CS 744: NEXUS

Shivaram Venkataraman
Fall 2021
Course Project Proposals
- Due Oct 25!
- See Piazza for template

Midterm details
- Oct 28\textsuperscript{th}: Includes papers from Datacenter as a Computer to Nexus
- Open book, open notes
- Held in class time 9.30-10.45am Central Time
MACHINE LEARNING: INFERENCENumber of lines: 1
EXAMPLE APPLICATION

Video analysis service

- Thousands of streams, thousands of tenants
- Each stream is processed by a DNN-based “query”
- Latency SLOs (10s to 100s of ms)
GOAL: HIGH GPU UTILIZATION

Placement

Batching

\[ \text{batch} \_\text{lat}(b) = \alpha b + \beta, \]
SCHEDULING BATCHED EXECUTION

(b) Residual workload

<table>
<thead>
<tr>
<th>Batch</th>
<th>Lat</th>
<th>Req/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model A</td>
<td>4</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>100</td>
</tr>
<tr>
<td>Model B</td>
<td>4</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>90</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>125</td>
</tr>
<tr>
<td>Model C</td>
<td>4</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>95</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>125</td>
</tr>
</tbody>
</table>
**BATCH-AWARE SCHEDULING**

Inputs: Request rate, SLO for each model, Profiles at batch size
Approach: Allocate “full” GPUs based on load. Handle residuals

Greedy Approximation
Challenge:

How do we set latency SLOs for complex queries?
SCHEDULING COMPLEX QUERIES

Query Analysis to determine latency SLO splits
Inputs: Models with request rate $R_i$ latency SLO $L$

\[
\begin{align*}
\text{minimize} & \quad \sum_{\nu} R_{\nu} l_{\nu}(b_{\nu})/b_{\nu} \\
\text{subject to} & \quad \sum_{u:M_{\text{root}} \sim M_{\nu}} l_{u}(b_{u}) \leq L \quad \forall \nu \in \text{leaf}
\end{align*}
\]
ADAPTIVE BATCHING

Clipper: Adapt the batch size based on the oldest request
BATCH-AWARE DISPATCH

Early-dropping scheme

1. Scans queue using sliding window of batch size

2. Stop at the first request with that can execute entire window
OTHER FEATURES

Prefix Batching

GPU Multiplexing

Overlapping CPU and GPU computation
SUMMARY

• ML Inference goals: latency SLO, GPU utilization
• Nexus: Handle multiple tenants, multiple DNNs
• Schedule using squishy bin packing
• Breakdown SLO for complex queries, adaptive batching
**Pytorch Distributed**
DataParallel Training API
Overlap compute, communication

**PipeDream**
Generalize parallelism: Pipeline parallel
Reduce network, maintain consistency

**Ray**
Reinforcement learning applications
Actors and tasks, Local and global scheduler

**Pollux**
Scheduler ML training jobs in a cluster
Co-adaptive scheduling to set batch size, LR

**Nexus**
System for ML Inference
Meet latency SLOs while ensuring high utilization
DISCUSSION

https://forms.gle/XQ4CfNzTTFsSrVv7A
Consider a scenario where you have a model that takes variable amount of time depending on the input. For example if a frame contains 100 cars it takes 250ms to process but if the frame has 1 car then it finishes in 10ms. What could be one shortcoming in using Nexus to schedule this model?
Next class: SQL