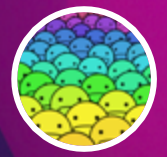
AI编译器-系列之PyTorch

PyTorch2.0 新特性

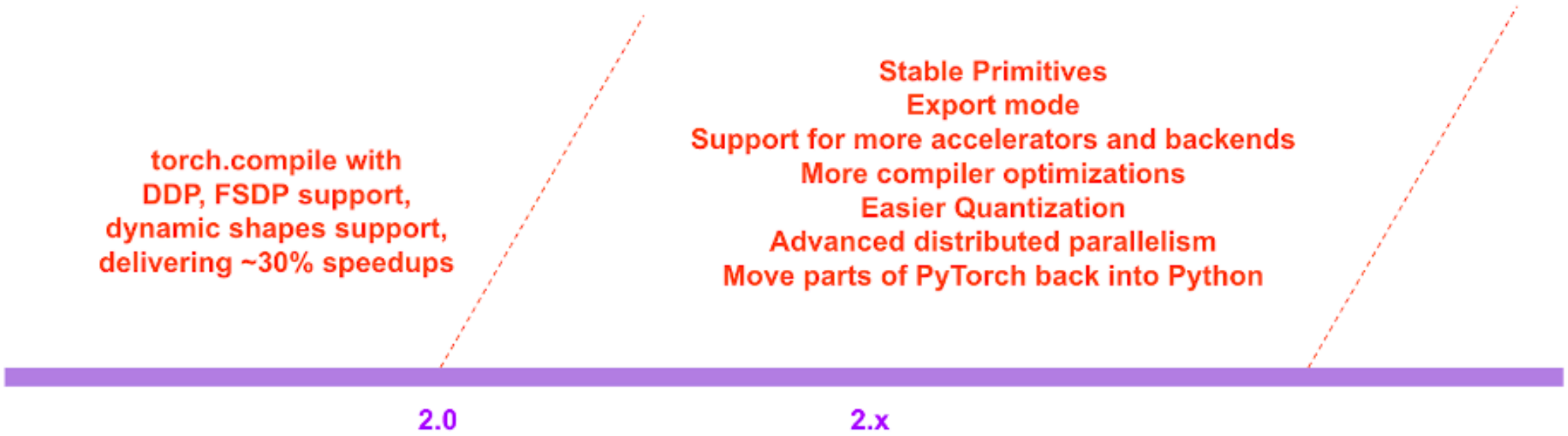


ZOMI

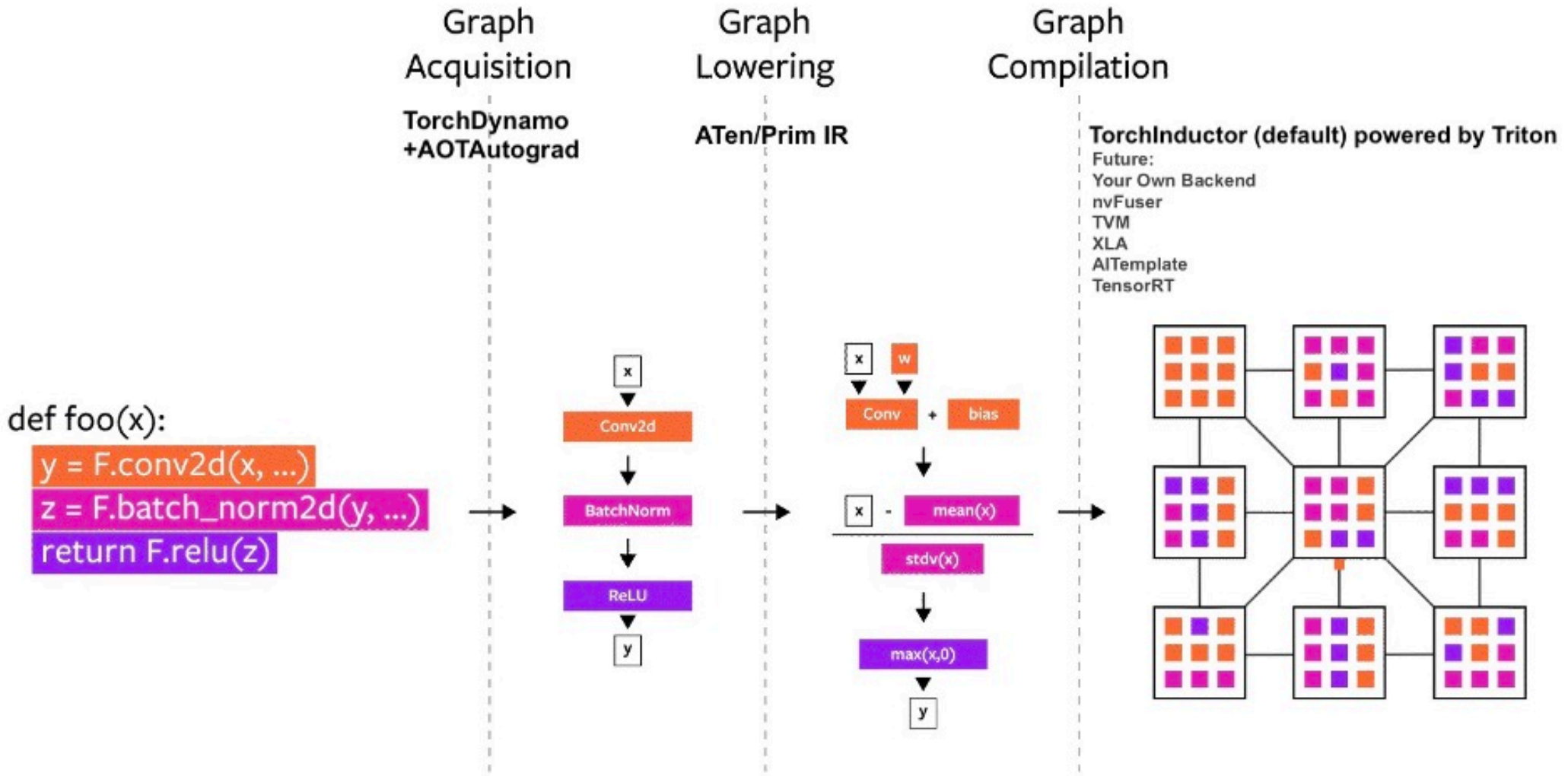
Talk Overview

1. **PyTroch 2.0 新特性**
   - 2.0 新特性回顾
   - PyTorch 2.0 安装与新特性使用
   - PyTorch 2.0 对厂商的启发和思考
2. **TorchDynamo 解读**
   - TorchDynamo 效果
   - TorchDynamo 实现方案
3. **AOTAutograd 解读**
   - AOTAutograd 效果
   - AOTAutograd 实现方案
4. **TorchInductor 新特性**
   - Triton 使用解读
   - Triton 深度剖析

未来版本

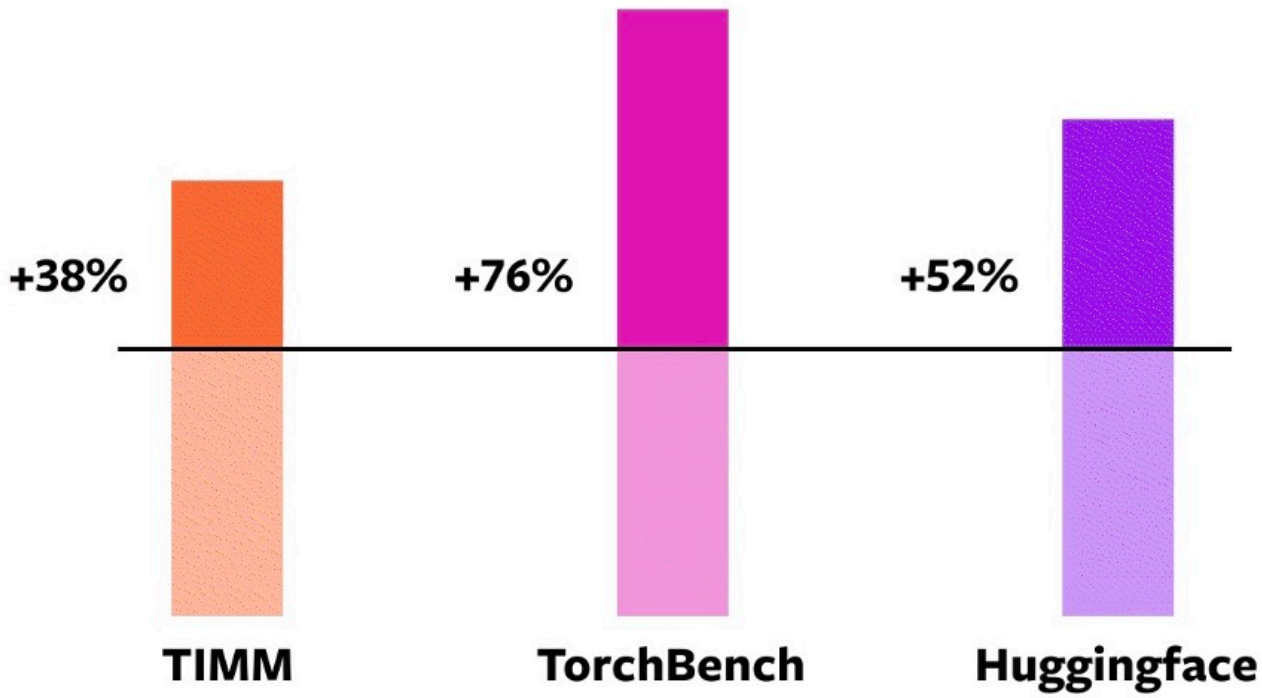


PyTorch编译过程



模型泛化性测试

- **46 models from HuggingFace Transformers:** It is most notable for its Transformers library built for natural language processing applications and its platform that allows users to share machine learning models and datasets.
- **61 models from TIMM:** a collection of state-of-the-art PyTorch image models by Ross Wightman.
- **56 models from TorchBench:** a curated set of popular code-bases from across github.



TorchDynamo: Acquiring Graphs reliably and fast

- Earlier this year, we started working on TorchDynamo, an approach that uses a CPython feature introduced in PEP-0523 called the Frame Evaluation API. We took a data-driven approach to validate its effectiveness on Graph Capture. We used 7,000+ Github projects written in PyTorch as our validation set. While TorchScript and others struggled to even acquire the graph 50% of the time, often with a big overhead, TorchDynamo acquired the graph 99% of the time, correctly, safely and with negligible overhead – without needing any changes to the original code. This is when we knew that we finally broke through the barrier that we were struggling with for many years in terms of flexibility and speed.

AOTAutograd: reusing Autograd for ahead-of-time graphs

- For PyTorch 2.0, we knew that we wanted to accelerate training. Thus, it was critical that we not only captured user-level code, but also that we captured backpropagation. Moreover, we knew that we wanted to reuse the existing battle-tested PyTorch autograd system. AOTAutograd leverages PyTorch's **torch_dispatch** extensibility mechanism to trace through our Autograd engine, allowing us to capture the backwards pass "ahead-of-time". This allows us to accelerate both our forwards *and* backwards pass using TorchInductor.

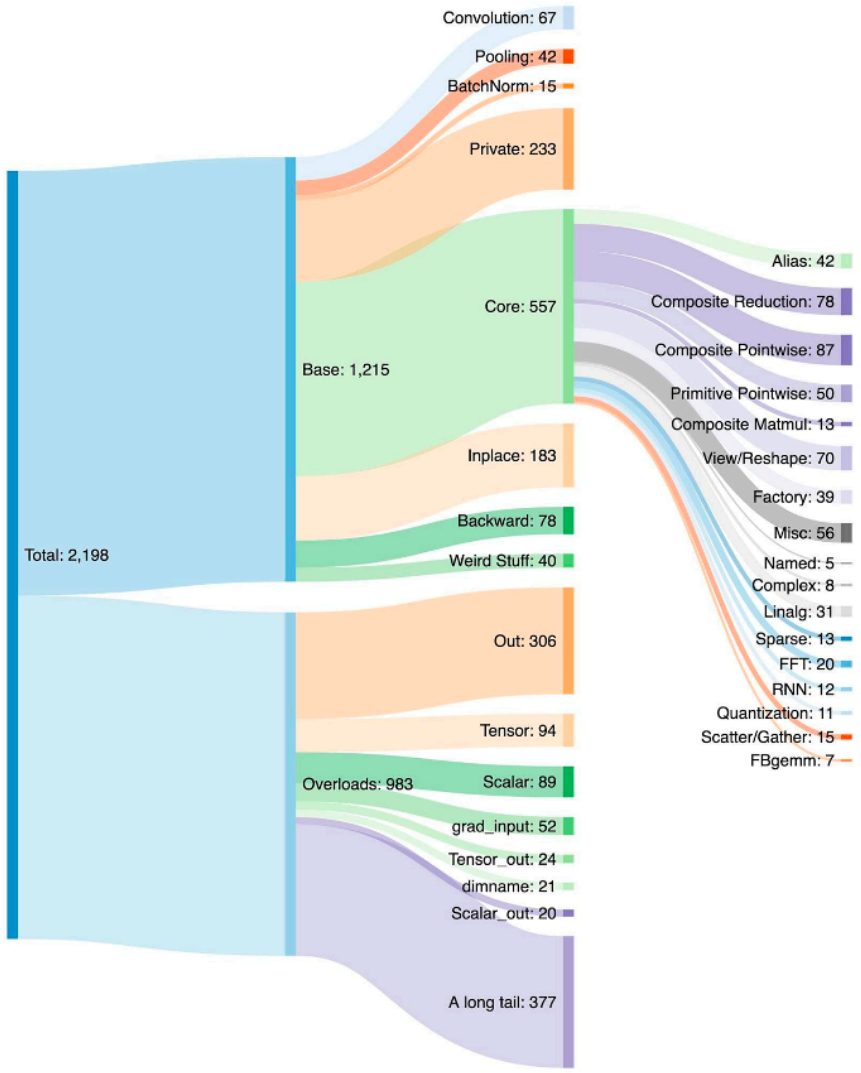
TorchInductor: fast codegen using a define-by-run IR

- For a new compiler backend for PyTorch 2.0, we took inspiration from how our users were writing high performance custom kernels: increasingly using the Triton language. We also wanted a compiler backend that used similar abstractions to PyTorch eager, and was general purpose enough to support the wide breadth of features in PyTorch. TorchInductor uses a pythonic define-by-run loop level IR to automatically map PyTorch models into generated Triton code on GPUs and C++/OpenMP on CPUs. TorchInductor's core loop level IR contains only ~50 operators, and it is implemented in Python, making it easily hackable and extensible.

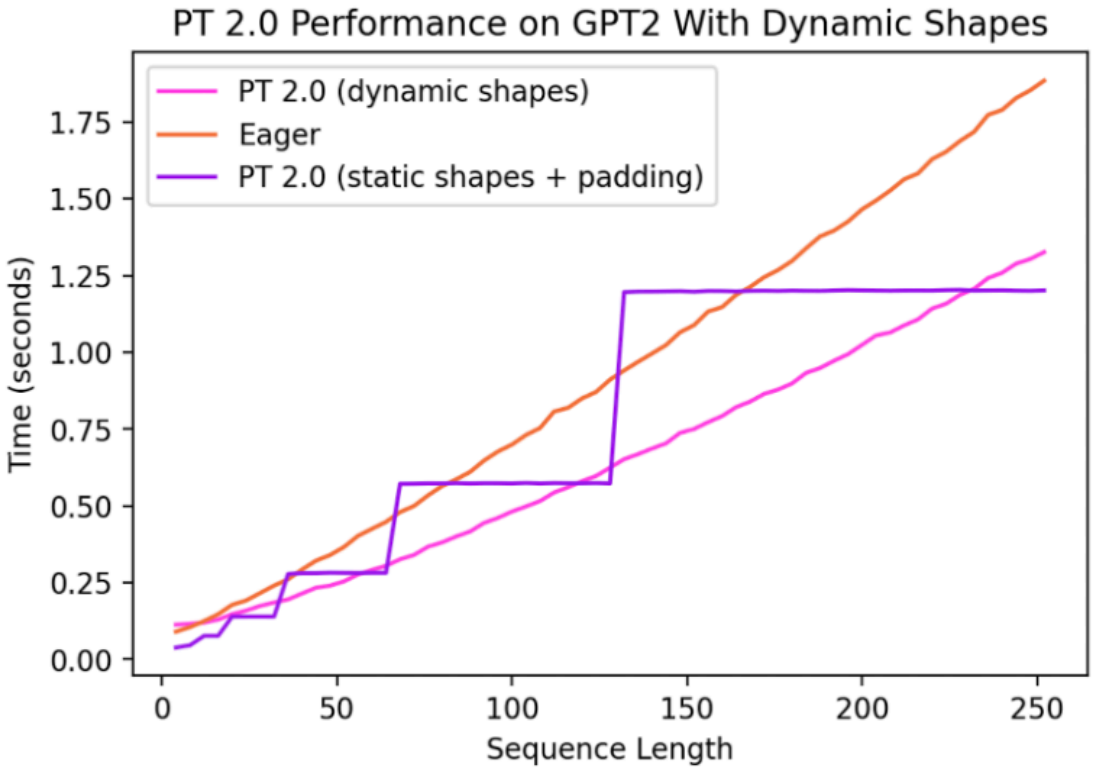
PrimTorch: Stable Primitive operators

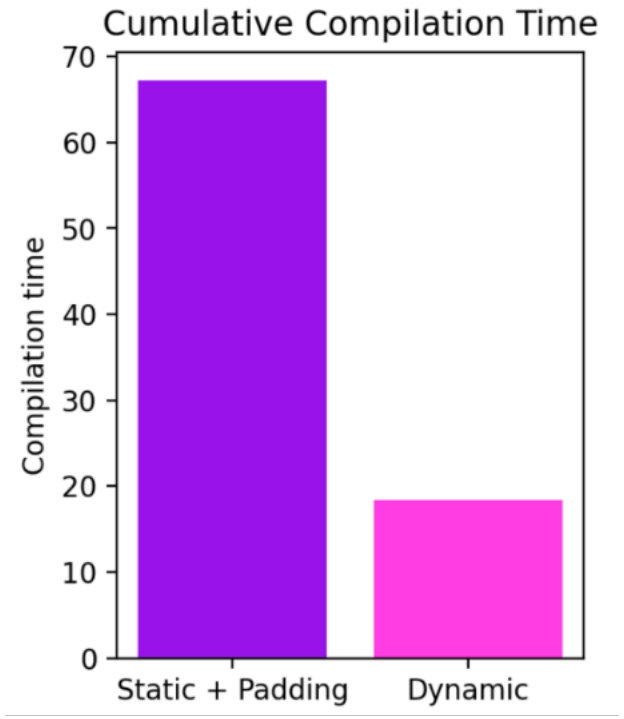
Writing a backend for PyTorch is challenging. PyTorch has 1200+ operators, and 2000+ if you consider various overloads for each operator.

- Prim ops with about ~250 operators, which are fairly low-level. These are suited for compilers because they are low-level enough that you need to fuse them back together to get good performance.
- ATen ops with about ~750 canonical operators and suited for exporting as-is. These are suited for backends that already integrate at the ATen level or backends that won't have compilation to recover performance from a lower-level operator set like Prim ops.

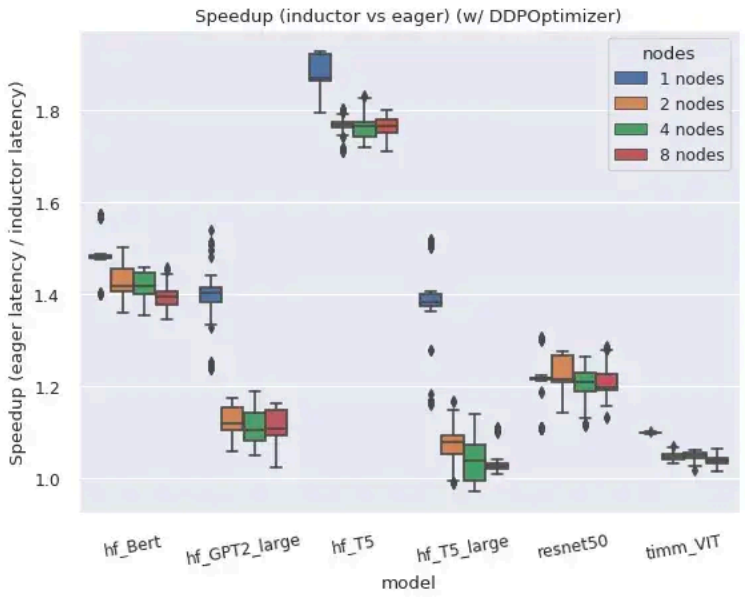


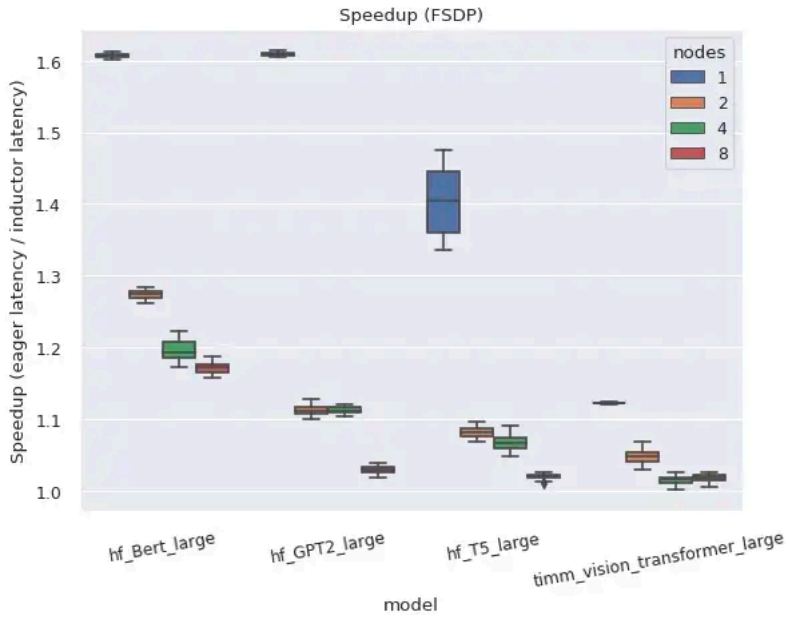
动态Shape性能

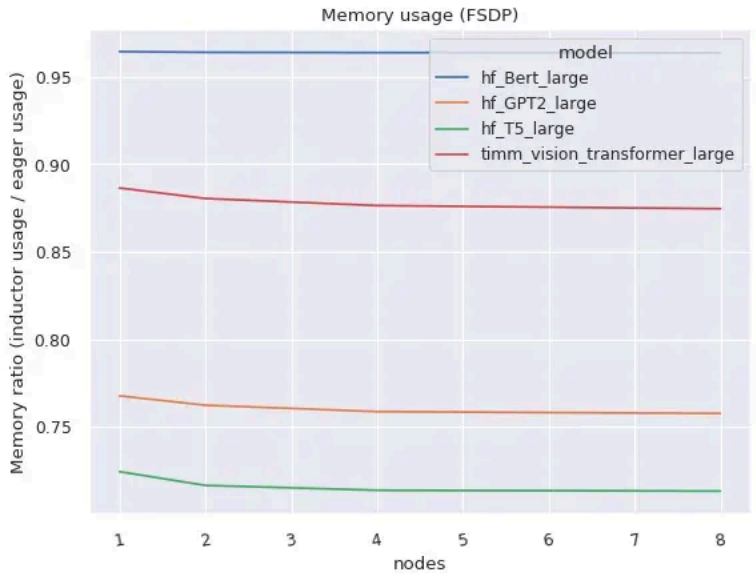




分布式支持 DDP FSDP







启发和思考

1. PyTorch 2.0 最大的改进引入 torch.compile() 图编译模式，PyTorch走别人正在走的路，会不会让别人无路可走？
2. TorchInductor 引入 OpenAI Triton 支持用 Python 取代 CUDA 编程来写底层硬件的代码，用户习惯是否埋单得让开发者去验证；
3. PrimTorch 把2000个算子用 250 个基础算子实现，更加生态环保，让新的厂商对接更加方便，聚焦250基础算子，如果要提升性能可以针对Aten的 750+算子进行融合优化

启发和思考

4. PyTorch 2.0 的官方反复强调了，引入了图编译模式后肯定完全向后兼容，TensorFlow 2.0首先我没有惹任何人，后向兼容性对框架 API 设计之初提出很大的挑战。
5. 9月份 PyTorch 加入 Linux 基金会后更加拥抱开源社区，从引入 Triton、后向兼容API、新特性充分考虑模型泛化性、收编算子等；
6. 既然 PyTorch 献身了，还有必要开发自己的 AI 框架吗？