Ascend CANN and ONNX: inference interoperability for better performance
Memory Lane - Huawei's Participation In ONNX
ONNX Edge Working Group

This is the artifact repository where ONNX Edge working group will capture various artifacts and deliverables. Structure of the space will evolve over time.

Working Group Status

ACTIVE

Contributors

Note: Contributors list will be updated as per participation and contributions.

- Milan Oljea (Qualcomm) (co-chair)
- Ofer Rosenberg (Qualcomm) (co-chair)
- Yedong Liu (Huawei)
- Saurabh Tangri (Intel)
- Manash Goswami (Microsoft)

MindSpore ONNX Exporter Introduction

1. Use MindSpore model train API to perform model training with saving checkpoint parameters
2. Load model parameters into the network to be exported (such like LeNet)
3. Call train.export() to convert MindSpore model to ONNX model
4. Perform model inference on ONNX Runtime

The black box is the "profiling phase" and the orange box is the "execution phase"
AI Heterogeneous Computing Architecture: CANN 5.0
(Compute Architecture for Neural Networks)

<table>
<thead>
<tr>
<th>Inference Application</th>
<th>Training Application</th>
<th>MindSpore</th>
<th>TensorFlow</th>
<th>MindStudio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ACL: Ascend Compute Language Development API
(Ops Dev API (TBE) /Graph Dev API (AIR) /App Dev API)

Ascend Computing Service Layer

<table>
<thead>
<tr>
<th>Ascend Ops Library (AOL)</th>
<th>NN Lib</th>
<th>...</th>
<th>Ascend Optimization Engine (AOE)</th>
<th>OPAT</th>
<th>AMCT</th>
<th>SGAT</th>
<th>GDAT</th>
<th>Framework Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ascend Computing Compilation Layer

<table>
<thead>
<tr>
<th>Ascend Tensor Compiler (ATC)</th>
<th>Graph Compiler</th>
<th>TBE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ascend Computing Execution Layer

<table>
<thead>
<tr>
<th>ACE</th>
<th>RunTime</th>
<th>Graph Executor</th>
<th>DVPP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HCCL</td>
<td>AIPP</td>
<td>......</td>
</tr>
</tbody>
</table>

ABL

| ABL | SVM | VM | HDC | ...
|-----|-----|----|-----|-----|

Ascend

<table>
<thead>
<tr>
<th>Ascend</th>
<th>Ascend-310</th>
<th>Ascend-710</th>
<th>Ascend-910</th>
<th>IP &amp; Chip</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

What Is CANN

CANN is an AI Heterogeneous Computing Architecture which supports users to quickly develop AI applications on Ascend hardware platform via providing multiple layer of programming interfaces.

Key Features

- **Unified Application Programming Interface**: ACL as the standardized programming interface which abstract underlying hardwares.
- **Unified Neural Network Graph Construction Interface**: AIR as the standarded graph construction interface which supports multiple frameworks.
- **High Performance Compute Engine and Operator Library**
- **Basic Service**: capabilities include drivers, virtualization, media, communications, etc.
CANN 5.0 and onnx: accelerating inference model on Ascend

- Currently support 140+ onnx inference models, will reach to 200+ by the end of the year
- Support opset 8~13 with opset 11 as the key set, 90+% of the Ops will be supported on CANN by the end of the year
Future Thoughts

Pain points need to be addressed in the community:

1. PyTorch NLP and Audio models' export to ONNX is still very difficult, training model's export to ONNX is also difficult for developers:
   - Trace doesn't support loop and if;  
   - torchscript is underutilized in the process of exporting to ONNX  
   - There is a 35%~40% failure rate when exporting pytorch model to onnx

2. The iteration of Opset is very fast which creates difficulties for hardware engineers to do the adaption work