PyTorch + oneAPI

Lucy Hyde, Sr. Program Manager, PyTorch Foundation

August 21, 2023

Agenda

- PyTorch Foundation
- Overview
- Members
- PyTorch 2.0
  - Generative AI
  - Parallelism
- Join us!

PyTorch Foundation

PyTorch Overview

A powerful and developer-centric approach to deep learning. Its flexibility, efficiency, and community support make it an ideal choice for constructing and training machine learning models.

- ➔ User friendly interface
- ➔ Interoperability and integration within Python ecosystem
- ➔ Flexibility simplifies development process and facilities easier debugging
- ➔ On the fly definition and modification of your models

PyTorch Foundation Members

amazon

AMD

Google

IBM

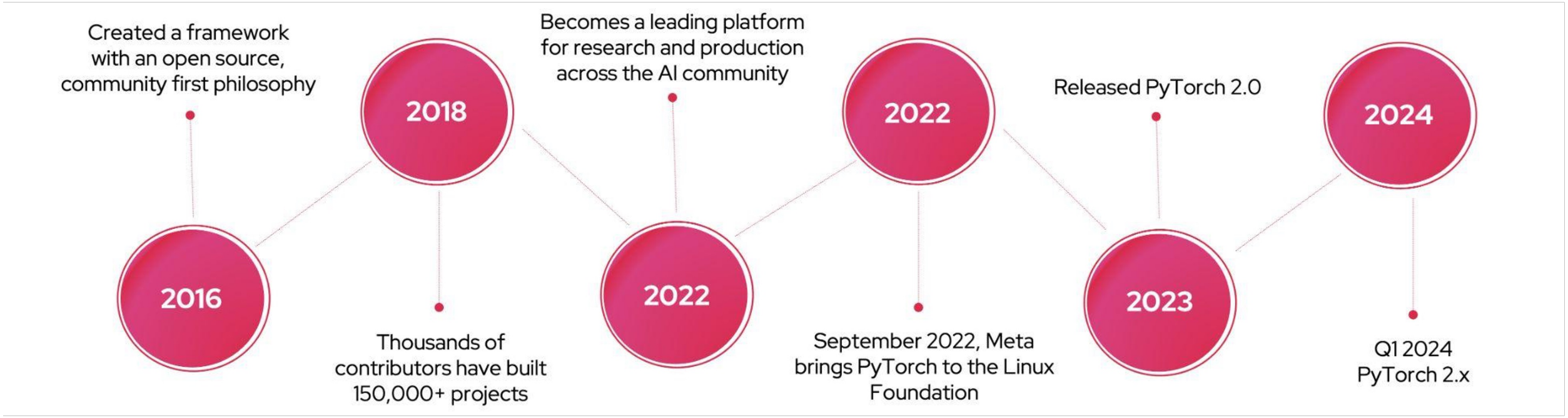
intel

Meta

Microsoft

NVIDIA

PyTorch Major Milestones



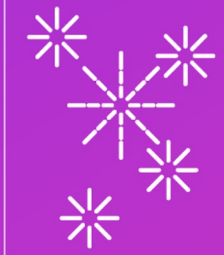
FIRST SIX MONTHS OF THE PYTORCH FOUNDATION

The number of commits across all repositories **increased 36%**

2,315 unique contributors in the last 6 months

2600k+ users created their first post on the discussion forum

Our YouTube channel boasts **34K followers** with **62.6K views** in the past month.



PyTorch 2.0 comprised of **4,500+ commits** and **400 contributors**

The Pytorch Mastodon account has attracted **1,200 followers**

The number of new community members contributing is **up by 18%**

Total technical contributions have **increased 23%**

The PyTorch 2022 Conference hosted **900+ attendees**

PyTorch 2.0

PyTorch 2.0

- Work with Dynamic Shapes
- Adapt your models on the fly
- Seamlessly adjust to varying input sizes
- Optimize resource utilization

PyTorch 2.0

- **Flexible** – developers can unlock new levels of flexibility and adaptability, and tackle complex tasks easily.
- **Distributed** – Supports distributed environments,
- **Scalable** – Seamlessly scale your PyTorch applications across multiple devices and nodes
- **Powerful** – Harness the power of distributed computing to tackle the most challenging problems.

PyTorch 2.0

- **Pythonic**- PyTorch is moving parts from C++ back into Python, making it faster.
- **torch.compile** is the main API for PyTorch 2.0, which wraps your model and returns a compiled model. It is a fully additive feature and 2.0 is 100% backward compatible.
- **TorchInductor** with Nvidia and AMD GPUs will rely on OpenAI Triton deep learning compiler to generate performant code and hide low level hardware details
- **Accelerated Transformers** introduce high-performance support for training and inference using a custom kernel architecture for scaled dot product attention (SPDA).

PyTorch + Generative AI

Accelerated Generative AI Diffusion Models with 2.0

- Out-of-the-box performance improvement for Generative Diffusion models by using the new torch.compile() compiler
- Optimized implementations of Multihead Attention integrated with PyTorch 2.
  - Compilation and fast attention implementations
    - Optimizations give up to 49% inference speedup relative to the original implementation without xFormers
    - 39% inference speedup relative to using the original code with xFormers depending on the GPU architecture and batch size.
    - Speedup comes without a need to install xFormers or any other extra dependencies

Parallelism in PyTorch

PyTorch and Data Parallelism

- PyTorch provides multiple tools/libraries to enable mixed parallelism in large language models
  - Built-in support for data parallelism through its torch.nn.DataParallel module
  - torch.nn.DataParallelCriterion extends DataParallel module to enable parallelized computations for loss functions
  - Parallelize arbitrary operations across multiple GPUs or devices using torch.nn.parallel.parallel_apply
- Custom parallelization techniques, including data/model partitioning, independent computations, synchronization, communication/aggregation, and parameter updates
- Simplifies the process of parallelizing computations across multiple GPUs
- Utilizes the available hardware resources to produce faster training times and improved performance

Introducing PiPPy: Pipeline Parallelism

- PiPPy aims to provide a framework for pipeline parallelism in PyTorch by allowing a model to be trained on larger datasets than utilizing data parallelism alone
- Pipeline parallelism is a process that speeds up the training of models through the following:
  - Divides the model into a sequence of stages (micro-batches)
  - Runs each stage of the pipeline on a separate GPU
- Current features include:
  - Automatic splitting of model code via torch.fx, enabling user to provide code as-is for parallelization without needing to make substantive changes
  - API for defining pipelined models and runtime system for scheduling/execution of pipeline models
  - Performance optimizations; composability with other parallelism schemes such as data parallelism or tensor splitting model parallelism, support for non-trivial topologies, including skip connections and tied weights/layers
- While new, PiPPy has been used to train LLMs, including Megatron-Turing NLG

Get engaged with the PyTorch Team

Dr. Ibrahim Haddad
Executive Director

Lucy Hyde
Sr. Program Manager

PyTorch.org

PyTorch 2023 Conference

PyTorch 2.0

Twitter/X

LinkedIn


PyTorch Conference
2023
October 16 - 17 | San Francisco, CA | #pytorchconf


PyTorch