QUANTPIPE: APPLYING ADAPTIVE POST-TRAINING QUANTIZATION FOR DISTRIBUTED TRANSFORMER PIPELINES IN DYNAMIC EDGE ENVIRONMENTS

Haonan Wang¹ ², Connor Imes², Souvik Kundu¹ ³, Peter A. Beerel¹ ², Stephen P. Crago¹ ², and John Paul Walters²

¹University of Southern California, Los Angeles, CA, USA
²USC Information Sciences Institute, Arlington, VA, USA
³Intel Labs, San Diego, CA, USA
BACKGROUND AND MOTIVATION

- AI faces a growing problem in the Transformer era
- High demands on local inference
  - E.g., run ChatGPT in your house
  - Concerns of privacy or connectivity to Internet
- Pushing these increasingly large models to the edge adds additional challenges
  - Resources/power constrains of Edge devices
- Pipeline parallelism can be employed to parallelize large-scale transformer models across devices

![Fig. Illustration of the pipeline parallelism paradigm](image-url)
BACKGROUND AND MOTIVATION

Property of the Edge System:

- **Unstable Connection**
  - Network fluctuation
  - Signal blocking
  - Movement
- **Low-speed protocols**
  - Bluetooth
  - LP-WAN
  - ...

Timeline:

- Input Batch 1
  - Comm. 1
  - Compute 1
- Input Batch 2
  - Comm. 2
  - Compute 2
  - **Bottleneck**
- Input Batch 3
  - Comm. 2
  - Compute 2

Batched streaming:

Node 1 → Node 2 → Node 3

Generate data → Batched streaming → Outputs
Q: How to compress communication?

A: Post-training Quantization (PTQ).

Original Distribution (32bit float) vs. Distribution after PTQ (4-bit, 3-bit, and 2-bit integer)
ADAPTIVE POST-TRAINING QUANTIZATION

Challenges of applying PTQ:
- Where to do PTQ?
- How to do PTQ?
- What is the accuracy loss?

Property of PTQ:
- We insert PTQ only at the boundary of the pipeline where the model is partitioned, to lower the impact of quantization
- Experimental results show PTQ is suitable for the PipeEdge System
ADAPTIVE POST-TRAINING QUANTIZATION

How to do PTQ?

• Analytical Clipping for Integer Quantization (ACIQ) [1]
  ➢ A PTQ method for CNN models
  ➢ Clip the outliers to significantly improve accuracy
  ➢ Decide the best clip range that minimizes the mean square error (MSE)

• Applying PTQ to Visual Transformer (ViT) models
  ➢ Two types of activation distribution
  ➢ Mismatch of distribution estimation for real data


Fig. Distribution of the original data (top), after naive PTQ (middle), or after PTQ with ACIQ (bottom) from the ViT-Base model partitioned after 4th (left) and 6th (right) block.
Directed-search ACIQ (DS-ACIQ):

- For better estimation of the data distribution
- Search direction is determined by the peak of histogram curve
- Further decrease the MSE by ~50%
- Only incur < 1% computation overhead
- Accuracy of PTQ w/ DS-ACIQ (PDA):

**Table 1:** Average ViT-Base model accuracy with ImageNet.

<table>
<thead>
<tr>
<th></th>
<th>32bit</th>
<th>16bit</th>
<th>8bit</th>
<th>6bit</th>
<th>4bit</th>
<th>2bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTQ</td>
<td>80.26%</td>
<td>75.74%</td>
<td>43.03%</td>
<td>30.29%</td>
<td>0.44%</td>
<td></td>
</tr>
<tr>
<td>ACIQ</td>
<td>80.03%</td>
<td>79.35%</td>
<td>78.87%</td>
<td>76.46%</td>
<td>54.97%</td>
<td></td>
</tr>
<tr>
<td>PDA</td>
<td>78.94%</td>
<td>78.72%</td>
<td>78.21%</td>
<td>77.34%</td>
<td>70.82%</td>
<td></td>
</tr>
</tbody>
</table>

Fig. Comparison of ACIQ and DS-ACIQ
Adaptive PTQ with DS-ACIQ (Adaptive PDA):

- Implement PDA in our QuantPipe system:
  - monitor the output bandwidth $B_{k,t}$ of stage $k$ at inference iteration $t$
  - estimate the bitwidth $q_{k,t+1}$ required to achieve the target throughput $R$

$$q_{k,t+1} = \frac{32}{2} \log\left(\frac{V_{k,t} \times 32 / q_{k,t}}{S / R \times B_{k,t}}\right)$$

where $V_{k,t}$ represents the volume of quantized data under $q_{k,t}$

and $S$ denotes the microbatch size

- In the real implementation:
  - QuantPipe monitors the bandwidth every 50 batches
  - Switch quantization bitwidth at runtime to recover system performance
Experimental Settings:

• Hardware Testbed:
  - An Edge cluster with 6 NVIDIA Jetson AGX Orin devices
  - Each device has a 12-core ARM CPU, a 1792-core GPU, and runs Linux kernel 5.10.65-tegra.
  - 1Gbps Ethernet connection between devices

• Software:
  - We implement our QuantPipe on top of the PipeEdge, a distributed edge computing framework\textsuperscript{[2]}
  - using Python 3.8 and PyTorch 1.12.

• Bandwidth Control:
  - We simulate the network fluctuation using Linux traffic control tools (tc).

• Deep Learning Model:
  - Visual Transformer (ViT)

Experimental results: Adaptiveness

Stage Bandwidth

Quantization Bitwidth

Stage Performance

Model Accuracy

target output rate
Experimental results: Adaptiveness

**Stage Bandwidth**

- Do bandwidth control to 400 Mbps

**Quantization Bitwidth**

- 32 bit

**Stage Performance**

- Target output rate

**Model Accuracy**

- Top-1 correct %
Experimental results: Adaptiveness

Stage Bandwidth
- Set bandwidth to 400 Mbps
- Set bandwidth to 50 Mbps

Stage Performance
- 32 to 16 bit bring about 2X improvement to achieve performance target
- Target output rate

Quantization Bitwidth
- Check bandwidth condition every 50 batches
- Perform bitwidth switching: 32 -> 16 bit

Model Accuracy
- No accuracy loss under 16-bit PTQ
Experimental results: Adaptiveness

Stage Bandwidth
- Set bandwidth to 50 Mbps
- Set bandwidth to 200 Mbps

Quantization Bitwidth
- Check bandwidth condition every 50 batches
- Perform bitwidth switching: 16 -> 2 bit

Stage Performance
- 16 to 2 bit bring about 8X improvement to achieve performance target

Model Accuracy
- Around 5% acc loss under 2-bit PTQ
Experimental results: Adaptiveness

### Stage Bandwidth

- **Set bandwidth to 200 Mbps**

### Quantization Bitwidth

- **Perform bitwidth switching:** 2 -> 6 -> 8 bit

### Stage Performance

- **To not waste performance, choose bitwidth closest to target rate**
- **Target output rate**

### Model Accuracy

- **Accuracy recover back to no loss under 8-bit PTQ**

---

Information Sciences Institute

14

USC Viterbi
School of Engineering
Experimental results: Adaptiveness

Stage Bandwidth

Quantization Bitwidth

Stage Performance

Model Accuracy
Demo Devices
Thank You

Any further questions are welcome
please contact: haonan.wang@usc.edu