ASO LAB Seminar #week 4

Triton Server - Code

Soeun Uhm

CONTENTS

01. Experiments

02. Triton Instance

- Code

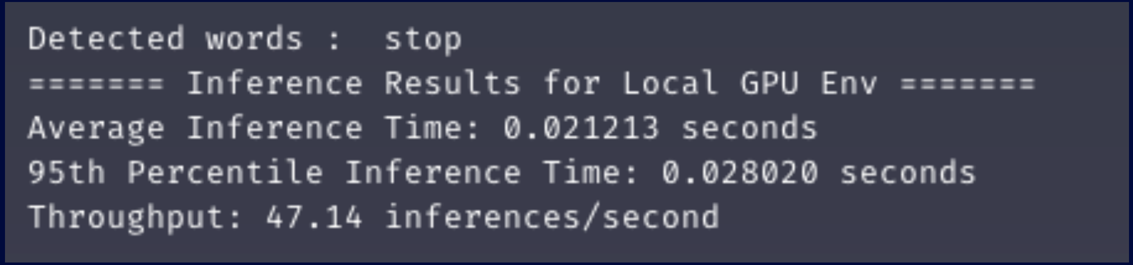
03. Optimization Analysis

- Enqueue
- Caching

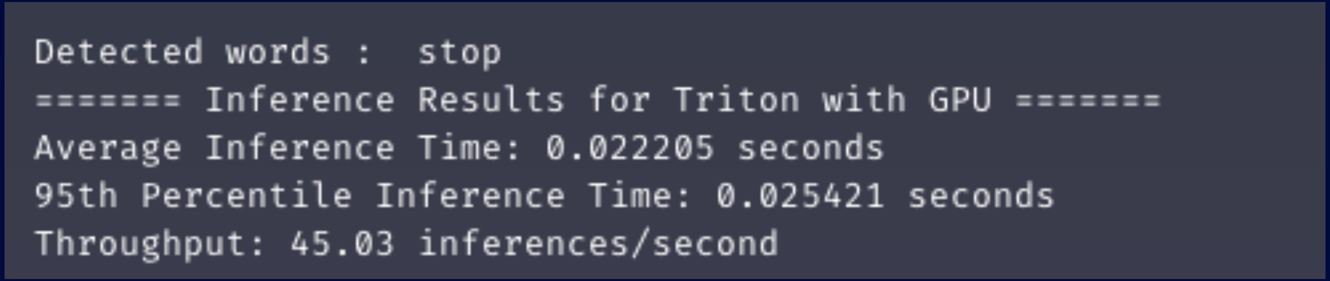
1. Experiments

OpenCV's East model(2 onnx models, using 2 GPUs)

Env.	Local Environment	Triton Server Inference
Throughput	47.14 infer/sec	45.03 infer/sec
Avg. Inference Time	0.02802 sec	0.022205 sec



```
Detected words :  stop
======= Inference Results for Local GPU Env =======
Average Inference Time: 0.021213 seconds
95th Percentile Inference Time: 0.028020 seconds
Throughput: 47.14 inferences/second
```



```
Detected words :  stop
======= Inference Results for Triton with GPU =======
Average Inference Time: 0.022205 seconds
95th Percentile Inference Time: 0.025421 seconds
Throughput: 45.03 inferences/second
```

- No dynamic batching
- No instances

1. Experiments

2 OpenCV’s East model onnx models, using 2 GPUs

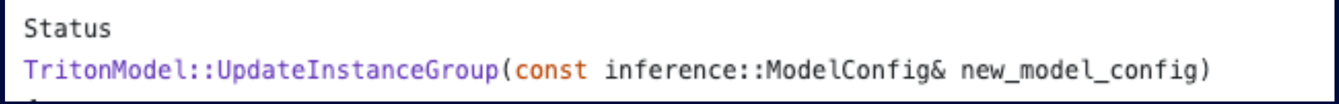
- 100 concurrent requests per one experiment
- 5 experiments

		Triton Server				
Env.	**Local Env.**	**Default**	**Dynamic batching**	**Dynamic batching & instances =1 per GPU**	**Dynamic batching & instances = 2 per GPU**	**Dynamic batching & instances = 3 per GPU**
Throughput (infer/sec)	47.14 infer/sec	45.03 infer/sec	45.76 infer/sec	43.89 infer/sec	49.03 infer/sec	43.80 infer/sec
Avg. Inference Time (sec)	0.02802 sec	0.022205 sec	0.02327 sec	0.022860 sec	0.020395 sec	0.022832 sec

2. Triton Instance

Triton Model & Instance & Request

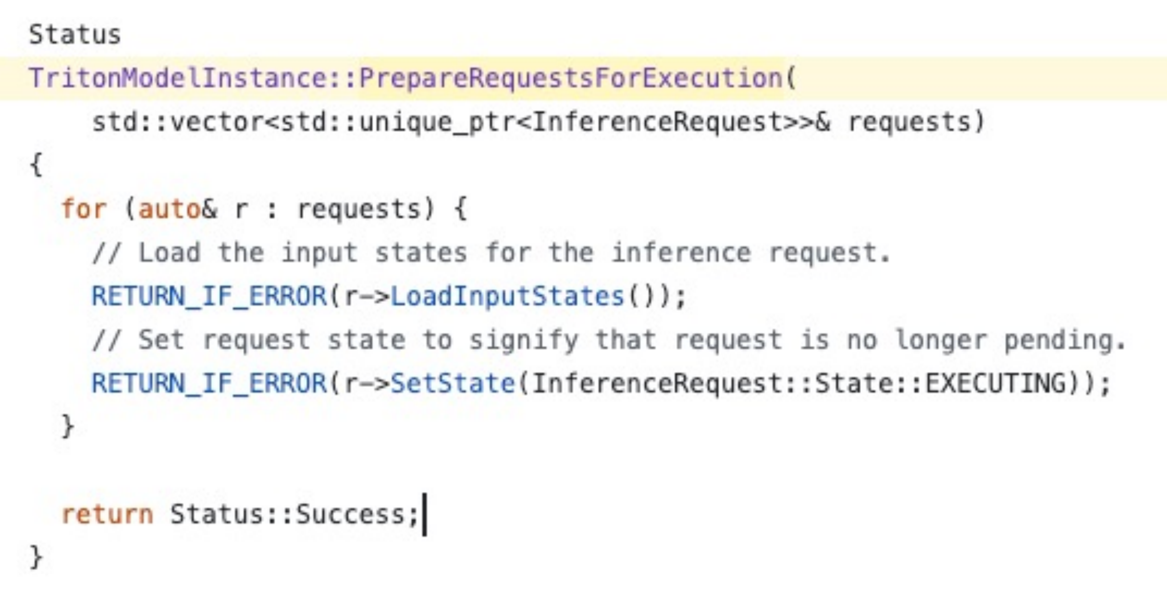
1. TritonModel::UpdateInstanceGroup



```
Status
TritonModel::UpdateInstanceGroup(const inference::ModelConfig& new_model_config)
```

- update and create model instance based on model configuration

2. TritonModelInstance::PrepareRequestsForExecution



```
Status
TritonModelInstance::PrepareRequestsForExecution(
    std::vector<std::unique_ptr<InferenceRequest>>& requests)
{
  for (auto& r : requests) {
    // Load the input states for the inference request.
    RETURN_IF_ERROR(r->LoadInputStates());
    // Set request state to signify that request is no longer pending.
    RETURN_IF_ERROR(r->SetState(InferenceRequest::State::EXECUTING));
  }

  return Status::Success;
}
```

Model Instance keeps vector of InferenceRequests

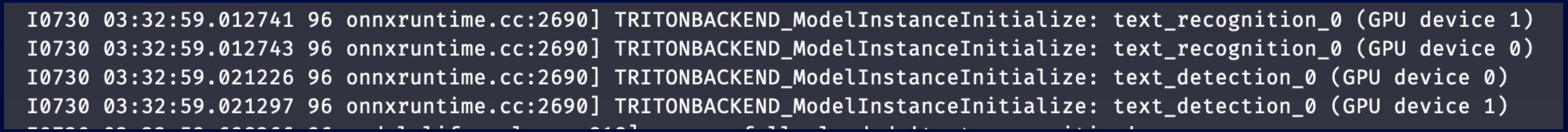
2. Triton Instance

Triton Model Instance

- No instances specified

➔ The number of instances are made according to number of GPU devices, and allocated to each GPU

➔ Optimization technique that Triton offers that users can utilize all GPU

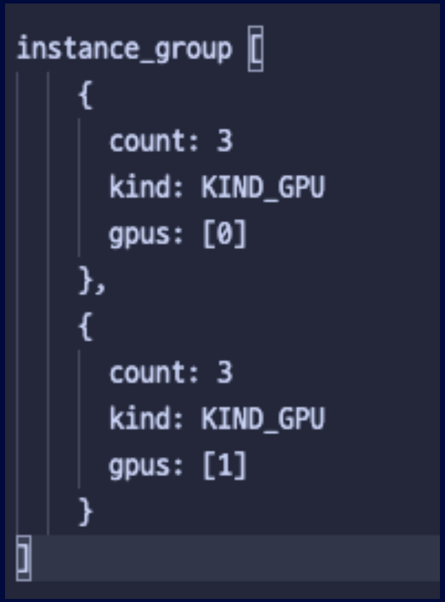


```
I0730 03:32:59.012741 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_recognition_0 (GPU device 1)
I0730 03:32:59.012743 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_recognition_0 (GPU device 0)
I0730 03:32:59.021226 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_detection_0 (GPU device 0)
I0730 03:32:59.021297 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_detection_0 (GPU device 1)
```

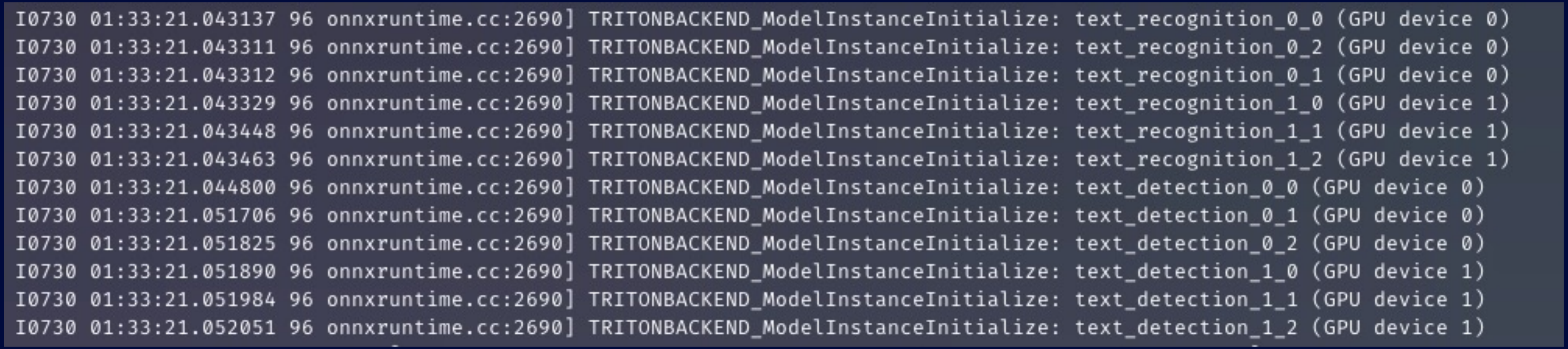
2. Triton Instance

Triton Model Instance

- Instance Group specified



```
instance_group [
    {
      count: 3
      kind: KIND_GPU
      gpus: [0]
    },
    {
      count: 3
      kind: KIND_GPU
      gpus: [1]
    }
]
```

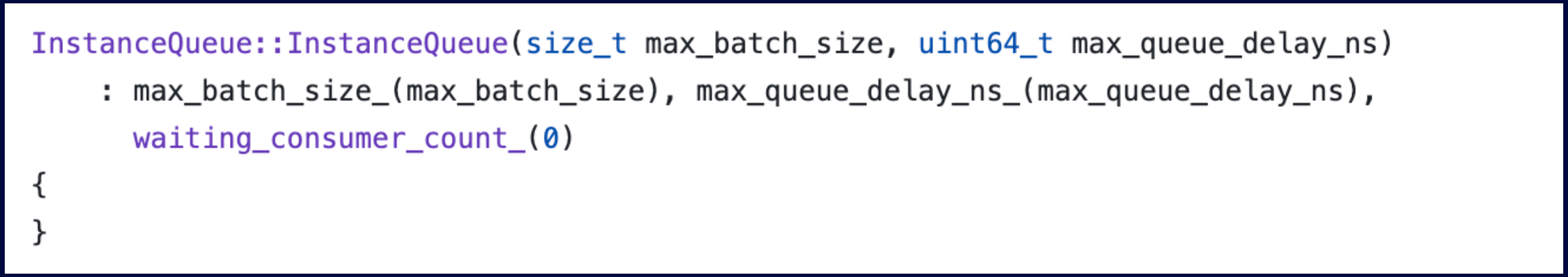


```
I0730 01:33:21.043137 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_recognition_0_0 (GPU device 0)
I0730 01:33:21.043311 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_recognition_0_2 (GPU device 0)
I0730 01:33:21.043312 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_recognition_0_1 (GPU device 0)
I0730 01:33:21.043329 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_recognition_1_0 (GPU device 1)
I0730 01:33:21.043448 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_recognition_1_1 (GPU device 1)
I0730 01:33:21.043463 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_recognition_1_2 (GPU device 1)
I0730 01:33:21.044800 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_detection_0_0 (GPU device 0)
I0730 01:33:21.051706 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_detection_0_1 (GPU device 0)
I0730 01:33:21.051825 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_detection_0_2 (GPU device 0)
I0730 01:33:21.051890 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_detection_1_0 (GPU device 1)
I0730 01:33:21.051984 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_detection_1_1 (GPU device 1)
I0730 01:33:21.052051 96 onnxruntime.cc:2690] TRITONBACKEND_ModelInstanceInitialize: text_detection_1_2 (GPU device 1)
```

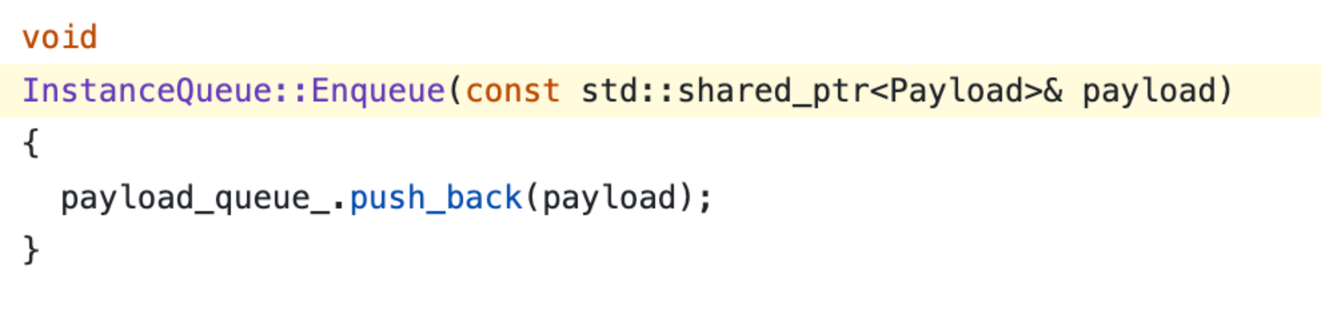
3. Optimization Analysis

Enqueue

- core/src/instance_queue.cc



```
InstanceQueue::InstanceQueue(size_t max_batch_size, uint64_t max_queue_delay_ns)
    : max_batch_size_(max_batch_size), max_queue_delay_ns_(max_queue_delay_ns),
      waiting_consumer_count_(0)
{
}
```



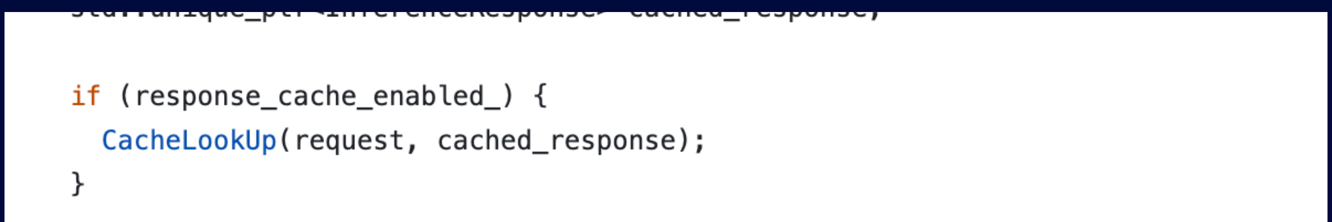
```
void
InstanceQueue::Enqueue(const std::shared_ptr<Payload>& payload)
{
  payload_queue_.push_back(payload);
}
```

- Instance Queue : specialized queue used for managing the inference requests that are waiting to be processed by a model instance
- Queues are created per instance, so that total sum of queue latency decreases
- Enqueue code is really simple… maybe do queue load balancing to optimize?

3. Optimization Analysis

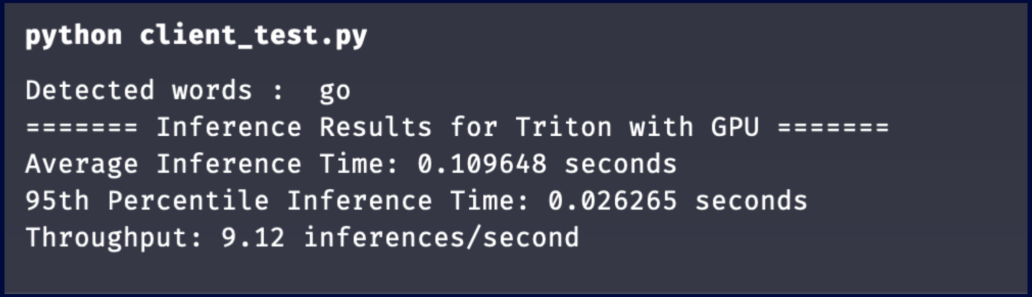
Cache Lookup

- core/src/dynamic_scheduler.cc

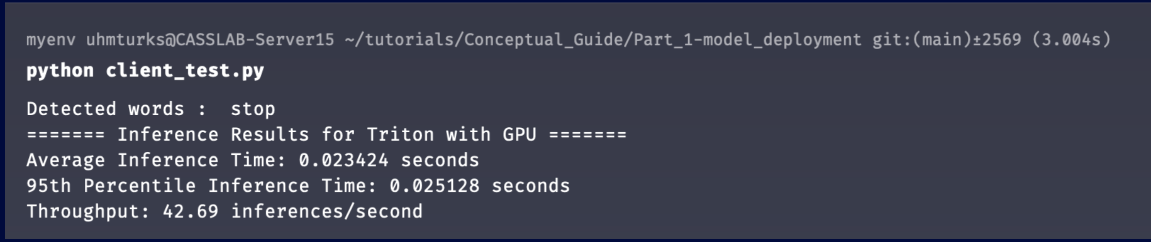


```
std::unique_ptr<InferenceResponse> cached_response;

if (response_cache_enabled_) {
  CacheLookUp(request, cached_response);
}
```



```
python client_test.py
Detected words :  go
======= Inference Results for Triton with GPU =======
Average Inference Time: 0.109648 seconds
95th Percentile Inference Time: 0.026265 seconds
Throughput: 9.12 inferences/second
```



```
myenv uhmturks@CASSLAB-Server15 ~/tutorials/Conceptual_Guide/Part_1-model_deployment git:(main)±2569 (3.004s)
python client_test.py
Detected words :  stop
======= Inference Results for Triton with GPU =======
Average Inference Time: 0.023424 seconds
95th Percentile Inference Time: 0.025128 seconds
Throughput: 42.69 inferences/second
```

- When making a inference request to Triton Server, latency for first request is long, but afterwards it is decreased (even for other inputs)
- To-do : What exactly is stored in Cache?

4. Plan

To-Do

- Code analysis for various optimization techniques(dynamic batching, queue, multiple instances)
- Run with heavier model