OPEN DATA SCIENCE CONFERENCE

London | Nov. 19 - Nov. 22 2019

FROM RESEARCH TO PRODUCTION:

performant cross-platform ML/DNN model inferencing on cloud and edge with ONNX Runtime

FAITH XU| Senior Program Manager @ Microsoft

PRABHAT ROY| Data and Applied Scientist @ Microsoft

Agenda – What we’ll cover today

- **INTRODUCTION**
- **PART A:** Train an image classification DNN model with PyTorch
- **PART B:** Inference the model locally using ONNX Runtime
- **PART C:** Deploy the model for inferencing using Azure Machine Learning

State of AI

Trends and Growth Areas

Research -> Industry

- Automated Machine Learning services
- Startups – applied AI
- Hosted services for cloud compute
- Hardware investments

Connectivity, compute, and resources

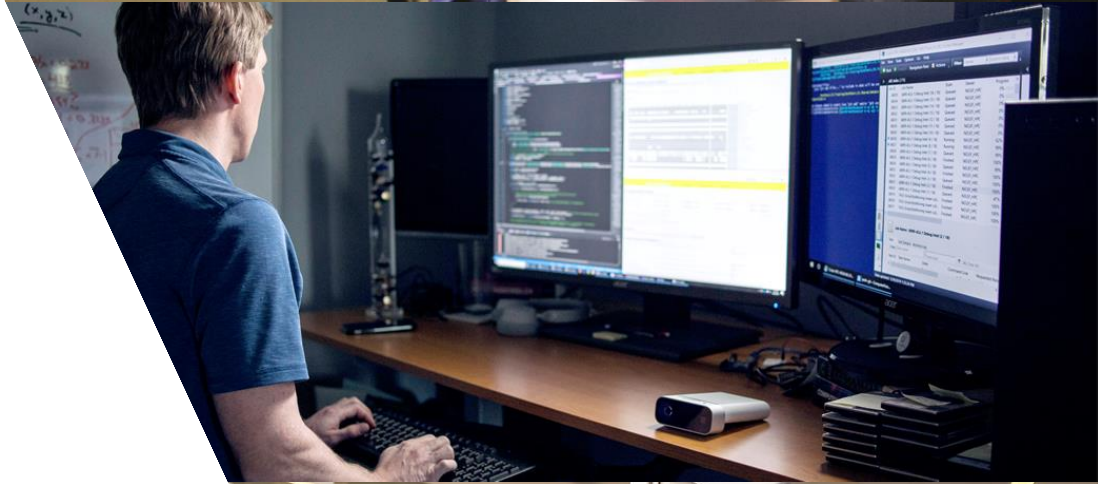
- Infinite storage and compute in the cloud
- CPU, GPUs for training
- LOTS of data

Application spans across all industries

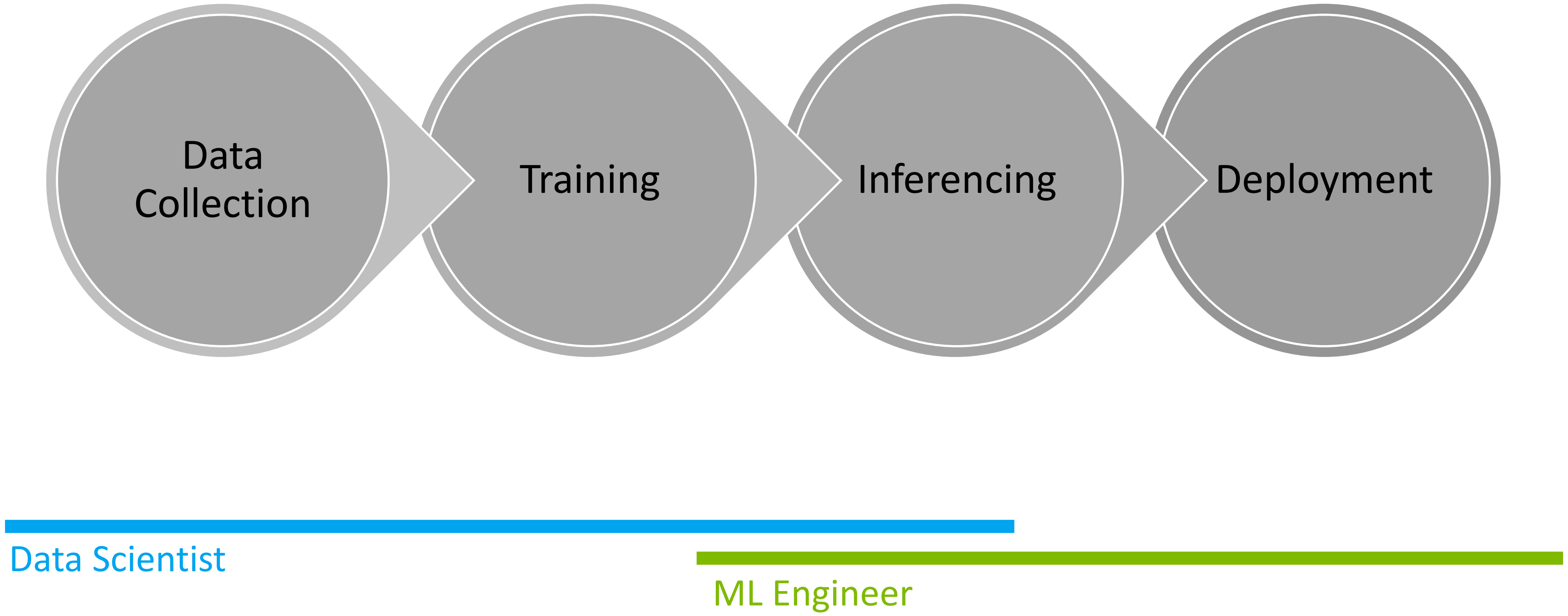
- Healthcare, farming, gaming, manufacturing, consumer products, and more

Investments in AI education and jobs

- Universities
- ML Engineer



ML Models: Research to Production



Data Collection

Training

Inferencing

Deployment

Data Scientist

ML Engineer

Product teams want to incorporate ML

Microsoft 365

Windows

Microsoft Dynamics 365

Skype

Bing

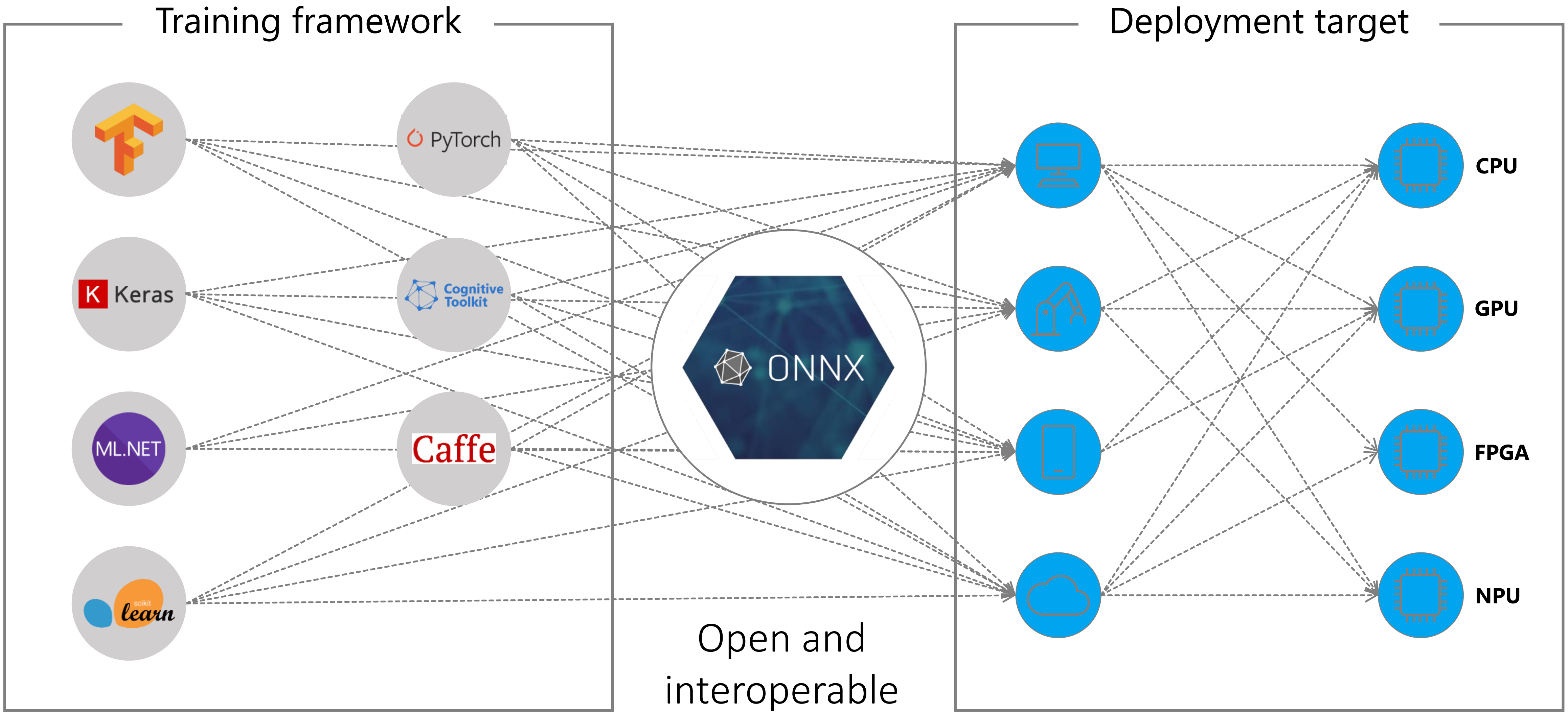
Microsoft HoloLens

Microsoft | Research

Office 365

XBOX

Reality



Open and interoperable

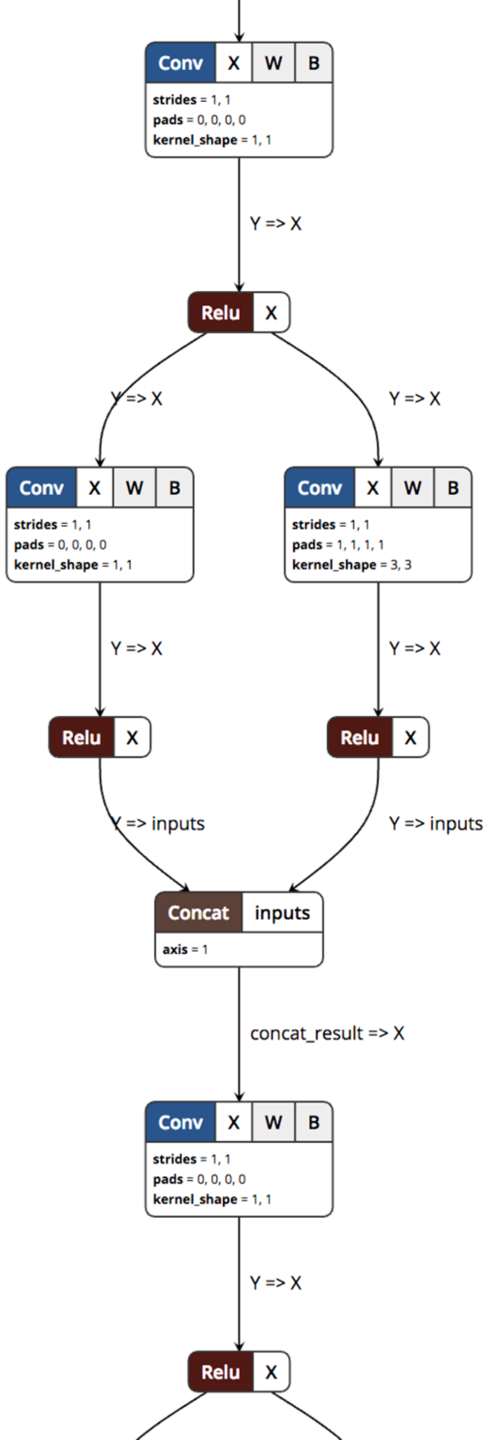
ONNX

https://github.com/onnx

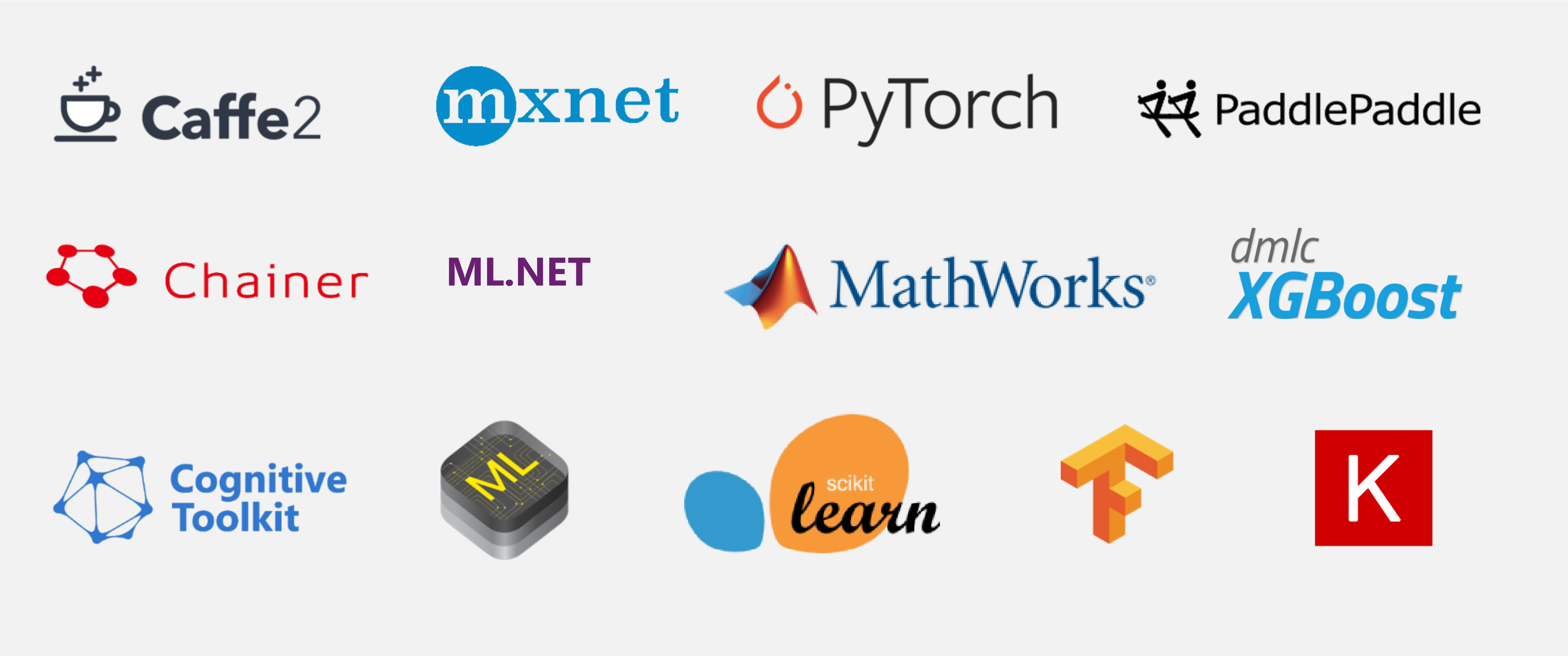
OPEN **N**EURAL **N**ETWORK E**X**CHANGE

What is ONNX?

- **Interoperable standard** format for AI models consisting of:
  - common Intermediate Representation (**IR**)
  - full operator **spec**
- Model = graph composed of computational nodes, based on Google protobuf
- Graph = Compact and cross-platform representation for serialization
- Supports both DNN and traditional ML
- Backward compatible with comprehensive versioning



Framework Compatibility



ONNX Community

LINUX FOUNDATION

Alibaba Group 阿里巴巴集团 AMD arm aws Bai du 百度 BITMAIN

CEVA Facebook Open Source GRAPHCORE habana Hewlett Packard Enterprise HUAWEI

IBM IDEIN Idein Inc intel AI MathWorks MAXAR MEDIATEK Microsoft

Neural Network Libraries mi NVIDIA NXP Oath: A Verizon company OctoML

Preferred Networks Qualcomm sas skymizer SYNOPSYS Tencent unity

Open Governance

Steering Committee

Prasanth Pulavarthi (Microsoft)

Joe Spisak (Facebook)

Vin Sharma (Amazon)

Harry Kim (Intel)

Dilip Sequeira (NVIDIA)

SIG (special interest group)

Architecture/Infrastructure

Lu Fang (Facebook)
Ke Zhang (Microsoft)

Operators

Michał Karzyński (Intel)
Emad Barsoum (Microsoft)

Converters

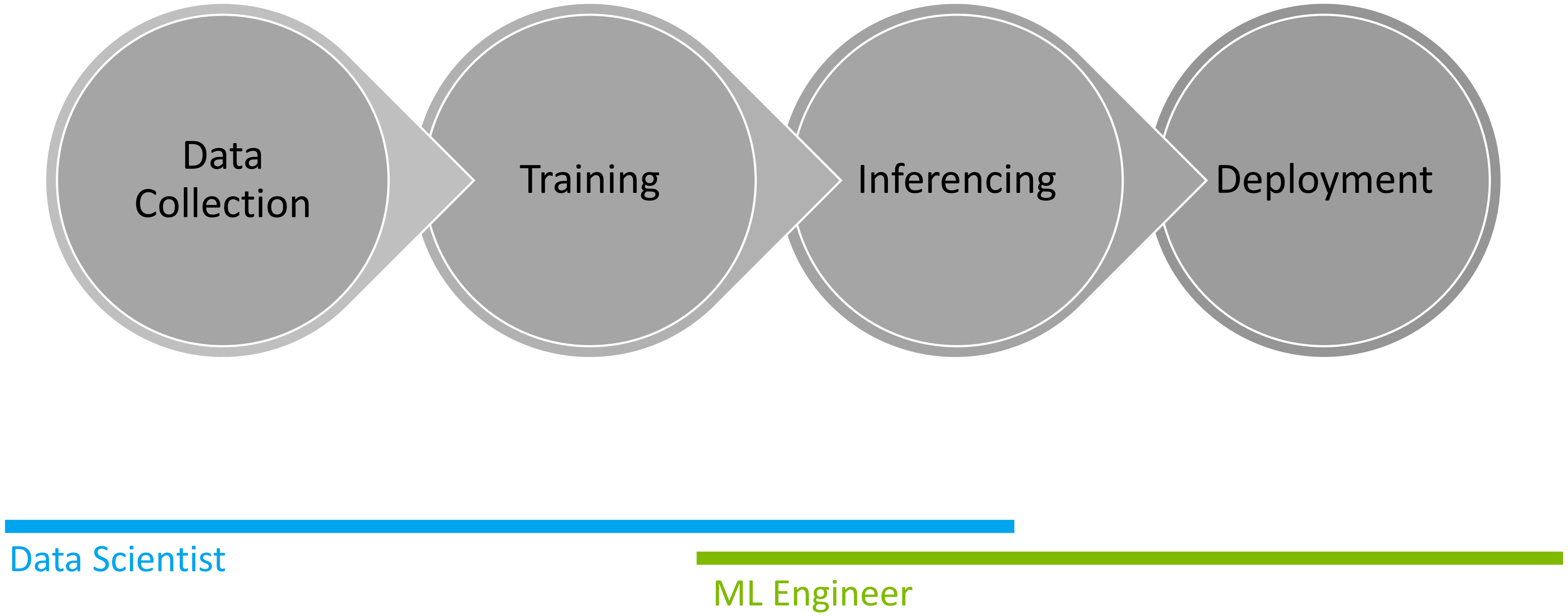
Chin Huang (IBM)
Guenther Schmuelling (Microsoft)

Working Groups

Training

Edge/Mobile

ML Models: Research to Production



Data Collection

Training

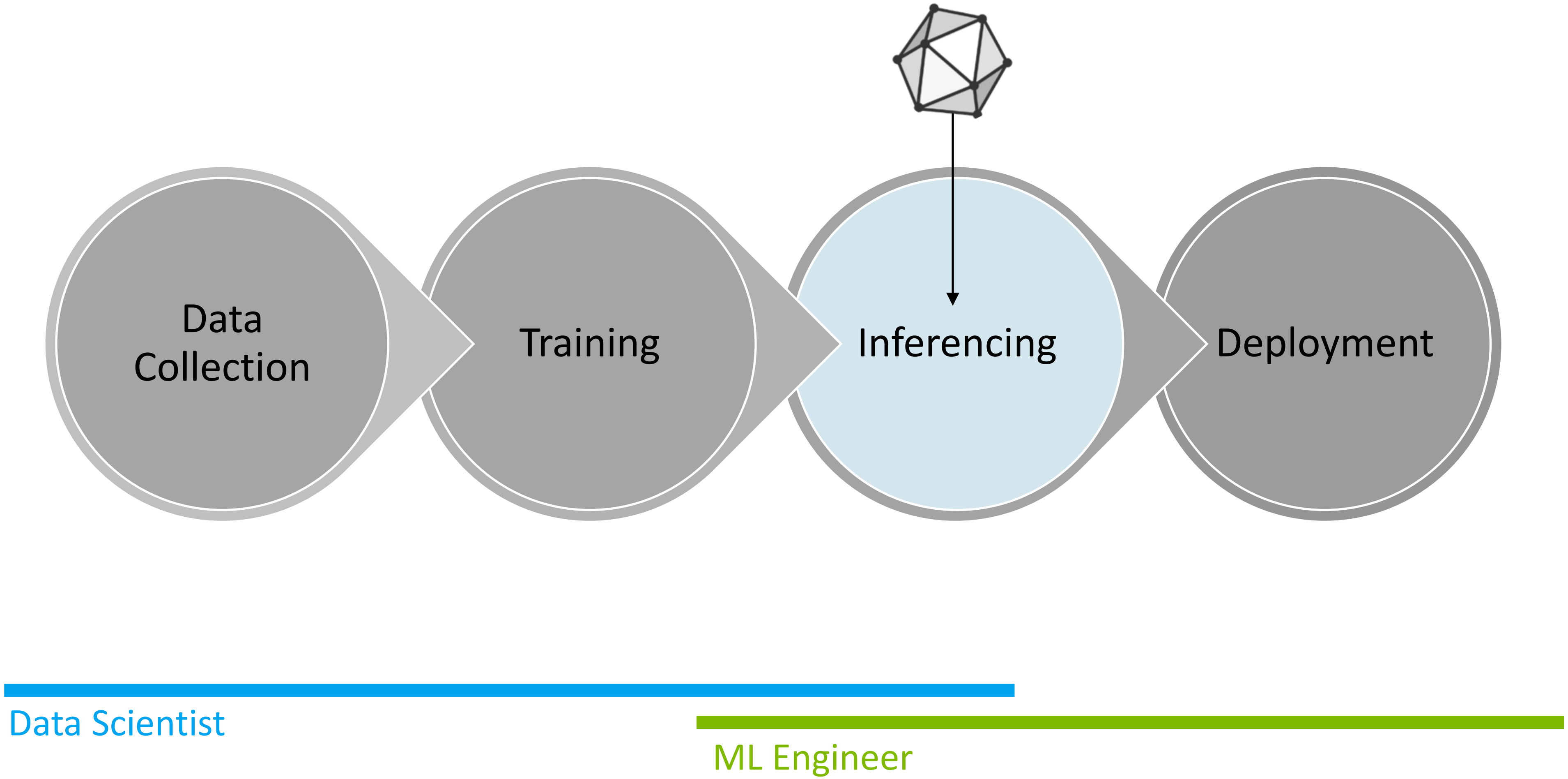
Inferencing

Deployment

Data Scientist

ML Engineer

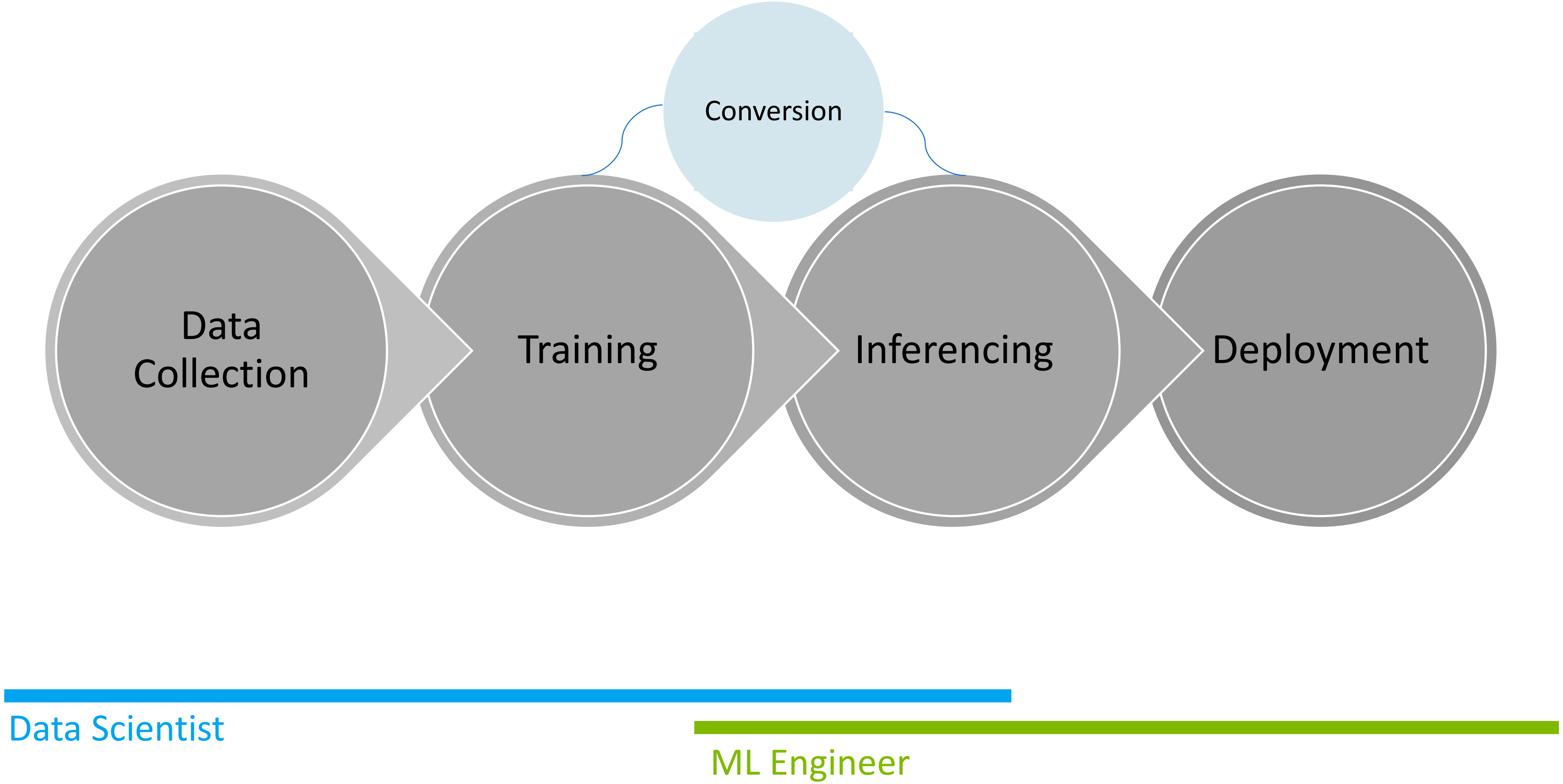
ML Models: Research to Production



How do I get an ONNX model?

- Get a pre-trained ready to use model from the ONNX Model Zoo
- Try a model builder service such as Azure Custom Vision and/or AutoML
- **Convert an existing model from another framework**
- **Train a model via systems such as Azure Machine Learning service and export/convert to ONNX**

ML Models: Research to Production



Conversion

Data Collection

Training

Inferencing

Deployment

Data Scientist

ML Engineer

Open Source converters for popular frameworks

Tensorflow: onnx/tensorflow-onnx

PyTorch (native export)

Keras: onnx/keras-onnx

Scikit-learn: onnx/sklearn-onnx

CoreML: onnx/onnxmltools

LightGBM: onnx/onnxmltools

LibSVM: onnx/onnxmltools

XGBoost: onnx/onnxmltools

SparkML (alpha): onnx/onnxmltools

CNTK (native export)

Examples: Model Conversion

```
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')
```

PyTorch

```
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")
```

```
python -m tf2onnx.convert
         --input frozen_model.pb
         --inputs input_batch:0, lengths:0
         --outputs top_k:1
         --fold_const
         --opset 8
         --output deepcc.onnx
```

Chainer

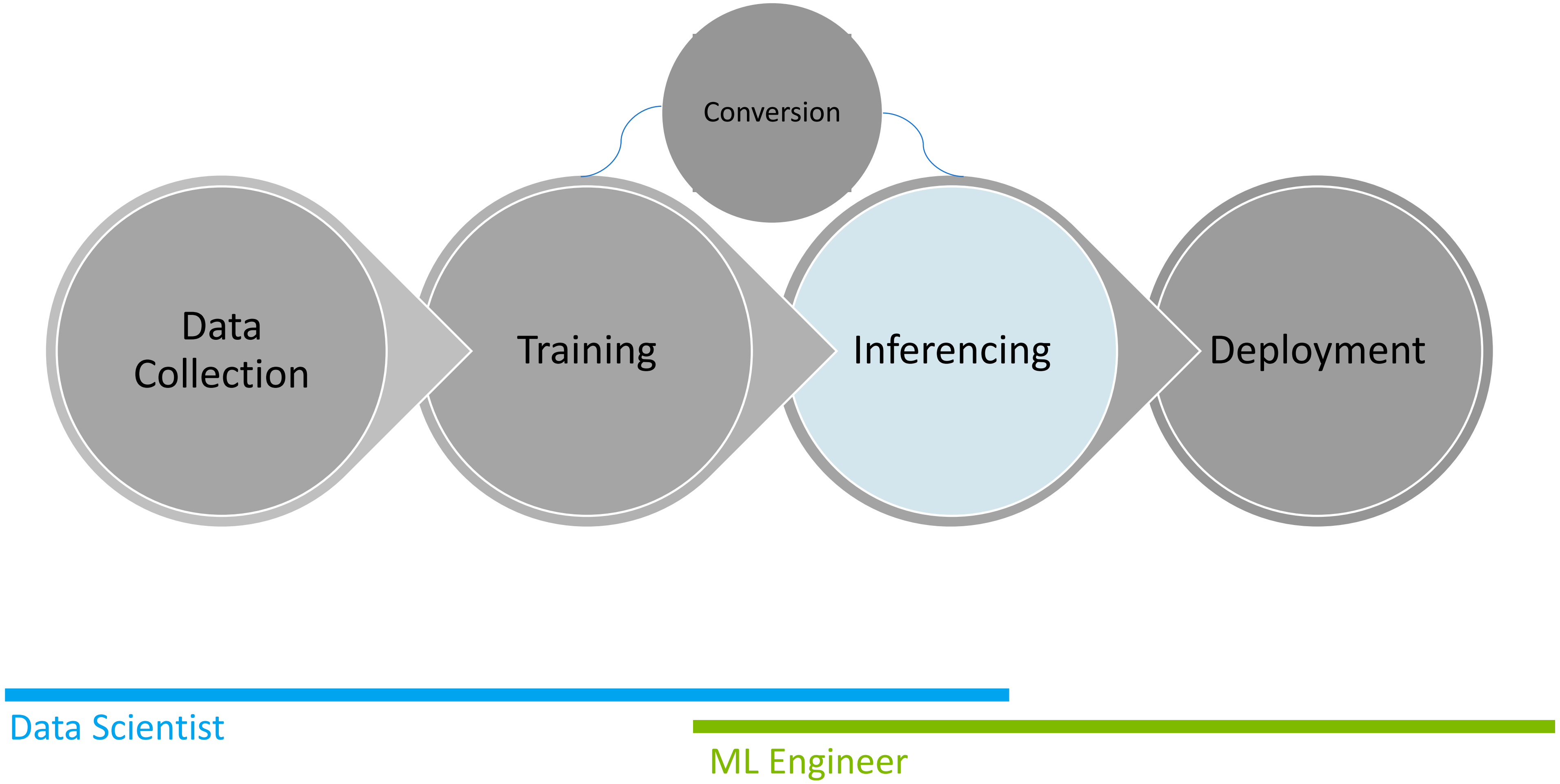
```
import numpy as np
import chainer
from chainer import serializers
import onnx_chainer

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")
```

ACTIVITY A

TRAIN A MODEL USING PYTORCH AND EXPORT TO ONNX FORMAT

Inferencing ONNX models

ML Models: Research to Production



Conversion

Data Collection

Training

Inferencing

Deployment

Data Scientist

ML Engineer

ONNX Runtime

aka.ms/onnxruntime

github.com/microsoft/onnxruntime

ONNX Runtime is an open source high performance **Inference Engine** for ONNX models

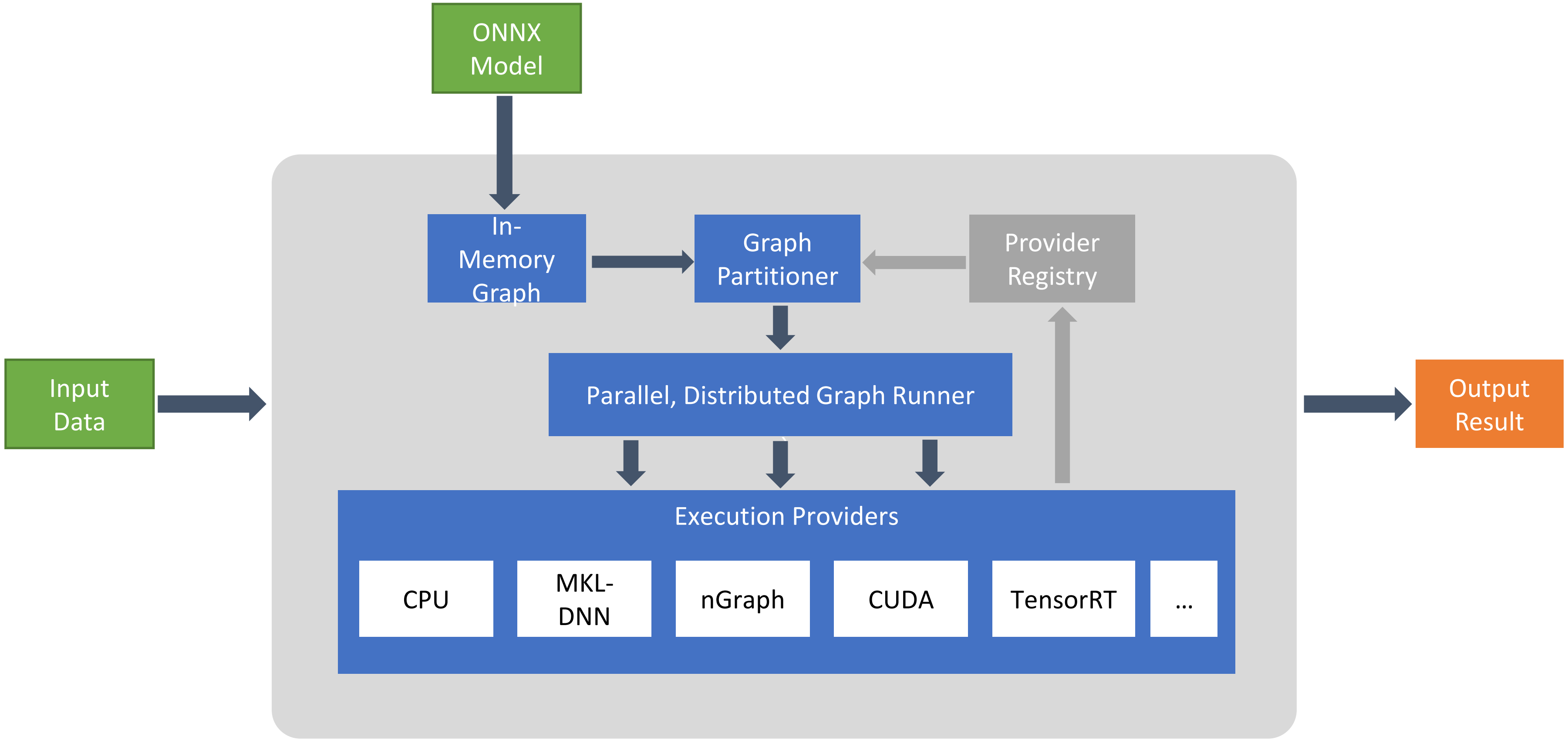
ONNX Runtime

- Cross platform, multi-language API
- Full ONNX spec support
  - Covers both ONNX and ONNX-ML domain model spec and operators
  - Backwards and forwards compatible
  - Flexibility for custom operators
- High performance through:
  - Graph optimizations – node fusions
  - Execution Providers – Leverage custom accelerators and runtimes to enable maximum performance
  - Model partitioning – Assign the best Execution Provider
- Extensible and modular framework
  - Clear API for plug-in graph optimizers, operators, and hardware accelerators

Supported Architectures and Languages

- Windows, Linux, Mac
- X64, X86, ARM
- CPU, GPU
- Python, C, C++, C#, Ruby, Java (soon)

ONNX Runtime – high level architecture



Examples: Inferencing with ONNX Runtime

```python
import onnxruntime

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

results = session.run([], {"input": input_data})
```

C#

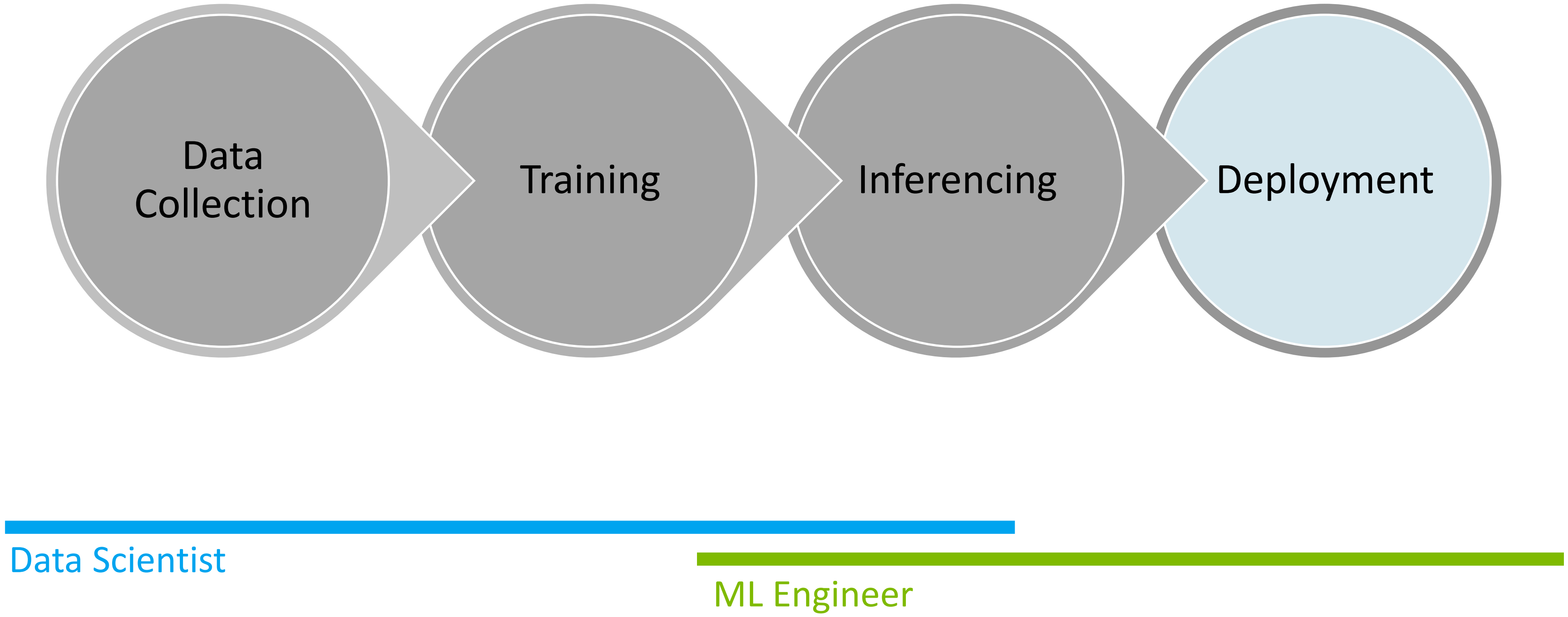
```csharp
using Microsoft.ML.OnnxRuntime;

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

var results = session.Run(input);
```

Execution Providers for acceleration

Base CPU Microsoft Linear Algebra Subprograms	NVIDIA CUDA	NVIDIA TensorRT
Intel nGraph	Intel MKL-DNN	Intel OpenVINO
NUPHAR TVM/LLVM-based model compiler	NN API for Android (future)	DirectML

Deployment

ML Models: Research to Production



Data Collection

Training

Inferencing

Deployment

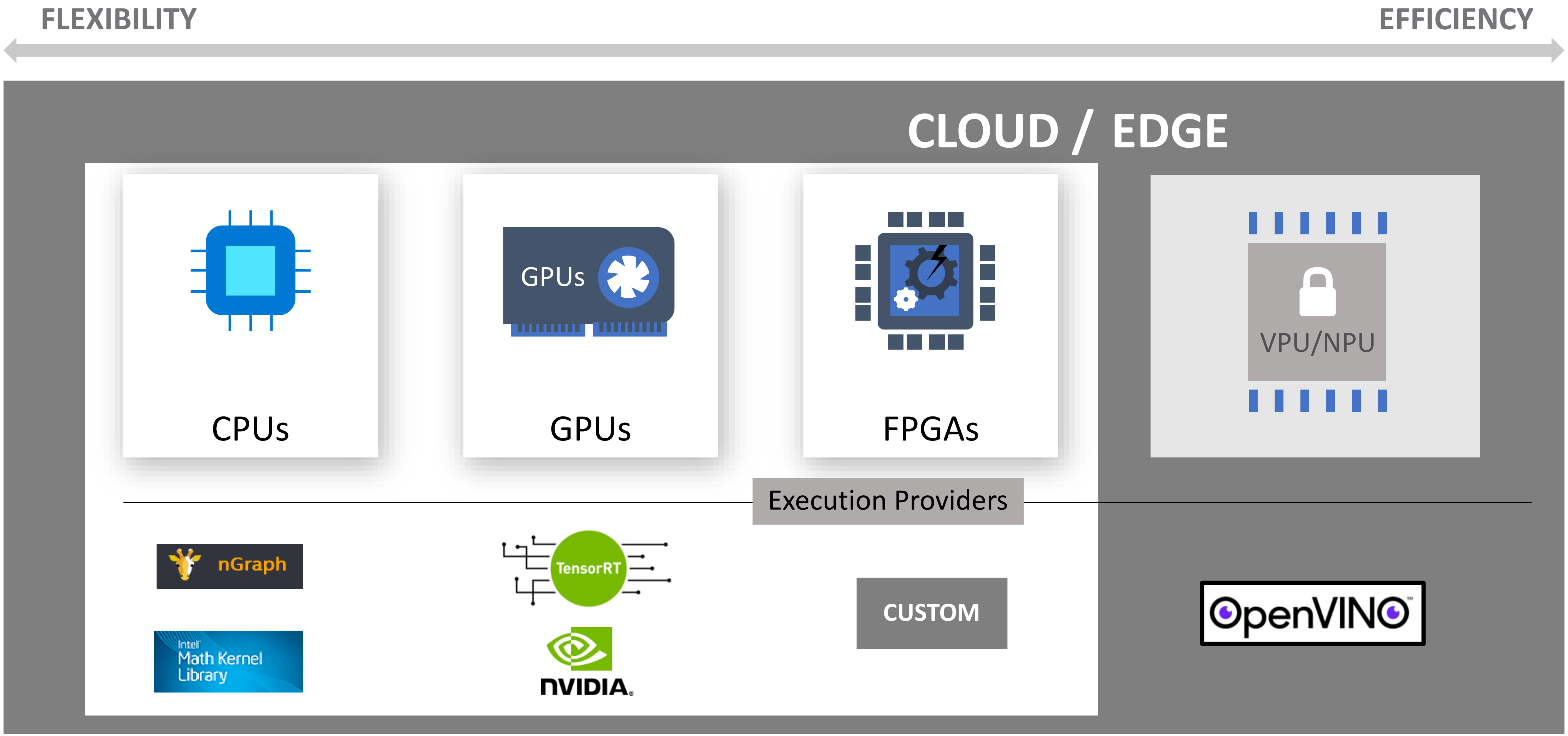
Data Scientist

ML Engineer

ACTIVITY B

DEPLOY YOUR ONNX MODEL AS A WEB SERVICE USING AZURE ML

Variety of Deployment Options



FLEXIBILITY

EFFICIENCY

CLOUD / EDGE

CPUs

GPUs

FPGAs

VPU/NPU

Execution Providers

nGraph

Intel Math Kernel Library

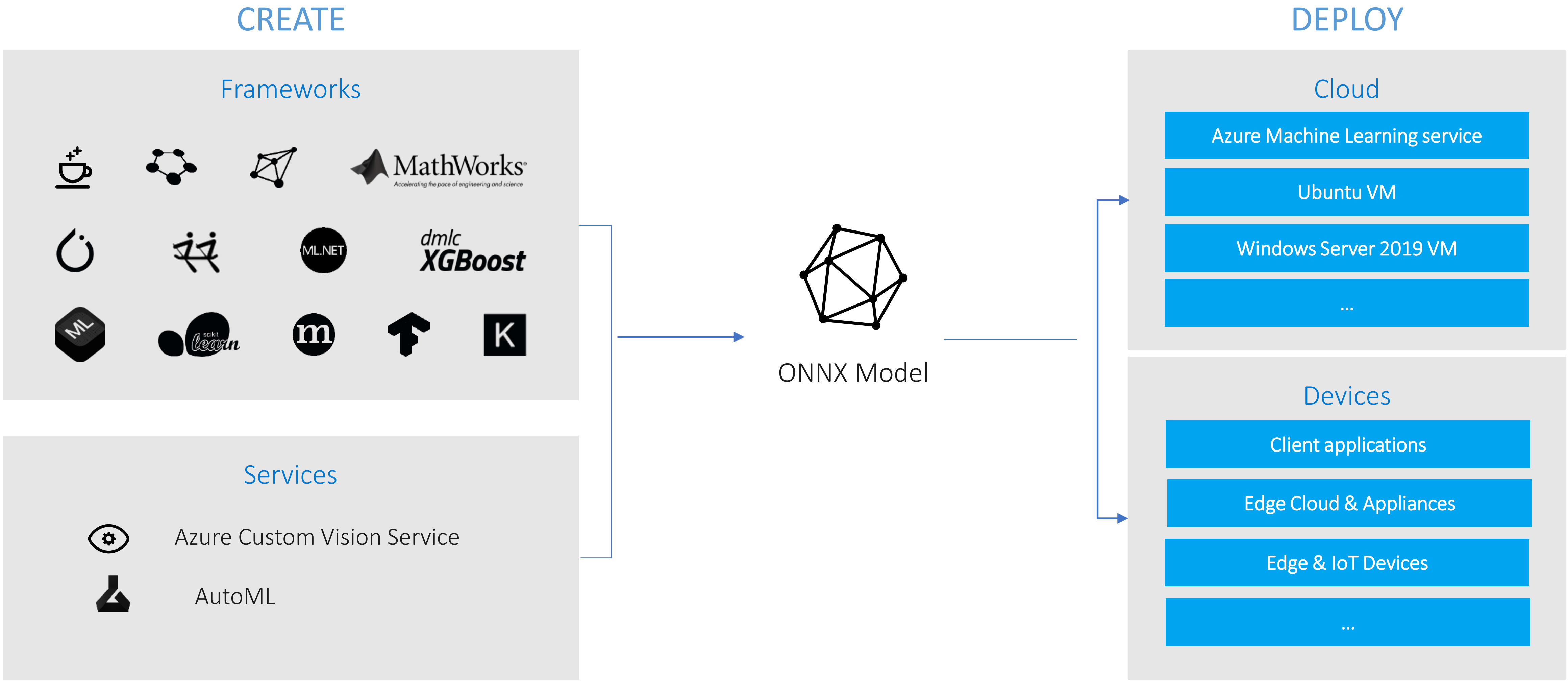
TensorRT

NVIDIA

CUSTOM

OpenVINO

Create and Deploy Models



CREATE

Frameworks

Services

Azure Custom Vision Service

AutoML

ONNX Model

DEPLOY

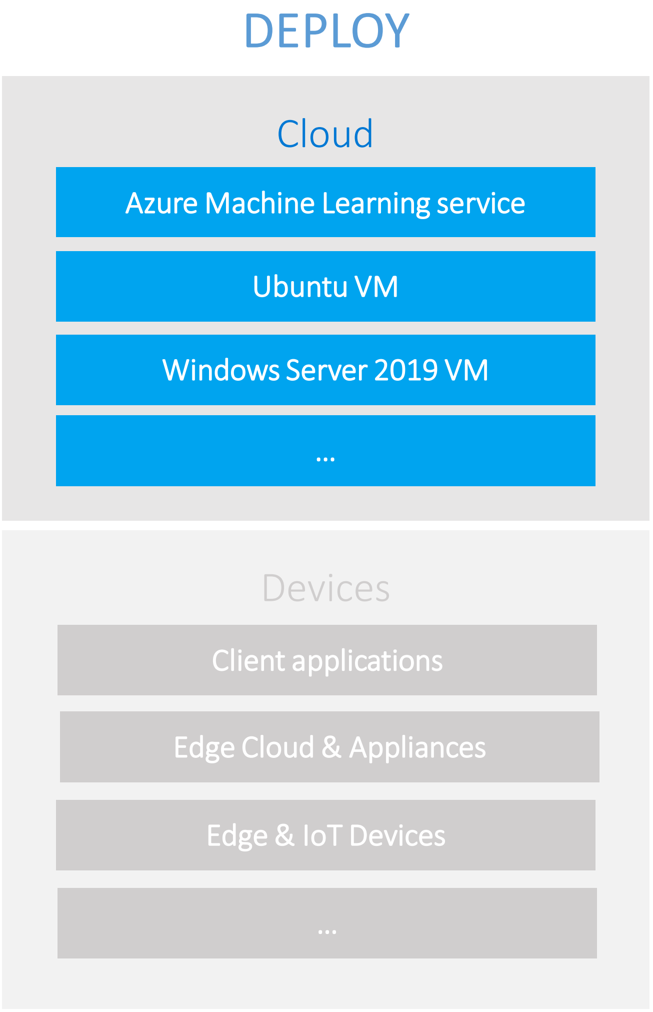
Cloud

- Azure Machine Learning service
- Ubuntu VM
- Windows Server 2019 VM
- ...

Devices

- Client applications
- Edge Cloud & Appliances
- Edge & IoT Devices
- ...

Cloud



DEPLOY

Cloud

Azure Machine Learning service

Ubuntu VM

Windows Server 2019 VM

...

Devices

Client applications

Edge Cloud & Appliances

Edge & IoT Devices

...

Azure Machine Learning Services

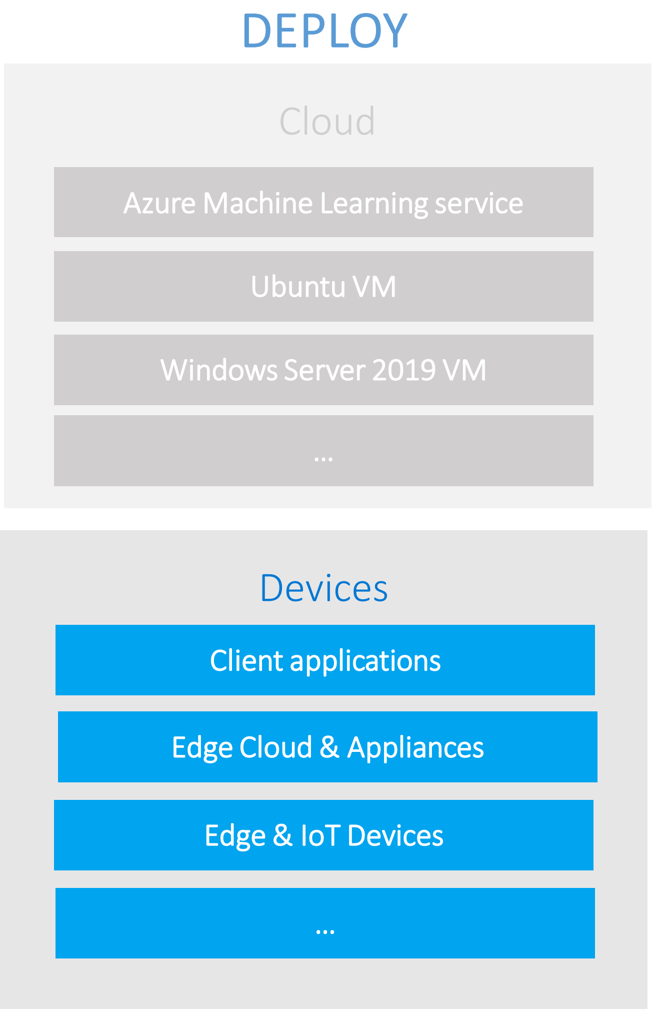
AzureML helps manage the process of deploying a model

Supports various compute targets including Container instances, Azure Kubernetes Service, and more.

General VMs

ONNX Runtime supports various versions and flavors of Linux as well as Windows, so it can run on your compute of choice from Ubuntu VMs to Windows Server

Devices



Client Applications

Download the published binaries from Nuget or PyPi (or build from source) to incorporate into your Windows, Linux, or Mac applications. For Windows 10, you can also use the built-in WinML APIs for inferencing ONNX models.

Edge Cloud and Appliances

The ONNX Runtime packages can be installed and used on private clouds and hosted on-premise servers.

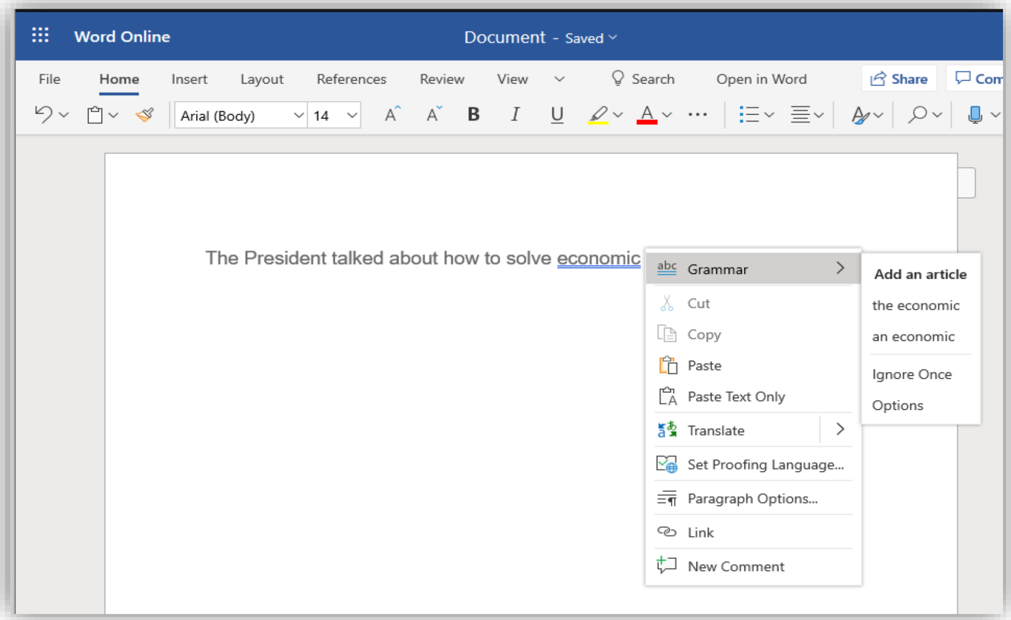
Edge and IoT Devices

Edge and IoT devices have more restrictions due to their smaller footprint, but are still compatible with ONNX Runtime. See these examples for deploying ONNX Runtime with on IoT devices:

- Deploying to Intel OpenVINO based devices
- Deploying to NVIDIA Jetson Nano (ARM64)

Real World Usage

Office : Missing Determiner



The President talked about how to solve economic

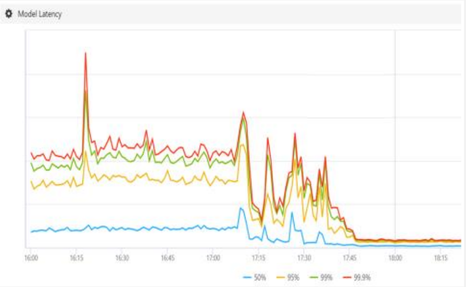
FEATURE OVERVIEW

The **Missing Determiner** model is used in Word Online to determine if a sentence is missing an "a", "an", "the", etc. in front of words.

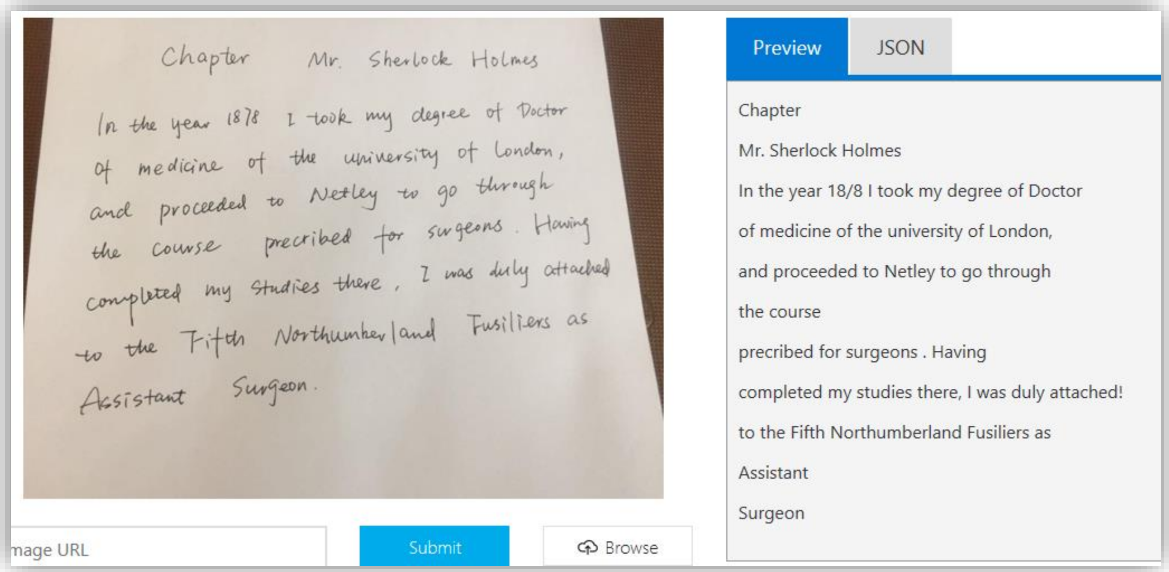
Double blue underlines appear on words with missing determiners, and suggested determiners appear in a context menu for easy correction.

PERFORMANCE

<u>14.6x</u> performance gain with ONNX and ONNX Runtime



Cognitive Service : Optical Character Recognition



Chapter Mr. Sherlock Holmes

In the year 1878 I took my degree of Doctor of medicine of the university of London, and proceeded to Netley to go through the course precribed for surgeons. Having completed my studies there, I was duly attached to the Fifth Northumberland Fusiliers as Assistant Surgeon.

Chapter

Mr. Sherlock Holmes

In the year 18/8 I took my degree of Doctor

of medicine of the university of London,

and proceeded to Netley to go through

the course

precribed for surgeons . Having

completed my studies there, I was duly attached!

to the Fifth Northumberland Fusiliers as

Assistant

Surgeon

FEATURE OVERVIEW

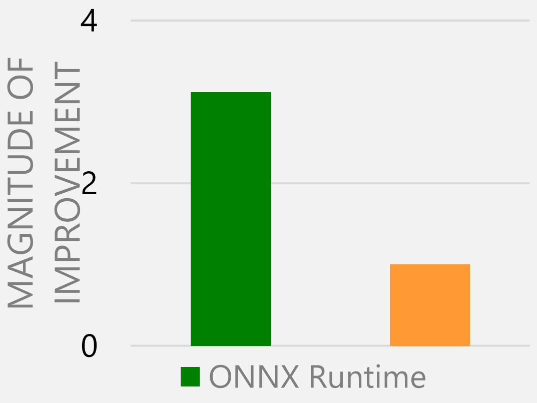
Extracting schematized digital data from analog-born entities like images, scanned documents, invoices, receipts, etc. is a fundamental computer vision capability.

Azure Optical Character Recognition (OCR) Service powers at-scale motions in Office 365, Dynamics, and Azure Search.

The OCR model is used to detect text in an image and extract the recognized words into a machine-readable character stream

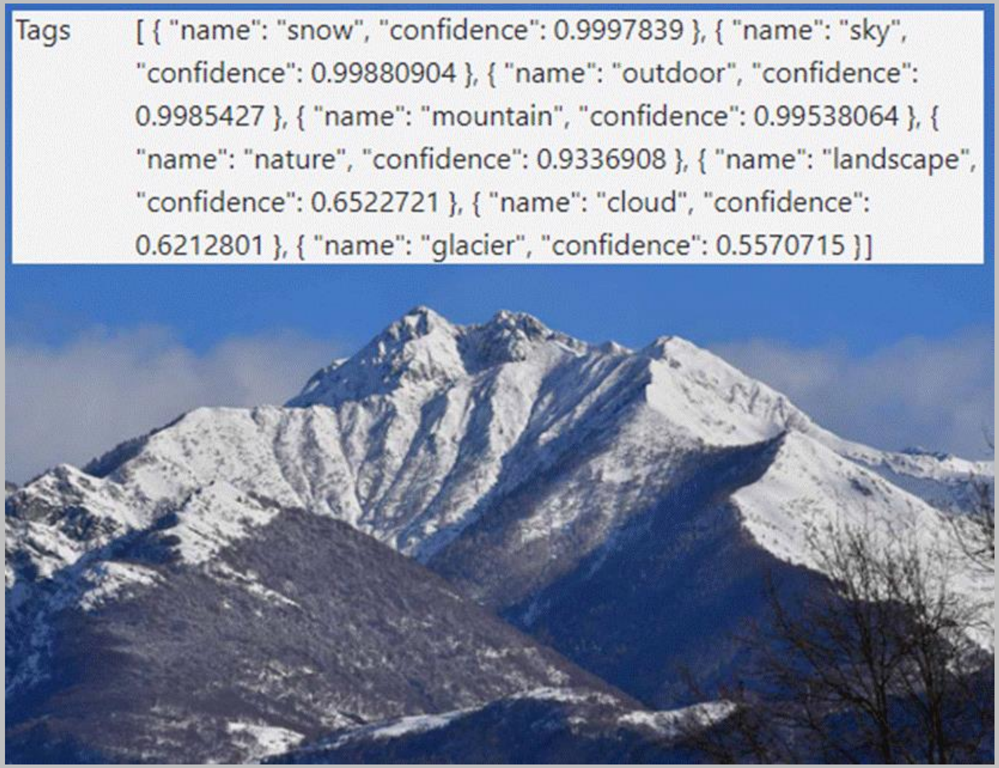
PERFORMANCE

>3x perf gain by using ONNX and ONNX Runtime



Cognitive Service: Computer Vision

FEATURE OVERVIEW



Tags [{ "name": "snow", "confidence": 0.9997839 }, { "name": "sky", "confidence": 0.99880904 }, { "name": "outdoor", "confidence": 0.9985427 }, { "name": "mountain", "confidence": 0.99538064 }, { "name": "nature", "confidence": 0.9336908 }, { "name": "landscape", "confidence": 0.6522721 }, { "name": "cloud", "confidence": 0.6212801 }, { "name": "glacier", "confidence": 0.5570715 }]

MODELS

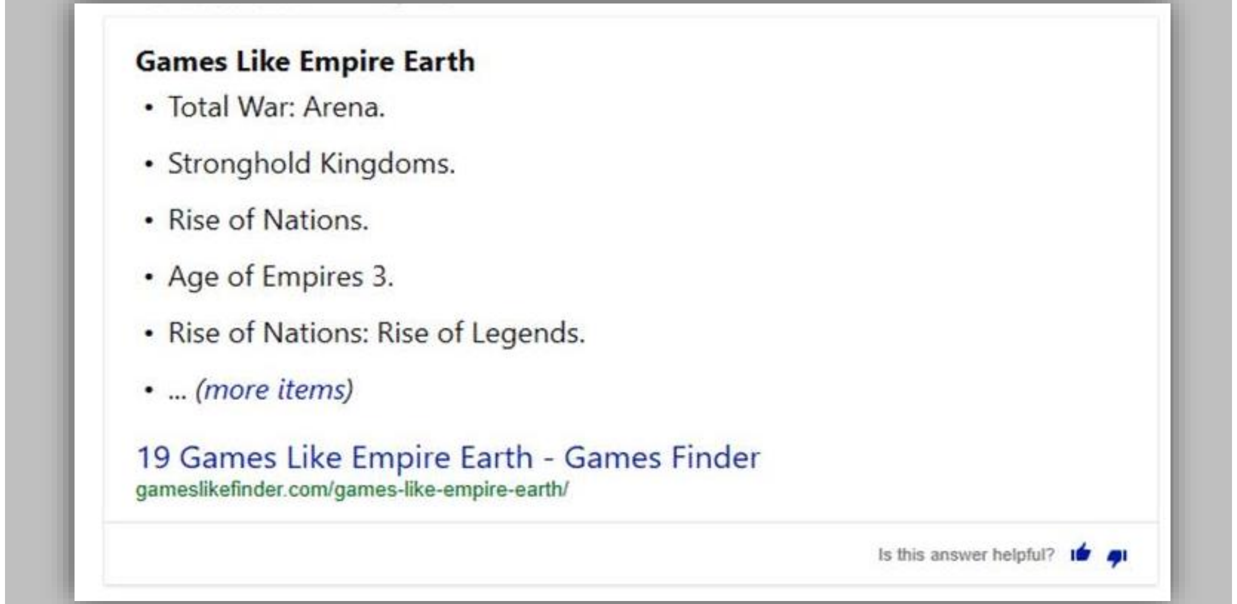
2 computer vision models are used for enriching images with metadata

PERFORMANCE

- Latency reduced by 43%
- Throughput increased 1.77x

Bing QnA : List and Segment

FEATURE OVERVIEW



Games Like Empire Earth

- Total War: Arena.
- Stronghold Kingdoms.
- Rise of Nations.
- Age of Empires 3.
- Rise of Nations: Rise of Legends.
- ... *(more items)*

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

Is this answer helpful?

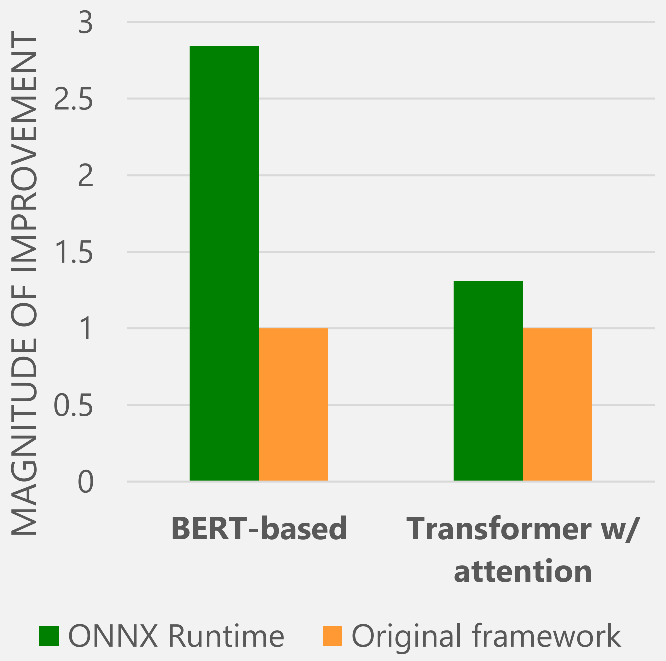
QUERY: “empire earth similar games”

MODELS

2 Bing models are used for generating answers from user queries

PERFORMANCE

Up to **2.8x** perf improvement with ONNX Runtime



References

- Training, Inferencing, and deployment in AzureML with ONNX models: https://aka.ms/onnxnotebooks
- AzureML resources:
- Windows ML:
- ONNX: https://github.com/onnx/onnx
- ONNX Runtime: https://github.com/microsoft/onnxruntime
- ONNX Runtime Tutorials: https://github.com/microsoft/onnxruntime#examples-and-tutorials
- ONNX Tutorials: https://github.com/onnx/tutorials
- ONNX Converters: https://github.com/onnx/onnxmltools/tree/master/onnxmltools
- ONNX Ecosystem Docker Image: https://github.com/onnx/onnx-docker/tree/master/onnx-ecosystem
- Perf Tuning: https://github.com/microsoft/onnxruntime/blob/master/docs/ONNX_Runtime_Perf_Tuning.md

FAITH XU| faxu@microsoft.com

PRABHAT ROY| Prabhat.Roy@microsoft.com