Triton Inference Server Quick Start On Jetson

Author:Kenny Zhou

Last Update: 2022-04-11 10:19:50

Jetson 安装 JetPack 4.6.1

因为 JetPack 5.0 未释出，在 Jetson Install Triton-inference-server 中选择 Release 2.19.0 corresponding to NGC container 22.02 版本：

安装步骤：

JetPack 4.6.1 的 Triton 版本在随附的 tar 文件中提供：tritonserver2.19.0-jetpack4.6.1.tgz

tar 文件包含 Triton 服务器可执行和共享库以及 C++ 和 Python 客户端库和示例。有关如何在 JetPack 上安装和使用 Triton 的更多信息，请参阅 jetson.md。

TIS on Jetson Quick Start

server

git clone https://github.com/triton-inference-server/server
model-repository 即为 server/docs/examples/model_repository

```
~/server/docs/examples$ sh fetch_models.sh
```

解压缩 tritonserver2.19.0-jetpack4.6.1.tgz

```
mkdir ~/TritonServer && tar -xzvf tritonserver2.19.0-jetpack4.6.1.tgz -C ~/TritonServer
```

在文件夹中
~/TIS/bin$ ./tritonserver --model-repository=/home/nvidia/server/docs/examples/model_repository --backend-directory=/home/nvidia/TIS/backends (--strict-model-config=false)
自动生成的模型配置

默认情况下，每个模型都必须提供包含所需设置的模型配置文件。但是，如果 Triton 使用 --strict-model-config=false 选项启动，那么在某些情况下，模型配置文件的所需部分可以由 Triton 自动生成。模型配置的必需部分是最小模型配置中显示的那些设置。具体来说，TensorRT、TensorFlow 保存模型和 ONNX 模型不需要模型配置文件，因为 Triton 可以自动导出所有需要的设置。所有其他模型类型必须提供模型配置文件。

使用 --strict-model-config=false 时，您可以查看使用模型配置端点为模型生成的模型配置。最简单的方法是使用 curl 之类的实用程序：

$ curl localhost:8000/v2/models/<模型名称>/config 这将返回生成的模型配置的 JSON 表示。您可以从中获取 JSON 的 max_batch_size、输入和输出部分，并将其转换为 config.pbtxt 文件。Triton 只生成模型配置的最小部分。您仍然必须通过编辑 config.pbtxt 文件来提供模型配置的可选部分。

PS: if error with libb64.so.0d:

sudo apt-get install libb64-0d

and error with libre4.so.4:

sudo apt-get install libre2-dev

PPS:if error: creating server: Internal - failed to load all models

You can try using --exit-on-error=false when launching the server.Use Triton's ready endpoint to verify that the server and the models are ready for inference. From the host system use curl to access the HTTP endpoint that indicates server status.

$ curl -v localhost:8000/v2/health/ready

...

< HTTP/1.1 200 OK

< Content-Length: 0

< Content-Type: text/plain

client

使用 docker pull 从 NGC 获取客户端库和示例图像。

$ docker pull nvcr.io/nvidia/tritonserver:<xx.yy>-py3-sdk 其中 <xx.yy> 是您要提取的版本（22.02）。运行客户端映像。

$ docker run -it --rm --net=host nvcr.io/nvidia/tritonserver:<xx.yy>-py3-sdkFrom

within the nvcr.io/nvidia/tritonserver:<xx.yy>-py3-sdk image, run the example image-client application to perform image classification using the example densenet_onnx model.

To send a request for the densenet_onnx model use an image from the /workspace/images directory. In this case we ask for the top 3 classifications.

```
$ /workspace/install/bin ./image_client -m densenet_onnx -c 3 -s INCEPTION /workspace/images/mug.jpg

Request 0, batch size 1

Image '/workspace/images/mug.jpg':

    15.346230 (504) = COFFEE MUG

    13.224326 (968) = CUP

    10.422965 (505) = COFFEEPOT
```

-i flag to use GRPC:

```
/workspace/install/bin# ./image_client -m densenet_onnx -c 3 -s INCEPTION /workspace/images/mug.jpg -i grpc
```

-u flag to use specific host and port:

```
/workspace/install/bin# ./image_client -m densenet_onnx -c 3 -s INCEPTION /workspace/images/mug.jpg -u 192.168.4.138:8000
```

so like that:

```
root@kenny-System-Product-Name:/workspace/install/bin# ./image_client -m densenet_onnx -c 3 -s INCEPTION /workspace/images/mug.jpg -u 192.168.4.138:8001 -i grpc

Request 0, batch size 1

Image '/workspace/images/mug.jpg':

    15.349568 (504) = COFFEE MUG

    13.227467 (968) = CUP

    10.424896 (505) = COFFEEPOT
```

PS: http is port 8000 gRPC is 8001