TVM Streamer - Accelerating multimedia framework with TVM

Cecilia Albertsson¹
Hiroki Endoh²
Shinya Kaji¹

¹ Fixstars Corporation, ²NTT TechnoCross Corporation
Agenda

• Our Challenges
  • Business needs for a high-performance video streaming system

• TVM Streamer Overview
  • Accelerating inference for video stream processing

• Benchmark Results
  • Performance comparison for 4K and HD video streams

• Future Work
Our Challenges

• Background
  • Growing demand for intelligence video analytics.
  • NTT needs a high-performance video streaming system that can efficiently process large amounts of data such as 4K video.
  • We employ a variety of vendor-neutral and high-performance devices.

• Motivation
  • TVM has the versatility and potential to satisfy our requirements, but we know of no video streaming system with TVM that meets the above expectations.

• Proposal
  • Implement an inference application using TVM in GStreamer, a framework for multimedia processing.
TVM Streamer Overview

• TVM Streamer is implemented as a filter-type GStreamer plugin called tvminfer

• tvminfer implements image processing including inference with TVM

• tvminfer executes image processing on CPU and GPU

• Current support:
  • x86_64 and ARM64 CPUs
  • NVIDIA Jetson TX2, NVIDIA Tesla T4, and NVIDIA A100 GPUs
  • Single input layer DNN models in pre-compiled TVM format
TVM Streamer Processing

• TVM Streamer applies the following processing to images in a video stream:
  - **Preprocessing**: resizing, batching etc.
  - **Inference**: loads and runs pre-compiled model in TVM
  - **Postprocessing**: can be anything, supplied as a function that receives an image and the associated inference results

• Parameters for preprocessing and inference may be tweaked via properties passed to tvminfer
TVM Streamer Structure

- TVM Streamer employs a system of queues to pass images between processing stages
- Each processing stage runs in a separate thread
TVM Streamer Concurrency

- TVM Streamer exploits CPU-side multithreading and GPU-side CUDA streams to optimize **concurrency**
  - Processing of each image overlaps with that of the previous image, saving time
Benchmark Results

• We compared performance results of TVM Streamer to those of DL Streamer (for CPU) and DeepStream SDK (for GPU)
  • DL Streamer: inference with Intel OpenVINO
  • DeepStream SDK: inference with NVIDIA TensorRT
• We used GStreamer pipelines that reproduce, as closely as possible, the same processing for each framework
• We measured latency, throughput, and power efficiency
• We used AutoTVM to tune models for the TVM Streamer benchmarks

• Result:
  **TVM Streamer exhibits significantly higher performance** than DeepStream SDK in some cases
Benchmarks x86_64 CPU

- Comparison between TVM Streamer and DL Streamer on x86_64 CPU

<table>
<thead>
<tr>
<th>Model</th>
<th>Resolution</th>
<th>Latency (msec)</th>
<th>Throughput (FPS)</th>
<th>Power efficiency (FPS/average Watts)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>TVM Streamer</td>
<td>DL Streamer</td>
<td>TVM Streamer</td>
</tr>
<tr>
<td>mobilnetv3_large</td>
<td>4K</td>
<td>9.33</td>
<td>10.41</td>
<td>107.24</td>
</tr>
<tr>
<td></td>
<td>HD</td>
<td>1.98</td>
<td>1.19</td>
<td>506.29</td>
</tr>
<tr>
<td>yolo3_darknet53_coco</td>
<td>4K</td>
<td>57.70</td>
<td>20.27</td>
<td>17.33</td>
</tr>
<tr>
<td></td>
<td>HD</td>
<td>57.96</td>
<td>18.60</td>
<td>17.25</td>
</tr>
</tbody>
</table>
### Benchmarks NVIDIA Tesla T4

**Comparison between TVM Streamer and DeepStream SDK on NVIDIA Tesla T4**

<table>
<thead>
<tr>
<th>Model</th>
<th>Resolution</th>
<th>Latency (msec)</th>
<th>Throughput (FPS)</th>
<th>Power efficiency (FPS/average Watts)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>TVM Streamer</td>
<td>DeepStream SDK</td>
<td>TVM Streamer</td>
</tr>
<tr>
<td><code>mobilenetv3_large</code> (224x224)</td>
<td>4K</td>
<td>5.60</td>
<td>11.36</td>
<td>178.55</td>
</tr>
<tr>
<td></td>
<td>HD</td>
<td>0.98</td>
<td>1.75</td>
<td>1016.42</td>
</tr>
<tr>
<td><code>yolo3_darknet53_coco</code> (416x416)</td>
<td>4K</td>
<td>19.99</td>
<td>18.72</td>
<td>50.03</td>
</tr>
<tr>
<td></td>
<td>HD</td>
<td>20.24</td>
<td>19.18</td>
<td>49.41</td>
</tr>
</tbody>
</table>
Benchmarks NVIDIA Jetson TX2

• Comparison between TVM Streamer and DeepStream SDK on NVIDIA Jetson TX2
  • We did not measure power efficiency on Jetson TX2

<table>
<thead>
<tr>
<th>Model</th>
<th>Resolution</th>
<th>Latency (msec)</th>
<th>Throughput (FPS)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>TVM Streamer</td>
<td>DeepStream SDK</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mobilenetv3_large</td>
<td>4K (†1)</td>
<td>20.30</td>
<td>N/A</td>
</tr>
<tr>
<td>(224x224)</td>
<td>HD</td>
<td>7.18</td>
<td>7.81</td>
</tr>
<tr>
<td>yolo3_darknet53_coco</td>
<td>4K</td>
<td>225.64</td>
<td>175.15</td>
</tr>
<tr>
<td>(416x416)</td>
<td>HD</td>
<td>222.41</td>
<td>166.86</td>
</tr>
</tbody>
</table>

*1: DeepStream SDK does not support input/model resolution ratios in excess of a factor of 16 on NVIDIA Jetson TX2
Future Work

• Additional benchmarks
• Support for edge devices
  • E.g., Google TPU, Qualcomm Snapdragon
• Adding useful functions related to inference processing
Thank you!