.2
libnvonnxparser_static.a
```

onnx parser

```python
def onnx2trt(onnxFile, plan_name):
    logger = trt.Logger(trt.Logger.VERBOSE)

    builder = trt.Builder(logger)
    config = builder.create_builder_config()
    profile = builder.create_optimization_profile()
    network = builder.create_network(1<<int(trt.NetworkDefinitionCreationFlag.EXPLICIT_BATCH))
    config.max_workspace_size = 3<<30

    parser = trt.OnnxParser(network, logger)

    with open(onnxFile, 'rb') as model:
        if not parser.parse(model.read()):
            print("Failed parsing ONNX file!")

    input_ids = network.get_input(0)
    token_type_ids = network.get_input(1)
    input_mask = network.get_input(2)
    profile.set_shape(input_ids.name, (1, 6), (1, 64), (1, 256))
    profile.set_shape(token_type_ids.name, (1, 6), (1, 64), (1, 256))
    profile.set_shape(input_mask.name, (1, 6), (1, 64), (1, 256))
    config.add_optimization_profile(profile)

    engine = builder.build_engine(network, config)

    serialized_engine = engine.serialize()

    with open(plan_name, "wb") as fout:
        fout.write(serialized_engine)
```

创建资源

Parser

build并序列化

onnx parser + onnx-graphsurgeon

ONNX Parser 的痛点（几乎是所有Parser的缺点）

分析

痛点	解决方案
支持算子不全，PyTorch 目前有100+算子且在不断增加	初级方案：lower + myelin，nvidia大力推进的方向
	进阶方案：onnx-graphsurgeon
ONNX lower 导致模型结构容易非常碎且冗余，影响速度	onnx-graphsurgeon
有些节点阻碍 TensorRT myelin 的自动合并优化	
无法支持其他模型转换ONNX模型失败的情况，比如 pytorch c++ extension	
无法满足手动合并算子进行深度优化的情景	

onnx parser + onnx-graphsurgeon

onnx-graphsurgeon: ONNX模型的编辑器，是NVIDIA推出的TensorRT开发辅助工具。

功能

修改计算图: 图属性/节点/张量/节点和张量的连接/权重

修改子图: 添加/删除/替换/ 隔离

优化计算图: 常量折叠/拓扑排序/去除无用层

开源例子：08-Tool/OnnxGraphSurgeon

共有9个例子，包含创建模型、隔离子图、替换节点、常量折叠、删除节点、shape 操作

https://www.bilibili.com/video/BV19T4y1e7XK/

https://github.com/NVIDIA/trt-samples-for-hackathon-cn/tree/master/cookbook/08-Tool/OnnxGraphSurgeon

onnx parser + onnx-graphsurgeon

实践操作：英伟达TensorRT加速AI推理 Hackathon 2022 —— Transformer模型优化

初赛是利用 TensorRT 加速 ASR 模型 WeNet（包含 encoder 和 decoder 两个部分）

- 使用的镜像：registry.cn-hangzhou.aliyuncs.com/trt2022/dev
- 讲座地址：https://www.bilibili.com/video/BV15Y4y1W73E
- 配套范例：https://github.com/NVIDIA/trt-samples-for-hackathon-cn
- /workspace含有供选手使用的输入文件和测试文件（只读，请勿修改）
- /workspace/encoder.onnx 和 /workspace/decoder.onnx 是在 PyTorch 中训练好的 WeNet 模型，ONNX格式

https://tianchi.aliyun.com/competition/entrance/531953/information

onnx parser + onnx-graphsurgeon

例子：英伟达TensorRT加速AI推理 Hackathon 2022 —— Transformer模型优化

问题一：解决不支持的算子；

问题二：合并LayerNorm算子，并插入plugin替换；

代码：https://github.com/LitLeo/TensorRT_Tutorial/tree/master/Hackathon2022。

初赛总结：https://www.bilibili.com/video/BV1i3411G7vN

复赛总结： https://github.com/NVIDIA/trt-samples-for-hackathon-cn/tree/master/Hackathon/2022

转换和调试工具-polygraphy

转换完模型后可能遇见的问题：

- 模型转换后计算结果不正确
- FP16 / INT8 结果精度差

TensorRT 转换调试器-polygraphy

Meet all your needs

解决方案：

- 找出计算错误的层
- 找到导致精度不足的层
- 合并容易溢出的层
- ……

转换和调试工具-polygraphy

polygraphy：深度学习模型调试器，包含 python API 和命令行工具 (这里只介绍命令行)功能。

功能

- 使用多种后端运行推理计算，包括 TensorRT, onnxruntime, TensorFlow
- 比较不同后端的逐层计算结果
- 由模型文件生成 TensorRT引擎并序列化为TensorRT模型（一般.plan后缀）
- 查看模型网络的逐层信息
- 修改 Onnx 模型，如提取子图，计算图化简
- 分析Onnx转TensorRT 失败原因，将原计算图中可以/不可以转 TensorRT 的子图分割保存
- 隔离 TensorRT 中的错误 tactic

https://docs.nvidia.com/deeplearning/tensorrt/polygraphy/docs/index.html

转换和调试工具-polygraphy

一共有7种模式：run, convert, inspect, surgeon, template, debug, data。

Run：运行模型模式，并可以同时运行ONNX模型，并逐层对比结果。

Convert：转换模型模式。

Inspect：检查模式。可以查看网络的详细信息；判断是否支持ONNX模型；切割子图。

Surgeon：优化计算图模式。

Template：命令转脚本。

Debug：检查转换错误并分离出一个最大子图。

Data：调整分析权值。

NVIDIA官方TensorRT教程：https://www.bilibili.com/video/BV19T4y1e7XK，第11分钟开始

转换和调试工具-polygraphy

学习资料：

- 官方文档：https://docs.nvidia.com/deeplearning/tensorrt/polygraphy/docs/index.html
- 教学视频：https://www.bilibili.com/video/BV19T4y1e7XK
- 博客：https://blog.csdn.net/TracelessLe/article/details/120656484
- 案例：https://zhuanlan.zhihu.com/p/436017991

深蓝学院
shenlanxueyuan.com

感谢聆听！

Thanks for Listening