TensorRT中的常见问题

以下部分有助于回答有关 NVIDIA TensorRT 典型用例的最常见问题。

14.1. FAQs

本部分旨在帮助解决问题并回答我们最常问的问题。

问：如何创建针对多种不同批次大小进行优化的引擎?

答：虽然 TensorRT 允许针对给定批量大小优化的引擎以任何较小的大小运行，但这些较小大小的性能无法得到很好的优化。要针对多个不同的批量大小进行优化，请在分配给 `OptProfilerSelector::kOPT` 的维度上创建优化配置文件。

问：引擎和校准表是否可以跨TensorRT版本移植?

答：不会。内部实现和格式会不断优化，并且可以在版本之间更改。因此，不保证引擎和校准表与不同版本的TensorRT二进制兼容。使用新版本的TensorRT时，应用程序必须构建新引擎和 INT8 校准表。

问：如何选择最佳的工作空间大小?

答：一些 TensorRT 算法需要 GPU 上的额外工作空间。方法 `IBuilderConfig::setMemoryPoolLimit()` 控制可以分配的最大工作空间量，并防止构建器考虑需要更多工作空间的算法。在运行时，创建 `IExecutionContext` 时会自动分配空间。即使在 `IBuilderConfig::setMemoryPoolLimit()` 中设置的数量要高得多，分配的数量也不会超过所需数量。因此，应用程序应该允许 TensorRT 构建器尽可能多的工作空间；在运行时，TensorRT 分配的数量不超过这个，通常更少。

问：如何在多个 GPU 上使用TensorRT ?

答：每个 `ICudaEngine` 对象在实例化时都绑定到特定的 GPU，无论是由构建器还是在反序列化时。要选择 GPU，请在调用构建器或反序列化引擎之前使用 `cudaSetDevice()` 。每个 `IExecutionContext` 都绑定到与创建它的引擎相同的 GPU。调用 `execute()` 或 `enqueue()` 时，如有必要，请通过调用 `cudaSetDevice()` 确保线程与正确的设备相关联。

问：如何从库文件中获取TensorRT的版本?

A: 符号表中有一个名为 `tensorrt_version_#_#_#_#` 的符号，其中包含TensorRT版本号。在 Linux 上读取此符号的一种可能方法是使用 nm命令，如下例所示：

```
$ nm -D libnvinfer.so.* | grep tensorrt_version
00000000abcd1234 B tensorrt_version_#_#_#_#
```

问：如果我的网络产生了错误的答案，我该怎么办?
答：您的网络生成错误答案的原因有多种。以下是一些有助于诊断问题的故障排除方法：

- 打开日志流中的VERBOSE级别消息并检查 TensorRT 报告的内容。
- 检查您的输入预处理是否正在生成网络所需的输入格式。
- 如果您使用降低的精度，请在 FP32 中运行网络。如果它产生正确的结果，则较低的精度可能对网络的动态范围不足。
- 尝试将网络中的中间张量标记为输出，并验证它们是否符合您的预期。
- 注意：将张量标记为输出会抑制优化，因此会改变结果。

您可以使用Polygraphy来帮助您进行调试和诊断。

问：如何在TensorRT中实现批量标准化?

答：批量标准化可以使用TensorRT中的 `IElementWiseLayer` 序列来实现。进一步来说：

```
adjustedScale = scale / sqrt(variance + epsilon)
batchNorm = (input + bias - (adjustedScale * mean)) * adjustedScale
```

问：为什么我的网络在使用 DLA 时比不使用 DLA 时运行得更慢?

答：DLA 旨在最大限度地提高能源效率。根据 DLA 支持的功能和 GPU 支持的功能，任何一种实现都可以提高性能。使用哪种实现取决于您的延迟或吞吐量要求以及您的功率预算。由于所有 DLA 引擎都独立于 GPU 并且彼此独立，因此您还可以同时使用这两种实现来进一步提高网络的吞吐量。

问：TensorRT支持INT4量化还是INT16量化?

答：TensorRT 目前不支持 INT4 和 INT16 量化。

问：TensorRT 何时会在 UFF 解析器中支持我的网络所需的层 XYZ?

答：UFF 已弃用。我们建议用户将他们的工作流程切换到 ONNX。 TensorRT ONNX 解析器是一个开源项目。

问：我可以使用多个 TensorRT 构建器在不同的目标上进行编译吗?

答：TensorRT 假设它所构建的设备的所有资源都可用于优化目的。同时使用多个 TensorRT 构建器（例如，多个 `trtexec` 实例）在不同的目标（`DLA0`、`DLA1` 和 GPU）上进行编译可能会导致系统资源超额订阅，从而导致未定义的行为（即计划效率低下、构建器失败或系统不稳定）。

建议使用带有 `--saveEngine` 参数的 `trtexec` 分别为不同的目标（DLA 和 GPU）编译并保存它们的计划文件。然后可以重用此类计划文件进行加载（使用带有 `--loadEngine` 参数的 `trtexec` ）并在各个目标（`DLA0`、`DLA1`、`GPU` ）上提交多个推理作业。这个两步过程在构建阶段缓解了系统资源的过度订阅，同时还允许计划文件的执行在不受构建器干扰的情况下继续进行。

问：张量核心(tensor core)加速了哪些层?

大多数数学绑定运算将通过张量核(tensor core)加速 - 卷积、反卷积、全连接和矩阵乘法。在某些情况下，特别是对于小通道数或小组大小，另一种实现可能更快并且被选择而不是张量核心实现。

14.2.Understanding Error Messages

如果在执行过程中遇到错误，TensorRT 会报告一条错误消息，旨在帮助调试问题。以下部分讨论了开发人员可能遇到的一些常见错误消息。

UFF 解析器错误消息

下表捕获了常见的 UFF 解析器错误消息。

Error Message	Description
`The input to the Scale Layer is required to have a minimum of 3 dimensions.`	This error message can occur due to incorrect input dimensions. In UFF, input dimensions should always be specified with the implicit batch dimension *not* included in the specification.
`Invalid scale mode, nbWeights: <X>`	
`kernel weights has count <X> but <Y> was expected`	
`<NODE> Axis node has op <OP>, expected Const. The axis must be specified as a Const node.`	As indicated by the error message, the axis must be a build-time constant in order for UFF to parse the node correctly.

ONNX 解析器错误消息

下表捕获了常见的 ONNX 解析器错误消息。有关特定 ONNX 节点支持的更多信息，请参阅 operators支持文档。

Error Message	Description
`<X> must be an initializer!`	These error messages signify that an ONNX node input tensor is expected to be an initializer in TensorRT. A possible fix is to run constant folding on the model using TensorRT's Polygraphy tool: `polygraphy surgeon sanitize model.onnx --fold-constants --output model_folded.onnx`
`!inputs.at(X).is_weights()`	
`getPluginCreator() could not find Plugin <operator name> version 1`	This is an error stating that the ONNX parser does not have an import function defined for a particular operator, and did not find a corresponding plugin in the loaded registry for the operator.

TensorRT 核心库错误消息

下表捕获了常见的 TensorRT 核心库错误消息。

	Error Message	Descriptio
Installation Errors	`Cuda initialization failure with error <code>. Please check cuda installation: http://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html.`	This error r occur if the NVIDIA driv corrupt. Re for instruct installing C NVIDIA driv operating s
Builder Errors	`Internal error: could not find any implementation for node <name>. Try increasing the workspace size with IBuilderConfig::setMemoryPoolLimit().`	This error r because th implement given node that can op given work usually occ workspace insufficient indicate a b the worksp suggested report a bu Do I Report
	`<layer-name>: (kernel|bias) weights has non-zero count but null values` `<layer-name>: (kernel|bias) weights has zero count but non-null values`	This error r when there between th count fields data struct the builder 0, then the must conta pointer; otl count must and values non-null po
	`Builder was created on device different from current device.`	This error r show up if 1. Create target then 2. Called `cudaS` target differe 3. Attem IBuild engine Ensure you `IBuilder` the GPU th create the
	You can encounter error messages indicating that the tensor dimensions do not match the semantics of the given layer. Carefully read th documentation on NvInfer.h on the usage of each layer and the expected dimensions of the tensor inputs and outputs to the layer.	

	Error Message	Descriptio
INT8 Calibration Errors	`Tensor <X> is uniformly zero.`	This warnir should be t error when distributior uniformly z network, th distributior uniformly z following s 1. Const all zer error. 2. Activa outpu negati an err 3. Data c forced to compu the pr emit a here.[1] 4. User c any ca image warni
	`Could not find scales for tensor <X>.`	This error r indicates th failure occu scaling fact This could INT8 calibr insufficient for networl more infor sampleINT opensourc directory ir repository calibration
	`The engine plan file is not compatible with this version of TensorRT, expecting (format|library) version <X> got <Y>, please rebuild.`	This error r occur if you TensorRT u PLAN file th incompatib current ver TensorRT. I the same v TensorRT v the engine
	`The engine plan file is generated on an incompatible device, expecting compute <X> got compute <Y>, please rebuild.`	This error r occur if you engine on a different co capability t that is used engine.

	Error Message	Descriptio
	`Using an engine plan file across different models of devices is not recommended and is likely to affect performance or even cause errors.`	This warning message can occur if you build an engine on a device with the same compute capabilit y but is not identical to the device that is used to run the engine. As indicate d by the warning, it is highly recomm ended to use a device of the same model when generati ng the engine and deployin g it to avoid compati bility issues.
	`GPU memory allocation failed during initialization of (tensor|layer): <name> GPU memory`	These erro occur if the GPU memo instantiate TensorRT e that the GP available m contain the weights an tensors.
	`Allocation failed during deserialization of weights.`	
	`GPU does not meet the minimum memory requirements to run this engine ...`	

	`Network needs native FP16 and platform does not have native FP16`	**Descriptio** This error r occur if you deserialize uses FP16 a GPU that d FP16 arithr need to reb without FP inference c GPU to a m supports FI inference.
	`Custom layer <name> returned non-zero initialization`	This error r occur if the `initializ` a given plu a non-zero the implem layer to del further. For information TensorRT L

14.3. Code Analysis Tools

14.3.1. Compiler Sanitizers

Google sanitizers 是一组代码分析工具。

14.3.1.1. Issues With dlopen And Address Sanitizer

`Sanitizer` 存在一个已知问题，在此处记录。在 `sanitizer` 下在 TensorRT 上使用 `dlopen` 时，会报告内存泄漏，除非采用以下两种解决方案之一：

1. 在 `Sanitizer` 下运行时不要调用 `dlclose` 。
2. 将标志 `RTLD_NODELETE` 传递给 `dlopen` 。

14.3.1.2. Issues With dlopen And Thread Sanitizer

从多个线程使用 `dlopen` 时，线程清理程序可以列出错误。为了抑制此警告，请创建一个名为 `tsan.supp` 的文件并将以下内容添加到文件中：

```
race::dlopen
```

在 thread sanitizer 下运行应用程序时，使用以下命令设置环境变量：

```
export TSAN_OPTIONS="suppressions=tsan.supp"
```

14.3.1.3. Issues With CUDA And Address Sanitizer

在此处记录的 CUDA 应用程序中存在一个已知问题。为了在地址清理器下成功运行 CUDA 库（例如 TensorRT），请将选项 `protect_shadow_gap=0` 添加到 `ASAN_OPTIONS` 环境变量中。

在 CUDA 11.4 上，有一个已知错误可能会在地址清理程序中触发不匹配的分配和释放错误。将 `alloc_dealloc_mismatch=0` 添加到 `ASAN_OPTIONS` 以禁用这些错误。

14.3.1.4. Issues With Undefined Behavior Sanitizer

UndefinedBehaviorSanitizer (UBSan)使用 `-fvisibility=hidden` 选项报告误报，如此处所述。您必须添加 `-fno-sanitize=vptr` 选项以避免 `UBSan` 报告此类误报。

14.3.2. Valgrind

`Valgrind` 是一个动态分析工具框架，可用于自动检测应用程序中的内存管理和线程错误。

某些版本的 `valgrind` 和 `glibc` 受到错误的影响，该错误会导致在使用 `dlopen` 时报告错误的内存泄漏，这可能会在 `valgrind` 的 `memcheck` 工具下运行 TensorRT 应用程序时产生虚假错误。要解决此问题，请将以下内容添加到此处记录的 `valgrind` 抑制文件中：

```
{
   Memory leak errors with dlopen
   Memcheck:Leak
   match-leak-kinds: definite
   ...
   fun:*dlopen*
   ...
}
```

在 CUDA 11.4 上，有一个已知错误可能会在 `valgrind` 中触发不匹配的分配和释放错误。将选项 `--show-mismatched-frees=no` 添加到 `valgrind` 命令行以抑制这些错误。

14.3.3. Compute Sanitizer

在计算清理程序下运行 TensorRT 应用程序时， cuGetProcAddress可能会因缺少函数而失败，错误代码为 500。可以使用 `--report-api-errors no` 选项忽略或抑制此错误。这是由于 CUDA 向后兼容性检查功能是否可用于 CUDA 工具包/驱动程序组合。这些功能在 CUDA 的更高版本中引入，但在当前平台上不可用。