Using ONNX for accelerated inferencing on cloud and edge

Prasanth Pulavarthi (Microsoft)
Kevin Chen (NVIDIA)
Agenda

- What is ONNX
- How to create ONNX models
- How to operationalize ONNX models (and accelerate with TensorRT)
Open and Interoperable AI
Open Neural Network Exchange

Open format for ML models

github.com/onnx
Key Design Principles

• Support DNN but also allow for traditional ML
• Flexible enough to keep up with rapid advances
• Compact and cross-platform representation for serialization
• Standardized list of well defined operators informed by real world usage
ONNX Spec

- File format
- Operators
File format

Model
• Version info
• Metadata
• Acyclic computation dataflow graph

Graph
• Inputs and outputs
• List of computation nodes
• Graph name

Computation Node
• Zero or more inputs of defined types
• One or more outputs of defined types
• Operator
• Operator parameters
Data types

- **Tensor type**
  - Element types supported:
    - int8, int16, int32, int64
    - uint8, uint16, uint32, uint64
    - float16, float, double
    - bool
    - string
    - complex64, complex128

- **Non-tensor types in ONNX-ML:**
  - Sequence
  - Map

```protobuf
def message TypeProto {
  message Tensor {
    optional TensorProto.DataType elem_type = 1;
    optional TensorShapeProto shape = 2;
  }
  // repeated T
  message Sequence {
    optional TypeProto elem_type = 1;
  }
  // map<K,V>
  message Map {
    optional TensorProto.DataType key_type = 1;
    optional TypeProto value_type = 2;
  }
  oneof value {
    Tensor tensor_type = 1;
    Sequence sequence_type = 4;
    Map map_type = 5;
  }
}
```
Operators

An operator is identified by <name, domain, version>

Core ops (ONNX and ONNX-ML)
- Should be supported by ONNX-compatible products
- Generally cannot be meaningfully further decomposed
- Currently 124 ops in ai.onnx domain and 18 in ai.onnx.ml
- Supports many scenarios/problem areas including image classification, recommendation, natural language processing, etc.

Custom ops
- Ops specific to framework or runtime
- Indicated by a custom domain name
- Primarily meant to be a safety-valve
Functions

• Compound ops built with existing primitive ops

• Runtimes/frameworks/tools can either have an optimized implementation or fallback to using the primitive ops
ONNX is a Community Project

Get Involved

Discuss
Participate in discussions for advancing the ONNX spec.

Contribute
Make an impact by contributing feedback, ideas, and code.

gitter.im/onnx
github.com/onnx
ML @ Microsoft

- LOTS of internal teams and external customers
- LOTS of models from LOTS of different frameworks
  - Cognitive Toolkit
  - PyTorch
  - Caffe
- Different teams/customers deploy to different targets
  - CPU
  - GPU
  - FPGA
Open and Interoperable AI

ONNX

PyTorch
Chainer
Caffe2
mxnet
Cognitive Toolkit
XGBoost
PaddlePaddle
ONNX @ Microsoft

• ONNX in the platform
  • Windows
  • ML.net
  • Azure ML

• ONNX model powered scenarios
  • Bing
  • Ads
  • Office
  • Cognitive Services
  • more
Bing QnA - List QnA and Segment QnA
- Two models used for generating answers
- Up to 2.8x perf improvement with ONNX Runtime

Query: empire earth similar games

Games Like Empire Earth
- Total War: Arena.
- Stronghold Kingdoms.
- Rise of Nations.
- Age of Empires 3.
- ... (more items)

19 Games Like Empire Earth - Games Finder
gameslikefinder.com/games-like-empire-earth/
ONNX @ Microsoft

Bing Multimedia - Semantic Precise Image Search
- Image Embedding Model - Project image contents into feature vectors for image semantic understanding
- 1.8x perf gain by using ONNX and ONNX Runtime

Query: newspaper printouts to fill in for kids

Image Embedding Model
- Original framework
- ONNX Runtime
ONNX @ Microsoft

• Teams are organically adopting ONNX and ONNX Runtime for their models – cloud & edge

• Latest 50 models converted to ONNX showed average $2\times$ perf gains on CPU with ONNX Runtime
Agenda

✓ What is ONNX

☐ How to create ONNX models

☐ How to operationalize ONNX models
4 ways to get an ONNX model

- ONNX Model Zoo
- Services like Azure Custom Vision
- Convert existing models
- Train models in systems like Azure Machine Learning service
ONNX Model Zoo: [github.com/onnx/models](https://github.com/onnx/models)

### Image Classification

This collection of models take images as input, then classifies the major objects in the images into a set of predefined classes.

<table>
<thead>
<tr>
<th>Model Class</th>
<th>Reference</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MobileNet</td>
<td>Sandler et al.</td>
<td>Efficient CNN model for mobile and embedded vision applications. Top-5 error from paper - ~10%</td>
</tr>
<tr>
<td>ResNet</td>
<td>He et al., He et al.</td>
<td>Very deep CNN model (up to 152 layers), won the ImageNet Challenge in 2015. Top-1 accuracy: 75.81%</td>
</tr>
<tr>
<td>SqueezeNet</td>
<td>Iandola et al.</td>
<td>A lightweight and efficient CNN that uses depthwise separable convolutions. Top-5 accuracy: 89.49%</td>
</tr>
<tr>
<td>VGG</td>
<td>Simonyan et al.</td>
<td>Deep convolutional neural network for image classification. Top-5 accuracy: 94.21%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>Download</th>
<th>Checksum</th>
<th>Download (with sample test data)</th>
<th>ONNX version</th>
<th>Opset version</th>
<th>Top-1 accuracy (%)</th>
<th>Top-5 accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-18</td>
<td>44.6 MB</td>
<td>MD5</td>
<td>42.9 MB</td>
<td>1.2.1</td>
<td>7</td>
<td>69.70</td>
<td>89.49</td>
</tr>
<tr>
<td>ResNet-34</td>
<td>83.2 MB</td>
<td>MD5</td>
<td>78.6 MB</td>
<td>1.2.1</td>
<td>7</td>
<td>73.36</td>
<td>91.43</td>
</tr>
<tr>
<td>ResNet-50</td>
<td>97.7 MB</td>
<td>MD5</td>
<td>92.0 MB</td>
<td>1.2.1</td>
<td>7</td>
<td>75.81</td>
<td>92.82</td>
</tr>
<tr>
<td>ResNet-101</td>
<td>170.4 MB</td>
<td>MD5</td>
<td>159.4 MB</td>
<td>1.2.1</td>
<td>7</td>
<td>77.42</td>
<td>93.61</td>
</tr>
<tr>
<td>ResNet-152</td>
<td>230.3 MB</td>
<td>MD5</td>
<td>216.0 MB</td>
<td>1.2.1</td>
<td>7</td>
<td>78.20</td>
<td>94.21</td>
</tr>
</tbody>
</table>
Custom Vision Service: customvision.ai

1. Upload photos and label

2. Train

3. Download ONNX model!
Convert models

- Caffe2
- Chainer
- Cognitive Toolkit
- mxnet
- PyTorch
- PaddlePaddle
- ML.NET
- MathWorks
- XGBoost
- scikit-learn
- TensorFlow
- Keras
from keras.models import load_model
import keras2onnx
import onnx

keras_model = load_model("model.h5")

onnx_model = keras2onnx.convert_keras(keras_model, keras_model.name)

onnx.save_model(onnx_model, 'model.onnx')
import numpy as np
import chainer
from chainer import serializers
import onnx_chainer

serializers.load_npz("my.model", model)

sample_input = np.zeros((1, 3, 224, 224), dtype=np.float32)
chainer.config.train = False

onnx_chainer.export(model, sample_input, filename="my.onnx")
import torch
import torch.onnx

model = torch.load("model.pt")

sample_input = torch.randn(1, 3, 224, 224)

torch.onnx.export(model, sample_input, "model.onnx")
Convert models: TensorFlow

Convert TensorFlow models from

- Graphdef file
- Checkpoint
- Saved model

```python
python -m tf2onnx.convert
    --input SOURCE_GRAPHDEF_PBB
    --graphdef SOURCE_GRAPHDEF_PBB
    --checkpoint SOURCE_CHECKPOINT
    --saved-model SOURCE_SAVED_MODEL
    [--inputs GRAPH_INPUTS]
    [--outputs GRAPH_OUTPUTS]
    [--inputs-as-nchw inputs_provided_as_nchw]
    [--target TARGET]
    [--output TARGET_ONNX_GRAPH]
    [--target TARGET]
    [--continue_on_error]
    [--verbose]
    [--custom-ops list-of-custom-ops]
    [--opset OPSET]
    [--fold_const]
```
ONNX-Ecosystem Container Image

- Quickly get started with ONNX
- Supports converting from most common frameworks
- Jupyter notebooks with example code
- Includes ONNX Runtime for inference

```bash
docker pull onnx/onnx-ecosystem
docker run -p 8888:8888 onnx/onnx-ecosystem
```

- TensorFlow
- Keras
- PyTorch
- MXNet
- SciKit-Learn
- LightGBM
- CNTK
- Caffe (v1)
- CoreML
- XGBoost
- LibSVM
Demo
BERT model using onnx-ecosystem container image
Agenda

✓ What is ONNX

✓ How to create ONNX models

☐ How to operationalize ONNX models
Demo

Style transfer in a Windows app
❖ High performance
❖ Cross platform
❖ Lightweight & modular
❖ Extensible
ONNX Runtime

- High performance runtime for ONNX models
- Supports full ONNX-ML spec (v1.2 and higher, currently up to 1.4)
- Works on Mac, Windows, Linux (ARM too)
- Extensible architecture to plug-in optimizers and hardware accelerators
- CPU and GPU support
- Python, C#, and C APIs
```python
import onnxruntime

session = onnxruntime.InferenceSession("mymodel.onnx")

results = session.run([], {"input": input_data})
```
using Microsoft.ML.OnnxRuntime;

var session = new InferenceSession("model.onnx");

var results = session.Run(input);
ONNX Runtime – C API

```c
#include <core/session/onnxruntime_c_api.h>

// Variables
OrtEnv* env;
OrtSession* session;
OrtAllocatorInfo* allocator_info;
OrtValue* input_tensor = NULL;
OrtValue* output_tensor = NULL;

// Scoring run
OrtCreateEnv(ORT_LOGGING_LEVEL_WARNING, "test", &env)
OrtCreateSession(env, "model.onnx", session_options, &session)
OrtCreateCpuAllocatorInfo(OrtArenaAllocator, OrtMemTypeDefault, &allocator_info)
OrtCreateTensorWithDataAsOrtValue(allocator_info, input_data, input_count * sizeof(float), input_dim_values, num_dims, ONNX_TENSOR_ELEMENT_DATA_TYPE_FLOAT, &input_tensor)
OrtRun(session, NULL, input_names, (const OrtValue* const*)&input_tensor, num_inputs, output_names, num_outputs, &output_tensor));
OrtGetTensorMutableData(output_tensor, (void **) &float_array);

//Release objects
...
Demo

Action detection in videos

Evaluation videos from:
Sports Videos in the Wild (SVW): A Video Dataset for Sports Analysis
Safdarnejad, S. Morteza and Liu, Xiaoming and Udpa, Lalita and Andrus, Brooks and Wood, John and Craven, Dean
Demo

Convert and deploy object detection model as Azure ML web service
ONNX Model

Graph

Partitioner

Provider Registry

Execution Providers
- CPU
- MKL-DNN
- nGraph
- CUDA
- TensorRT
- ...

Parallel, Distributed Graph Runner

Input Data

Output Result
Industry Support for ONNX Runtime
ONNX Runtime + TensorRT

- Now released as preview!
- Run any ONNX-ML model
- Same cross-platform API for CPU, GPU, etc.
- ONNX Runtime partitions the graph and uses TensorRT where support is available
**NVIDIA TensorRT**

Platform for High-Performance Deep Learning Inference

Optimize and deploy neural networks in production environments

Maximize throughput for latency-critical apps with optimizer and runtime

Optimize your network with layer and tensor fusions, dynamic tensor memory and kernel auto tuning

Deploy responsive and memory efficient apps with INT8 & FP16 optimizations

Fully integrated as a backend in ONNX runtime

developer.nvidia.com/tensorrt
ONNX-TensorRT Parser
Available at https://github.com/onnx/onnx-tensorrt

ONNX-TensorRT Ecosystem

- **Public APIs**
  - C++
  - Python

- **Supported Platforms**
  - Desktop
  - Embedded Linux

- **Upcoming Support**
  - Windows
  - CentOS
  - IBM PowerPC
TensorRT Execution Provider in ONNX Runtime

ONNX Model

In-Memory Graph

Graph Partitioner

Provider Registry

Parallel, Distributed Graph Runner

Execution Providers

CPU

MKL-DNN

nGraph

CUDA

TensorRT

Input Data

Output Result
Parallel, Distributed Graph Runner

Full or Partitioned ONNX Graph

ONNX-TensorRT Parser

INetwork Object

TensorRT Core Libraries

IEngine Object

Runtime

High-Speed Inference

Output Results
Demo

Comparing backend performance on emotion_ferplus ONNX zoo model
## Demo performance comparison

<table>
<thead>
<tr>
<th>ONNXRUNTIME-CPU</th>
<th>ONNXRUNTIME-GPU (using CUDA)</th>
<th>ONNXRUNTIME-TensorRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model prediction: surprise</td>
<td>Model prediction: surprise</td>
<td>Model prediction: surprise</td>
</tr>
<tr>
<td>Inference time: 61.03 ms</td>
<td>Inference time: 3.63 ms</td>
<td>Inference time: 2.47 ms</td>
</tr>
<tr>
<td>Model Input image: <img src="image1.jpg" alt="Image" /></td>
<td>Model Input image: <img src="image2.jpg" alt="Image" /></td>
<td>Model Input image: <img src="image3.jpg" alt="Image" /></td>
</tr>
</tbody>
</table>

Model: Facial Expression Recognition (FER+) model from ONNX model zoo
Hardware: Azure VM – NC12 (K80 NVIDIA GPU)
CUDA 10.0, TensorRT 5.0.2
ONNX Runtime + TensorRT @ Microsoft

Bing Multimedia team seeing 2X perf gains
ONNX Runtime + TensorRT

- Best of both worlds
- Run any ONNX-ML model
- Easy to use API across platforms and accelerators
- Leverage TensorRT acceleration where beneficial
Recap

✓ What is ONNX
ONNX is an open standard so you can use the right tools for the job and be confident your models will run efficiently on your target platforms

✓ How to create ONNX models
ONNX models can be created from many frameworks – use onnx-ecosystem container image to get started quickly

✓ How to operationalize ONNX models
ONNX models can be deployed to the edge and the cloud with the high performance, cross platform ONNX Runtime and accelerated using TensorRT
Try it for yourself

Available now with TensorRT integration preview!
Instructions at aka.ms/onnxruntime-tensorrt

Open sourced at github.com/microsoft/onnxruntime