Open Neural Network Exchange (ONNX)

The open standard for machine learning interoperability – Introduction

Dr. Andreas Fehlner (TRUMPF Laser GmbH, Heidelberg Institute for Theoretical Studies)

ONNX Steering Committee

Wiehl, 13.09.2022, Workshop: Conceptual Advances in Deep Learning for Research on Universe and Matter

THE LINUX FOUNDATION

LF AI & DATA

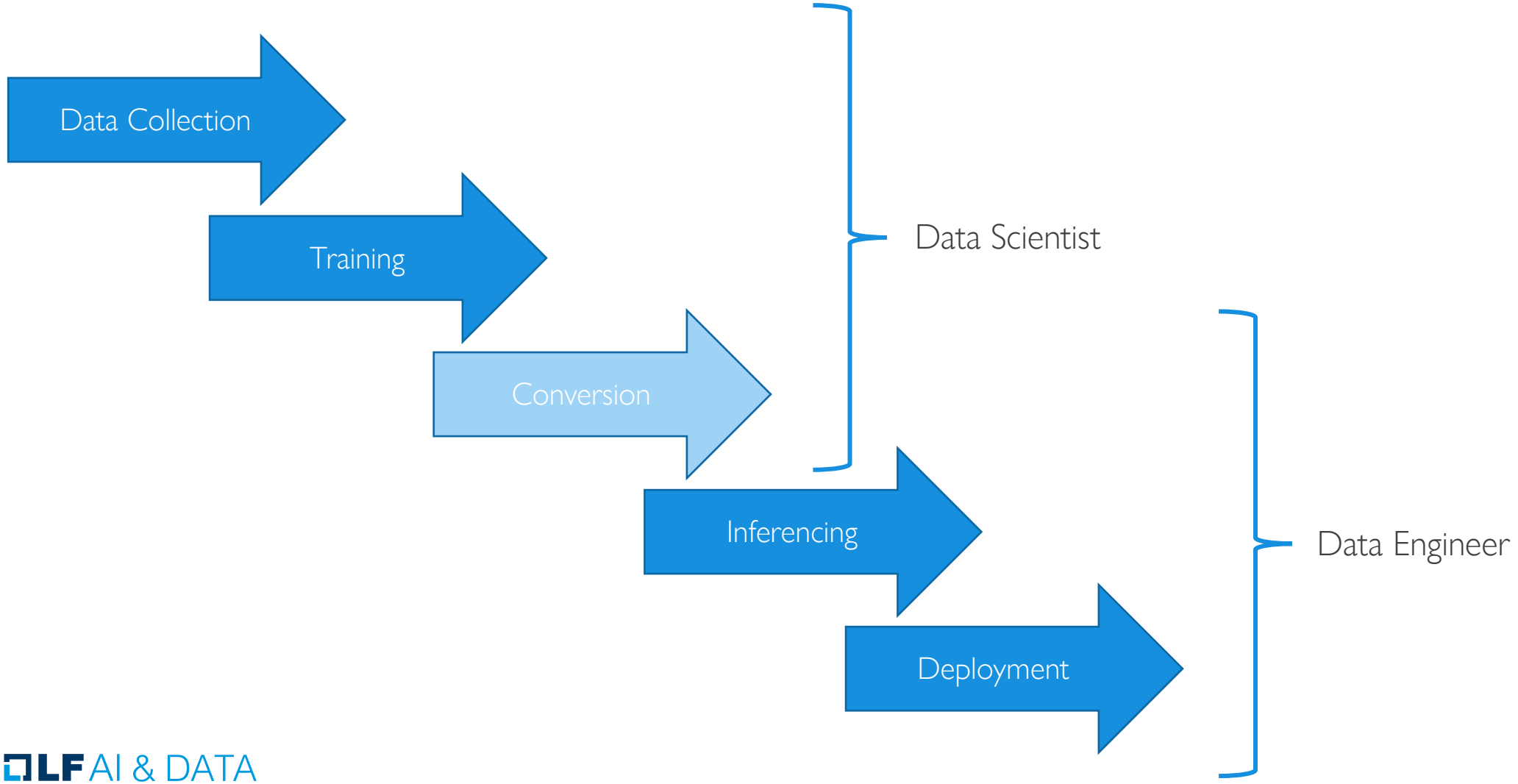
Outline

- Motivation
- ONNX Technical Design
- Community
- Onnxruntime Technical Design
- Current topics
- Future Directions

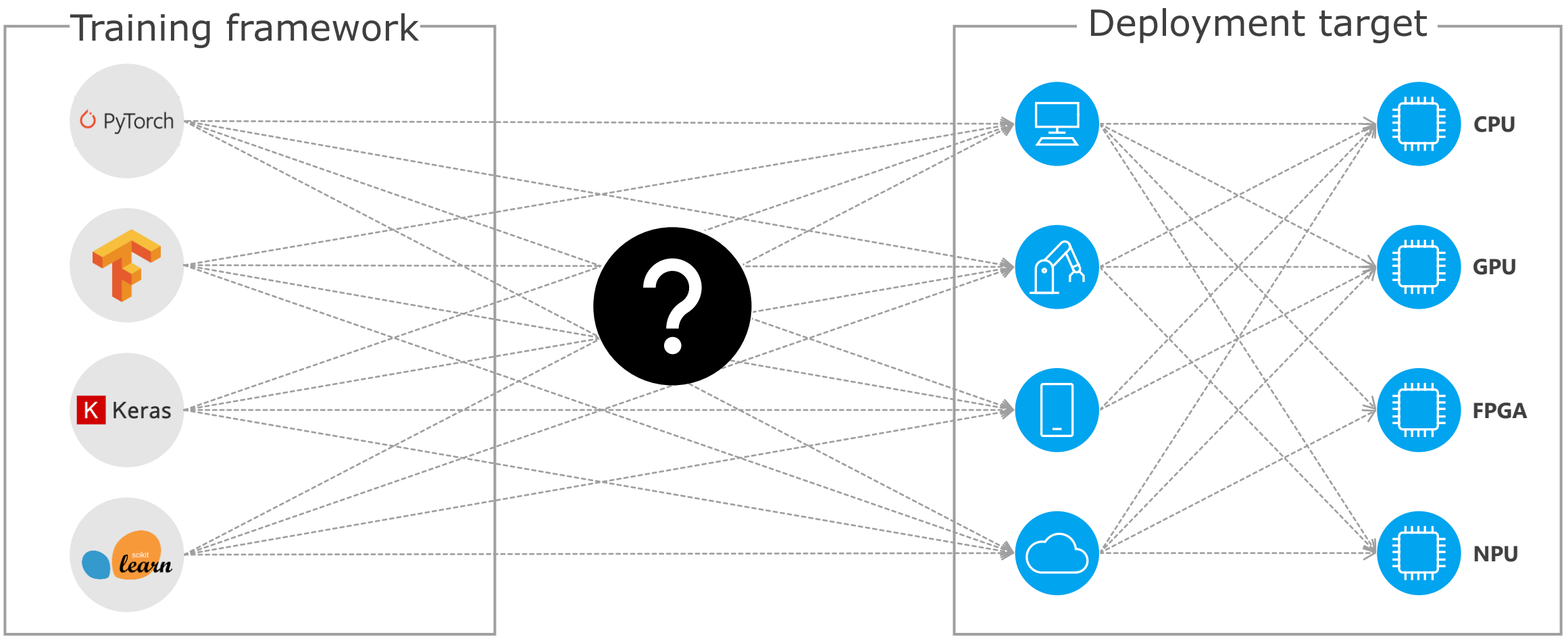
Motivation

LF AI & DATA

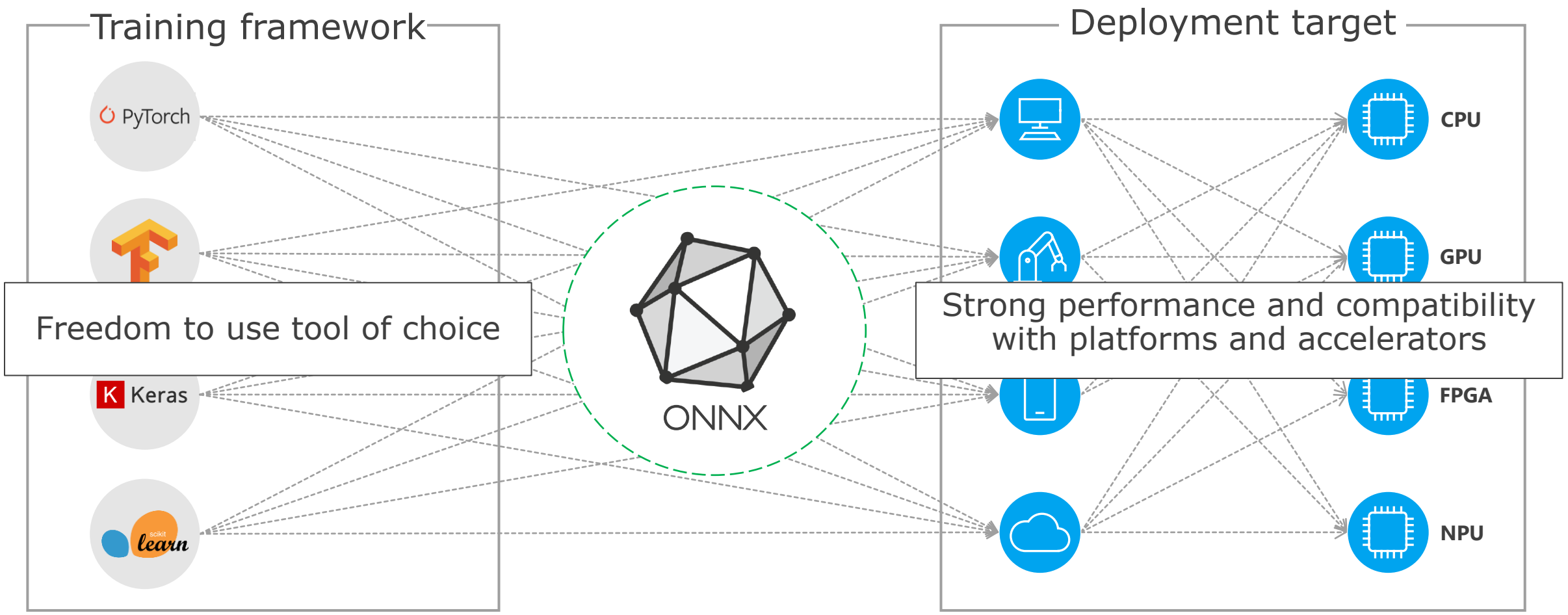
ML Models: Research to Production



Reality: Fragmentation of ML frameworks



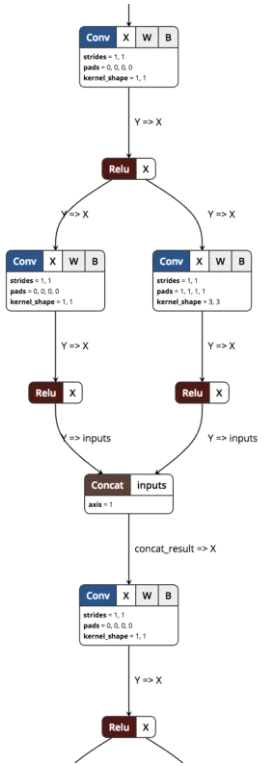
Improving ML productivity



ONNX – Technical Design

LF AI & DATA

What is Onnx



ONNX

› **O**PEN **N**EURAL **N**ETWORK E**X**CHANGE

Design Principles:

- Support both DNN and traditional ML
- Interoperable standard for AI models
- Backward compatible
- Consists of a common Intermediate Representation (IR) + full operator spec
- Compact and cross-platform representation for serialization

ONNX IS NOT ABOUT:

- Graph execution implementation
- Operator optimizations
- Framework API definition

Community

LF AI & DATA

ONNX Community

ONNX is a Community Project (Open Governance)

Steering Committee

https://github.com/onnx/steering-committee

Prasanth Pulavarthi (Microsoft)

Alexander Eichenberger (IBM)

Mayank Kaushik (NVIDIA)

Rajeev Nalawadi (Intel)

Andreas Fehlner (TRUMPF Laser)

Special Interest Groups (SIGs) and Working Groups

https://github.com/onnx/sigs

Architecture & Infra: Liqun Fu, Ke Zhang

Operators: Michał Karzyński, Ganesan Ramalingam

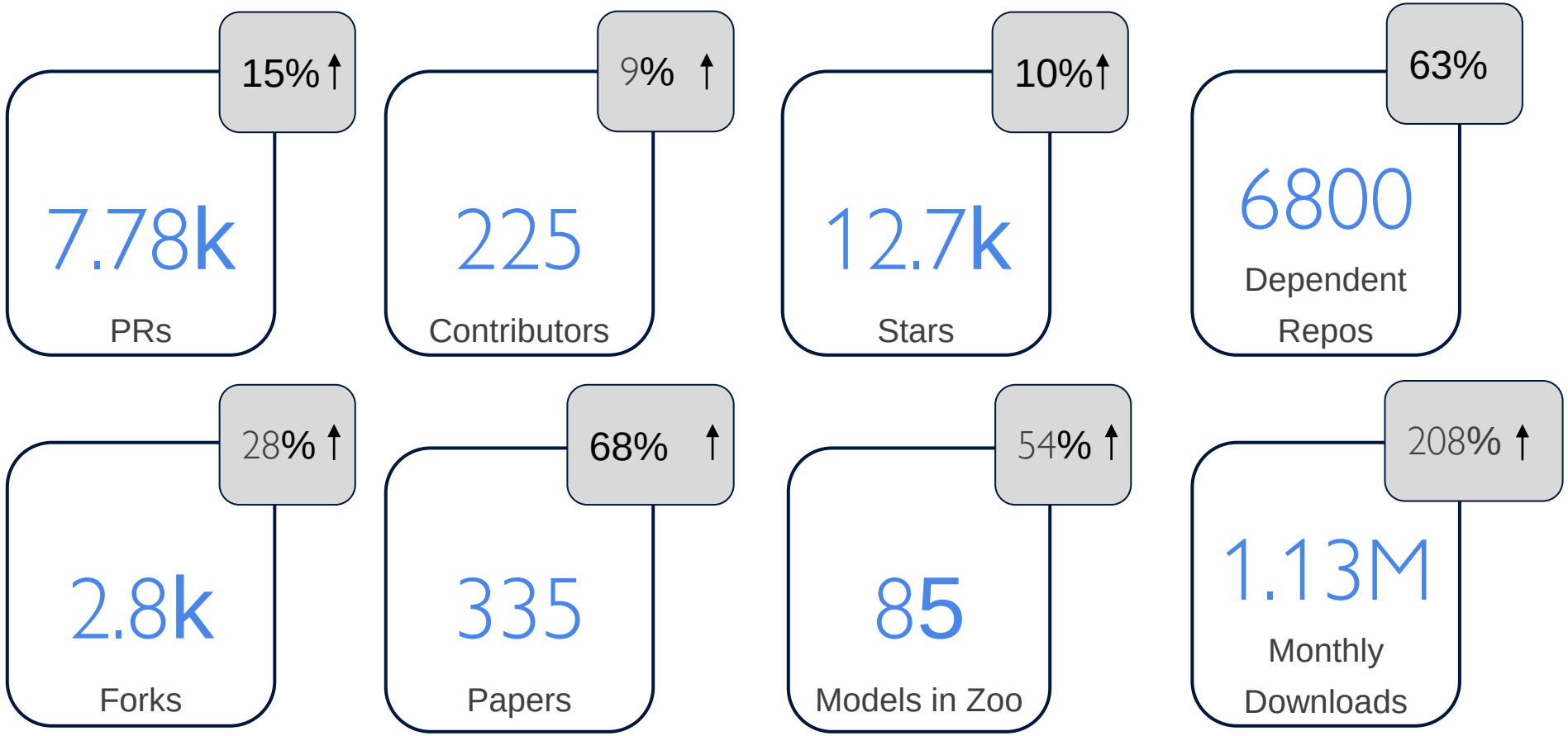
Converters: Thiago Crepaldi, Kevin Chen

Model Zoo & Tutorials: Jacky Chen

Pre-processing (WG): Joaquin Anton

Tools (and companies) that support Onnx

Engagement & usage (from 10/20/21 to 06/13/2022)



Community

- 2 Community Meetups per Year
- Agenda:
  - Steering Committee Updates
  - SIG/WG Updates
  - Partner and User Stories
- https://onnx.ai/meetups/june2022

Mauro Bennici, GhostWriter.AI	Designed to be Optimized
Dheeraj Peri, NVIDIA	INT8 Inference of Quantization-Aware trained models using ONNX-TensorRT
Alessandro Pappalardo, AMD	QONNX: A proposal for representing arbitrary-precision quantized NNs in ONNX
Daniel Huynh, Mithril Security	How to reconcile AI and privacy
Adam Pocock, Oracle	ONNX and the JVM
Rodolfo Gabe Esteves, Bhargavi Karumanchi, Ria Cheruvu, Rajeev Nalawadi, Intel	Responsible AI @ ONNX: Metadata, Model Cards, and Provenance
Qing Lan, AWS	Build your high-performance model inference solution with DJL and ONNX Runtime
Viet Yen Nguyen, Hypefactors	Billions of NLP Inferences on the JVM using ONNX and DJL
Ryan Hill, Microsoft	What's New in ONNX Runtime
Jeff Boudier, Hugging Face	Accelerating Machine Learning with ONNX Runtime and Hugging Face
Pranav Marathe, NVIDIA	ONNX Tools: Polygraphy and ONNX-GraphSurgeon
Zijian Xu, Preferred Networks	PFVM - A Neural Network Compiler that uses ONNX as its intermediate representation
Tung D. Le, IBM	Onnx-mlir: an MLIR-based Compiler for ONNX Models - The Latest Status

History

- ONNX was originally named Toffee and was developed by the PyTorch team at Facebook.
- In September 2017 it was renamed to ONNX and announced by Facebook and Microsoft.
- Later, IBM, Huawei, Intel, AMD, Arm and Qualcomm announced support for the initiative.
- In November 2019 ONNX was accepted as graduate project in Linux Foundation AI.

- Yesterday: Pytorch announces the foundation of the Pytorch foundation and becoming part of the Linux Foundation

LF AI
GRADUATE
PROJECT

How to I get an ONNX model

- ONNX Model Zoo
- Model creation services such as “Azure Custom Vision” and/or AutoML
- Convert an existing models from another framework

- End to End training via systems such as Azure Machine Learning service

Open Source converter for popular frameworks

- Tensorflow
- Scikit-learn
- Apple Core ML
- Spark ML
- LightGBM
- Libsvm
- XGBoost
- H2O
- CatBoost

Native export

- Pytorch
- CNTK

Examples: Model Conversion

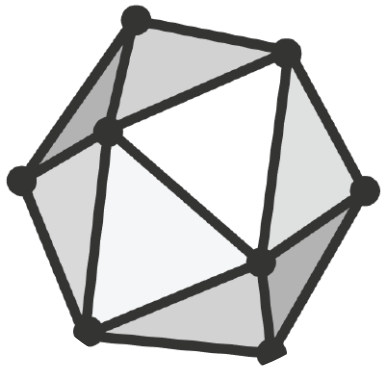
```
import torch
import torch.onnx

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
```

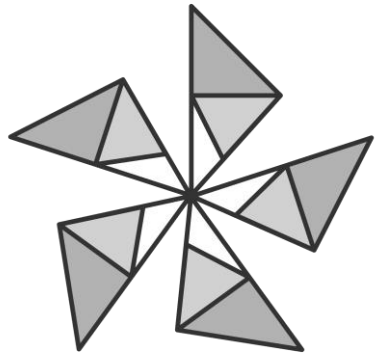
```
import sklearn
import skl2onnx

initial_type = [('float_input', FloatTensorType([1, 4]))]

onnx_model = skl2onnx.convert_sklearn(pipe,
initial_types=initial_type)

with open("logreg_iris.onnx", "wb") as f:
    f.write(onnx_model.SerializeToString())
```

Common problems impacting ML productivity

- Inference latency is too high to put into production
- Training in Python but need to deploy into a C#/C++/Java app
- Model needs to run on edge/IoT devices
- Same model needs to run on different hardware and operating systems
- Need to support running models created in several different frameworks
- (more recently) Training very large models takes too long



+



ONNX

https://onnx.ai

@onnxai

ONNX Runtime

https://onnxruntime.ai

@onnxruntime

Compatible with PyTorch, TensorFlow, Keras, SciKit-Learn, and more

ONNX Runtime

Get Started Easily

Optimize Inferencing	Optimize Training							
Platform	Windows	Linux	Mac	Android	iOS	Web Browser (Preview)		
API	Python	C++	C#	C	Java	JS	Obj-C	WinRT
Architecture	X64	X86	ARM64	ARM32	IBM Power			
Hardware Acceleration	Default CPU	CoreML	CUDA	DirectML	oneDNN			
	OpenVINO	TensorRT	NNAPI	ACL (Preview)	ArmNN (Preview)			
	MIGraphX (Preview)	Rockchip NPU (Preview)	SNPE	TVM (Preview)	Vitis AI (Preview)			
Installation Instructions	Please select a combination of resources							

Using ONNX Runtime

```
import onnxruntime

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

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

..... also available for C, C++, Java, and JavaScript (Node.js)

```
using Microsoft.ML.OnnxRuntime;

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

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

C#

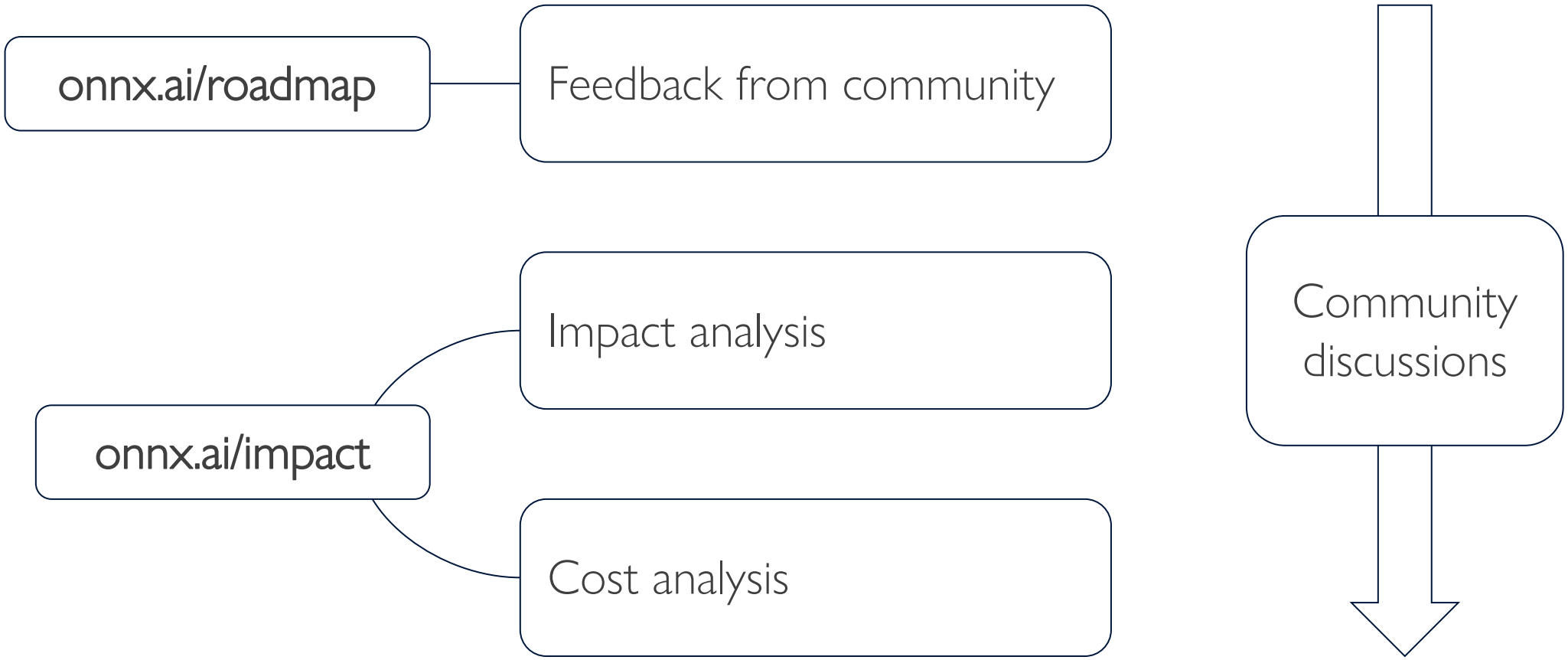
The future of ONNX is up to you

LF AI & DATA

ONNX has open governance

- Annual Steering Committee election
- Technical decisions made by SIGs and Working Groups
- All meetings open to everyone
- Calendar: https://onnx.ai/calendar
- GitHub: https://github.com/onnx
- Slack: #onnx-general on https://slack.lfai.foundation
- Mailing List: https://lists.lfai.foundation/g/onnx-announce

ONNX roadmap discussions



ONNX

Roadmap
Requests
Current status

ONNX Roadmap Items (Status) - 1

Topics	Proposed SIG's	Current Status or Future positioning
New operators for data processing to cover ML pipeline – Nakaike (IBM)	Operators, Pre-processing	Identified from operators group about significant overlap with extensions already implemented. Work continues on processing dates (feature)
C API for C++ components of ONNX (to assist in wrapper for model checker functionality) – Pocock (Oracle)	Arch/Infra	C API to wrap protobuf which python, C#, Java can interact to emit ONNX models continues as long term goal. Currently C#, Java have their own construction and validation code
Better support for emitting ONNX models from other languages beyond Python – Pocock (Oracle)	Arch/Infra	"Same as above"
Add meta information in tensors – Croome (Greenwaves)	Arch/infra, Converters, Release	Identifying path to get more structured quantization information in onnx to match/exceed tflite's abilities.
E2E pipeline with ONNX operators (include Keras, TF, Scikit-learn/Spark pipeline preprocessing flows) using single graph – Sica (IBM)	Arch/Infra, Model Tutorial, Operators, Pre-processing	Identified as long term intercept, further refining the proposal
Converters improvement suggestions (tensorflow-onnx, Keras2Onnx) for better graph optimizations – Sica (IBM)	Converters, Operators	Higher functioning ops specifically targeting (LSTM/GRU in particular) have gotten support in tfonnx (converters). Efforts will continue as new opportunities rise

ONNX Roadmap Items (Status) - 2

Topics	Proposed SIGs	Current Status or Future Positioning
Address gaps with Opset conversions across broad set of models – Sabharwal (Intel)	Arch/Infra, Converters, Release	Past two ONNX releases have fixed subset of issues with Opset conversions/compatibility. Efforts to continue in future as newer Opsets get introduced
ONNX model zoo example for E2E distributed training scenario of large models – Esteves (Intel)	Model Tutorial	Identified as long term intercept
Define concept of federated learning for ONNX – Esteves (Intel)	Operators	Identified no new ONNX operators required, exploring further on solutions that are Framework/runtime agnostic
Improvements to shape inference implementation – McCarter (Lighmatter)	Arch/Infra	Submitter analyzing further on the Shape inference improvements to be targeted for future
Introduce ONNX model provenance & security to safeguard against manipulations – Karumanchi (Intel)	Arch/Infra, Model Tutorial, Pre-processing	Initial metadata fields defined to establish machine readable ONNX model provenance https://github.com/onnx/onnx/issues/3958
ONNX model zoo support for quantized and mixed precision models.– Karumanchi (Intel)	Model Tutorial, Operators	Related to ONNX model metadata field definition, will target couple of mixed precision models in zoo once Issue #3958 finalized

Thank You!

Q & A

Resources

https://onnx.ai

https://onnxruntime.ai

LF AI & DATA

ONNX

Backups

ONNX

Release Update

ONNX 1.11 Released

Release v1.11.0 · onnx/onnx (github.com)

ONNX v1.11.0 comes with following updates:

- Opset 16 introduced with new and updated operators
- Added Model hub (to pull pre-trained models from zoo)
- Compose utilities to create combined model with preprocessing & inference
- Functionbuilder utility to help create function ops
- Bugfixes and infrastructure improvements
- Documentation updates

Visit the release page on GitHub for more details

Thank you everyone for your countless hours of work!

ONNX 1.12 Released

Release v1.12.0 · onnx/onnx (github.com)

ONNX v1.12 comes with following updates:

- Opset 17 introduced with new and updated operators
- Shape inference enhancements
- Bugfixes and infrastructure improvements
- Documentation updates
- Add Python 3.10 and drop Python 3.6 support
- Drop support for x86 (32-bit) Linux due to low usage

Visit the release page on GitHub for more details

Thank you everyone for your countless hours of work!

ONNX – Model 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
- Operators
- Operator parameters

ONNX – 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 meaninfully further decomposed
  - Currently 124 in ai.onnx domain and 18 in ai.onnx.ml
  - Support many scenaries/problems areas including image classification, recomendation, natural language processing, etc.
- Custom ops
  - Ops specific to framework or runtime
  - Indicated by a custom domain name

ONNX – Versioning

Versioning in ONNX is done at 3 levesl

- IR version (file format): currently at version 5 (p.ex. Protobuf format is changing)
- Opset version: ONNX models declare which operator sets they require as a list of two-part operator ids (domain, opset_version)
- Operator version: A given operator is identified by a three-tuple: (domain, op_type, and op_version)

ONNX Runtime – Design Principles

- Provide complete implementation of the ONNX standard – implement all versions of the operators (since opset 7)
- Backward compatibility
- High performance
- Cross plattform
- Leverage custom accelerators and runtimes to enable maximum performance (execution providers)
- Support hybrid execution of the models
- Extensible through pluggable modules

LF AI & Data Foundation Interactive Landscape

The LF AI & Data Foundation landscape (png, pdf) is dynamically generated below. It is modeled after the CNCF landscape and based on the same open source code. Please open a pull request to correct any issues. Greyed logos are not open source. Last Updated: 2022-09-13T05:37:12.

You are viewing 330 cards with a total of 2,741,215 stars, market cap of $15.2T and funding of $19.2B.

Landscape | Card Mode | LF AI & Data Members | Companies Hosting Projects

60