

# **A journey into PyTorch, the Ecosystem, and Deep Learning Compilers**

**Luca Antiga**

**La Sapienza, Nov 11 2024**

**vmtk** DOCUMENTATION DOWNLOAD TUTORIALS

# The Vascular Modeling Toolkit

Supported by Oròbix srl

vmtk is a collection of libraries and tools for 3D reconstruction, geometric analysis, mesh generation and surface data analysis for image-based modeling of blood vessels.

[LEARN MORE](#)

ORÒBIX IT

WHO WE ARE

# The AI Service Company


At Oròbix, we implement and manage the lifecycle of artificial intelligence (AI) solutions. They can be newly designed or integrated into existing systems, spanning

## PyTorch: An Imperative Style, High-Performance Deep Learning Library

<b>Adam Paszke</b> University of Warsaw adam.paszke@gmail.com	<b>Sam Gross</b> Facebook AI Research sgross@fb.com	<b>Francisco Massa</b> Facebook AI Research fmasa@fb.com
<b>Adam Lerer</b> Facebook AI Research alerer@fb.com	<b>James Bradbury</b> Google jekbradbury@gmail.com	<b>Gregory Chanan</b> Facebook AI Research gchanan@fb.com
<b>Trevor Killeen</b> Self Employed killeent@cs.washington.edu	<b>Zeming Lin</b> Facebook AI Research zlin@fb.com	<b>Natalia Gimelshein</b> NVIDIA ngimelshein@nvidia.com
<b>Luca Antiga</b> Oròbix luca.antiga@orobix.com	<b>Alban Desmaison</b> Oxford University alban@robots.ox.ac.uk	<b>Andreas Köpf</b> Xamla andreas.koepf@xamla.com
<b>Edward Yang</b> Facebook AI Research ezyang@fb.com	<b>Zach DeVito</b> Facebook AI Research zdevito@cs.stanford.edu	<b>Martin Raison</b> Nabla martinraison@gmail.com
<b>Alykhan Tejani</b> Twitter atejani@twitter.com	<b>Sasank Chilamkurthy</b> Qure.ai sasankchilamkurthy@gmail.com	<b>Benoit Steiner</b> Facebook AI Research benoitsteiner@fb.com
<b>Lu Fang</b> Facebook lufang@fb.com	<b>Junjie Bai</b> Facebook jbai@fb.com	<b>Soumith Chintala</b> Facebook AI Research soumith@gmail.com

1912.01703v1 [cs.LG] 3 Dec 2019

# Deep Learning with PyTorch



Eli Stevens  
Luca Antiga  
Thomas Viehmann  
Foreword by Soumith Chintala

MANNING

Lightning AI Creators of PyTorch Lightning

Studios Docs Community Products Features Solutions About Pricing Login Start free

## MEET STUDIO

Turn ideas into AI, Lightning fast.  
The only intuitive, cohesive AI development platform.

Pretrain LLMs Data pipes Datasets Notebooks **Develop** Batch jobs Train Deploy Host AI apps

Build together on cloud GPUs via the browser or local IDE - **zero setup.**  
Auto-sleeps when idle, keeps environment intact across sessions.  
VSCode, Cursor, any IDE. Your cloud or ours.

[Launch a free Studio](#) [Request demo](#)


Powering AI at 10K+ orgs Microsoft Stability AI Playground AI Meta NVIDIA Connect any local IDE Swap GPUs live Autosaves environment (persists)

Acme AI / Image Generation / Stable Diffusion

```

main.py
main.py > DiffusionData
class DiffusionModel
def training_step
loss = torch
self.log("tr
return loss
def configure_optimizers
optimizer =
scheduler = torch.optim.lr_scheduler.SequentialLR(optimizer, gamma=0.2)
return [optimizer], [scheduler]

```



# Lightning AI

October 08, 2024

## PyTorch Foundation Technical Advisory Council Leadership

by Team PyTorch

We are pleased to announce the first-ever Chair and Vice Chair of the PyTorch Foundation's Technical Advisory Council (TAC). We are pleased to announce the first-ever Chair and Vice Chair of the PyTorch Foundation's Technical Advisory Council (TAC). We are pleased to announce the first-ever Chair and Vice Chair of the PyTorch Foundation's Technical Advisory Council (TAC).

**Luca Antiga** as the Chair and **Jiong Gong** as Vice Chair. Both leaders bring extensive experience and deep knowledge of the PyTorch community, and they are set to guide the TAC in its mission to foster an open, diverse, and innovative community.

### MEET THE NEW LEADERSHIP





# The Lightning OSS stack

## Data processing



LitData

Transform datasets at scale. Optimize datasets for fast AI model training.

## Training



PyTorch Lightning

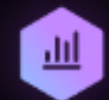
FLAGSHIP

Finetune and pretrain AI models on GPUs, TPUs and more. Focus on science, not engineering.



Lightning Fabric

Scale foundation models to 1000s of GPUs with expert-level control.



TorchMetrics

90+ easy-to-use PyTorch metrics optimized for distributed training.

## Inference



LitServe

Easily serve AI models Lightning fast. High-throughput serving engine for AI models.

## Optimization



Lightning Thunder

Make PyTorch models up to 40% faster! Thunder is a source to source compiler for PyTorch.



LitGPT

20+ high-performance LLMs with recipes to pretrain, finetune and deploy at scale.

# PyTorch

## GET STARTED

Choose Your Path: Install PyTorch Locally or Launch Instantly on Supported Cloud Platforms

[Get started >](#)

**PyTorch is an optimized tensor library for deep learning using GPUs and CPUs.**

### BLOG

Stay up-to-date on the latest news and technical topics from the PyTorch Foundation.

[Read more](#)

### PYTORCH 2.5

Featuring a new CuDNN backend for SDPA, improvements to TorchDynamo, regional compilation of torch.compile, and more.

[Learn more](#)

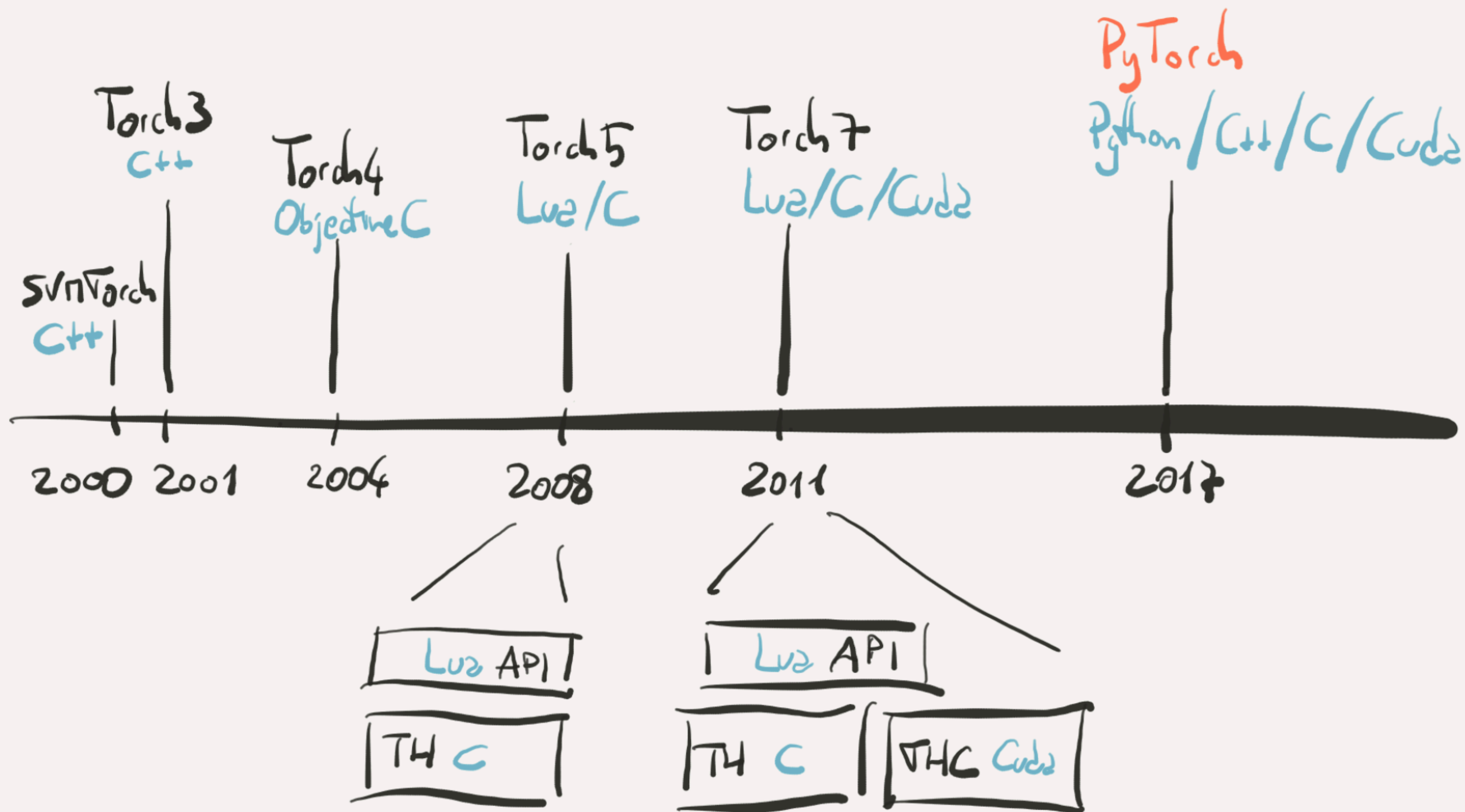
### MEMBERSHIP AVAILABLE

Become an integral part of the PyTorch Foundation, to build and shape the future of AI.

[Join](#)



# TORCH TIMELINE



# PyTorch

tensors on CPU, GPU

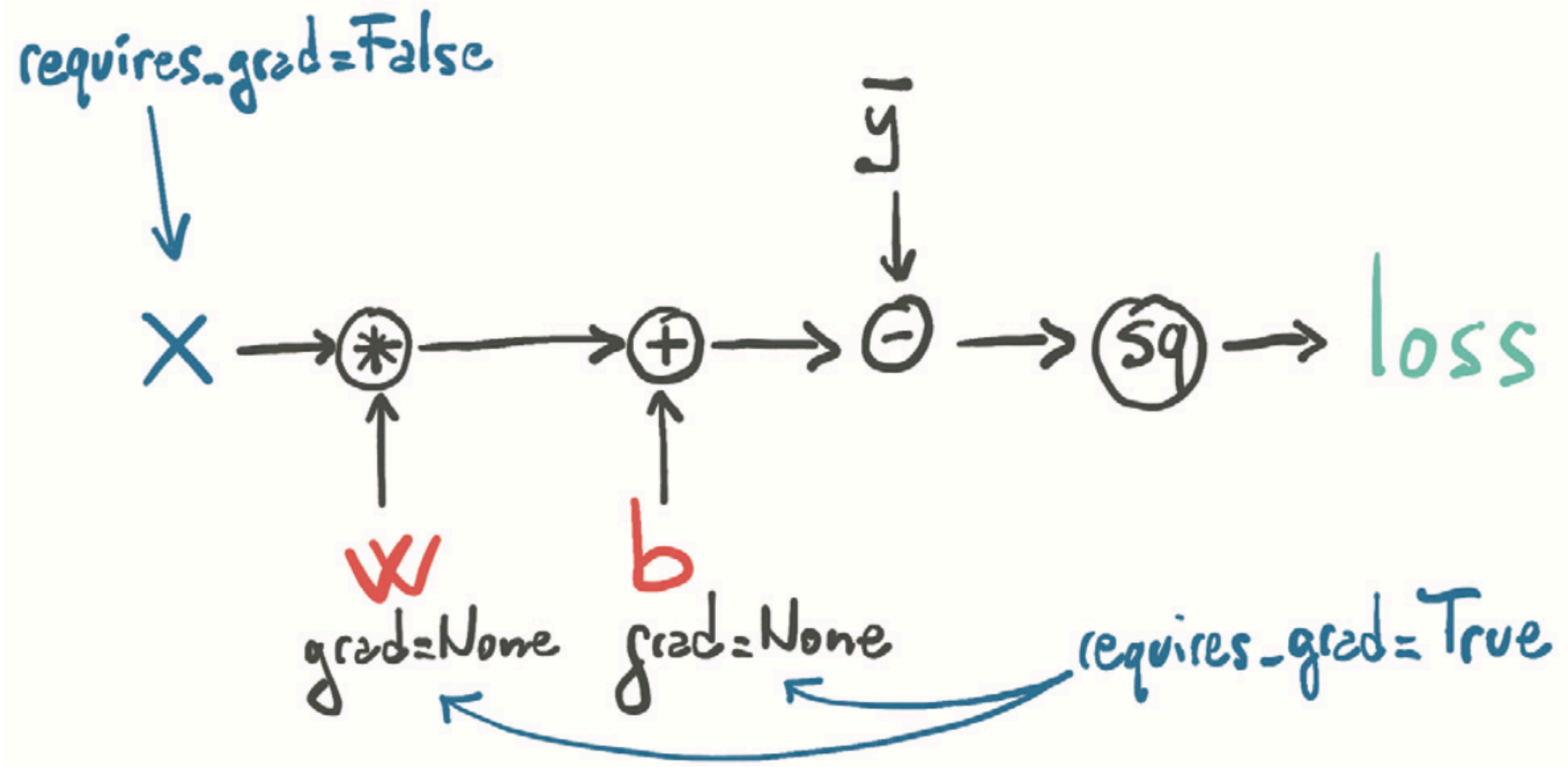
autograd

eager, *define-by-run*

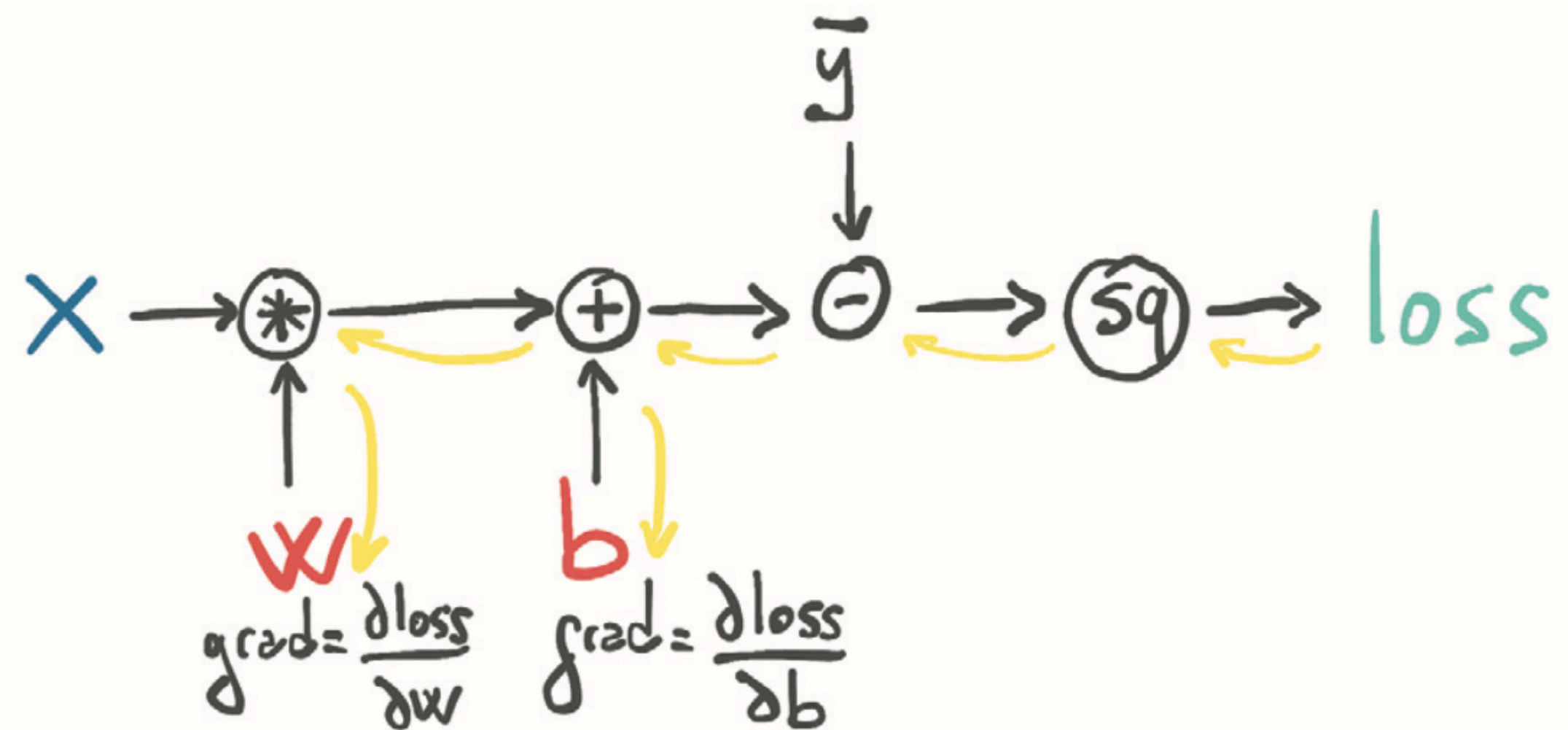
distributed



# Autograd



loss.backward()



# Dispatcher

`torch.add(a, b)`  
┌ `a.device`  
├ `a.layout`  
└ `a.dtype`



Dispatcher 🔍  
"`a.is_dense.cpu`"  
`a.is_cuda`  
`a.is_sparse`  
`a.is_quantized`

`add_cpu`

`a.dtype`

┌ `torch.long` → `add_cpu_kernel<int64_t>`  
├ `torch.float` → `" <float>`  
└ ...

...

...

...

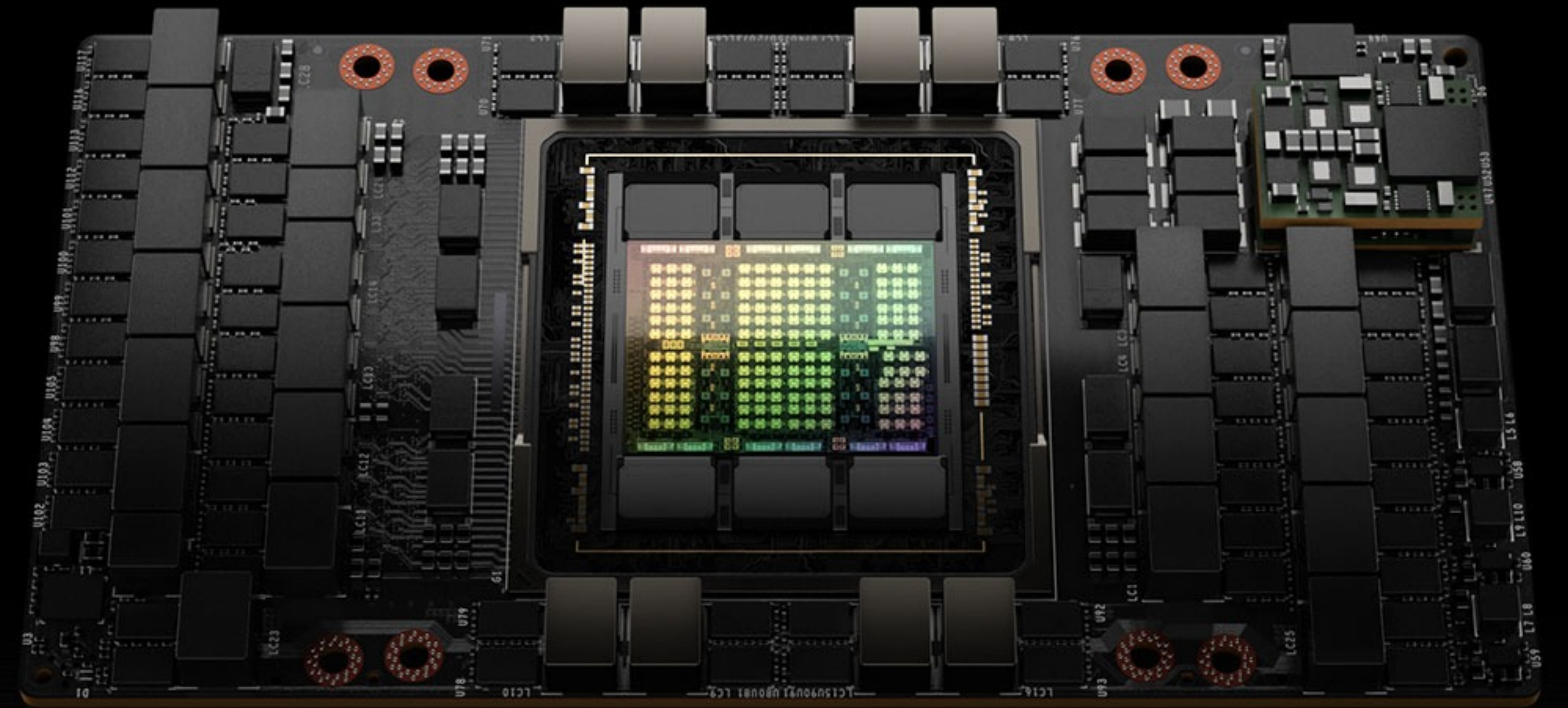


# Modern-day challenges

memory

memory bandwidth

parallelism



# Compilers

`torch.compile` (TorchDynamo + TorchInductor)

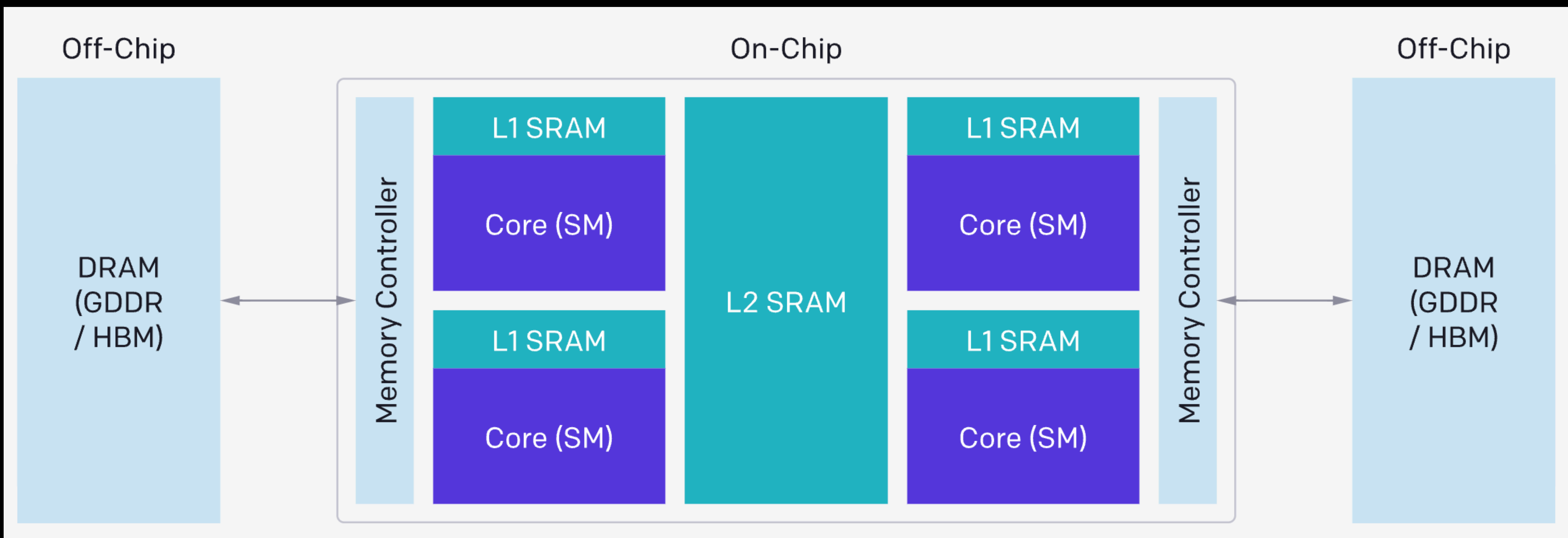
JAX + XLA, PyTorchXLA

Lightning Thunder



# OpenAI Triton

Open-source GPU programming for neural networks





Python

AST  
Visitor

```
@jit
def add(X, Y, Z, N):
    pid = program_id(0)
    idx = pid * 512 + arange(512)
```

Triton-IR

Triton  
Compiler

LLVM-IR

libLLVM

PTX

```
def void add(i32* X .aligned(16) , i32* Y .aligned(16) , i32* Z .aligned(16) , i32 N) {
entry:
    %0 = get_program_id[0] i32;
    %1 = mul i32 %0, 512;
    %3 = make_range[0 : 512] i32<512>;
    %4 = splat i32<512> %1;
    %6 = add i32<512> %4, %3;
    %9 = splat i32<512> N;
    %11 = icmp_slt i1<512> %6, %9;
    %14 = splat i32*<512> X;
    %16 = getelementptr i32*<512> %14, %6;
    %19 = broadcast i1<512> %11;
```

```
.visible .entry add(
    .param .u64 add_param_0, .param .u64 add_param_1,
    .param .u64 add_param_2, .param .u32 add_param_3
)
.maxntid 128, 1, 1
{
    .reg .pred    %p<4>;
    .reg .b32    %r<18>;
    .reg .b64    %rd<8>;
    ld.param.u64    %rd4, [add_param_0];
    ld.param.u64    %rd5, [add_param_1];
    mov.u32        %r13, %tid.x;
    ld.param.u32    %r14, [add_param_3];
```



Lightning Thunder

# Optimizing training or inference requires modifying computations

Model-specific

Implementation-specific

Hardware-specific

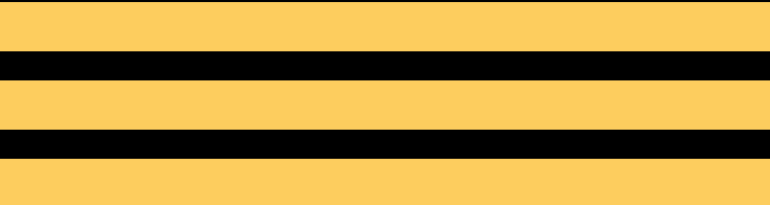
Topology-specific



Often performance comes from control  
(fusions, memory allocations, offloading, comms overlap)

for a specific model on specific hardware

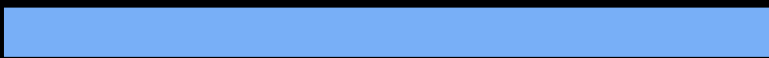
nn.Module



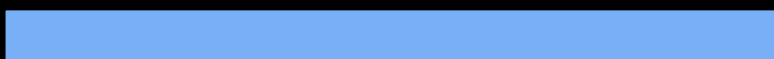
fusion



precision



specialized  
kernels



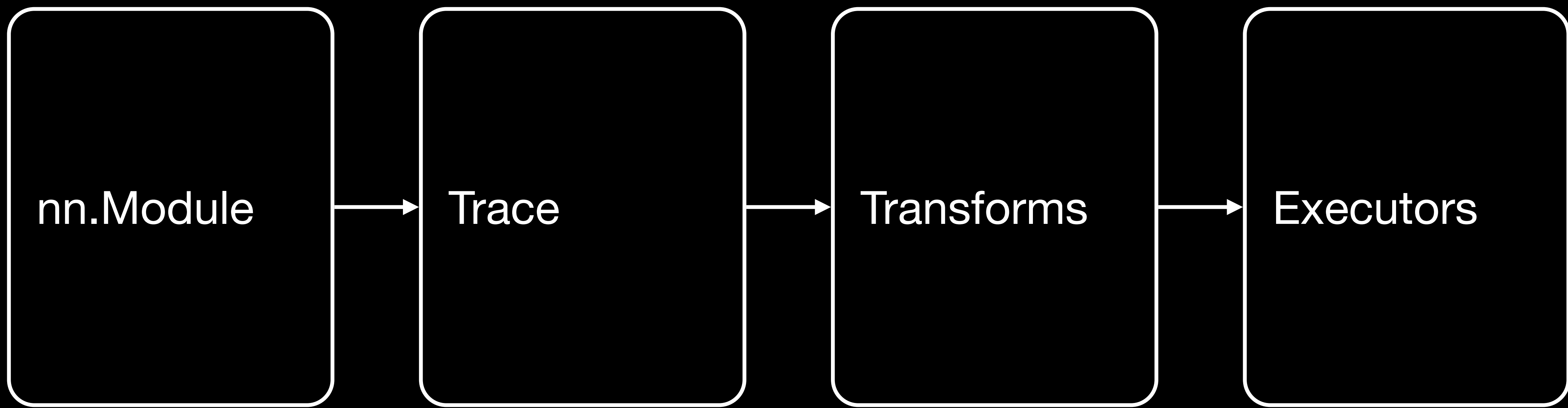
distributed

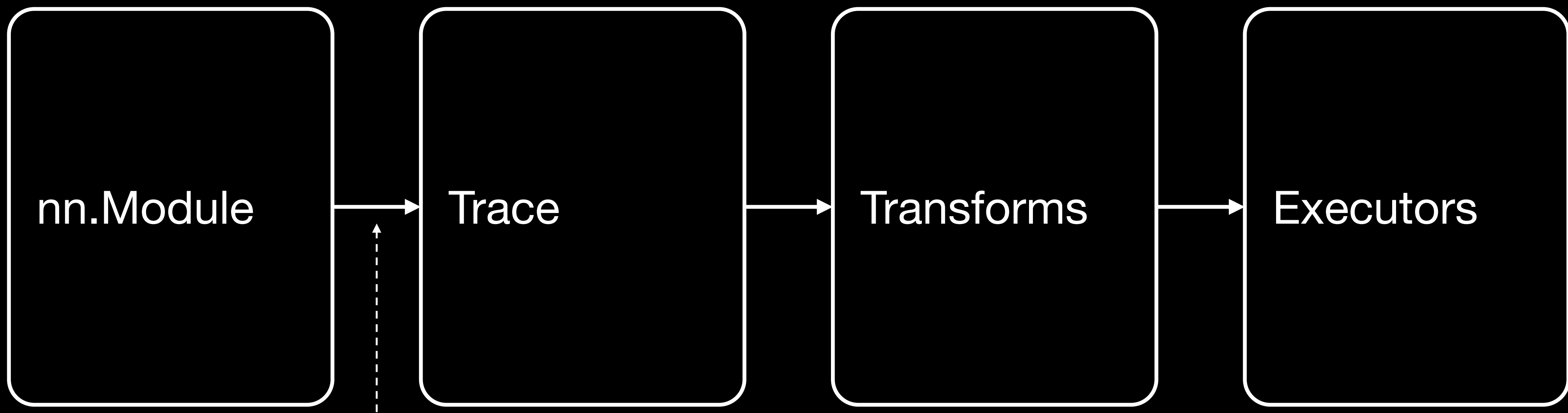


https://github.com/Lightning-AI/lightning-thunder

The screenshot shows the GitHub repository page for `Lightning-AI / lightning-thunder`. The repository is public and has 1.1k stars, 70 forks, and 33 unwatchers. The main branch is selected, and there are 35 branches and 1 tag. The repository description states: "Make PyTorch models up to 40% faster! Thunder is a source to source compiler for PyTorch. It enables using different hardware executors at once; across one or thousands of GPUs." The repository includes a README, Apache-2.0 license, and activity feed. A purple box is overlaid on the repository content, containing the command: `pip install lightning-thunder`.

File/Folder	Commit Message	Time
<code>.azure</code>	switch ci jobs to 2.4 (#1100)	2 weeks ago
<code>.github</code>	Remove thunder.compile and tests (#1114)	2 weeks ago
<code>dockers</code>	add lapack to build (#910)	2 months ago
<code>docs</code>	remove functional jit (#1120)	2 weeks ago
<code>examples</code>		
<code>notebooks</code>		
<code>requirements</code>	Bump litgpt from 0.3.1 to 0.4.11 (#1078)	2 weeks ago
<code>scripts</code>	drop standalone testing in favour of plain pytest-3 for di...	4 months ago
<code>thunder</code>	fix apex fused rms (#1166)	yesterday
<code>.codecov.yml</code>	ci/lint: yaml formatting with prettier (#141)	5 months ago
<code>.git-blame-ignore-revs</code>	ignore #772 in blame view (#773)	2 months ago
<code>.gitignore</code>	ci/ipynb: test some notebooks (PR2385)	6 months ago
<code>.pre-commit-config.yaml</code>	bump max file size (#970)	last month
<code>LICENSE</code>	Initial commit	2 years ago
<code>MANIFEST.in</code>	Add karpathy/llama2.c example (PR1204)	last year
<code>Makefile</code>	ci/docs: set Sphinx public theme as alternative (#236)	5 months ago
<code>README.md</code>	update nvfuser install instructions (#985)	last month

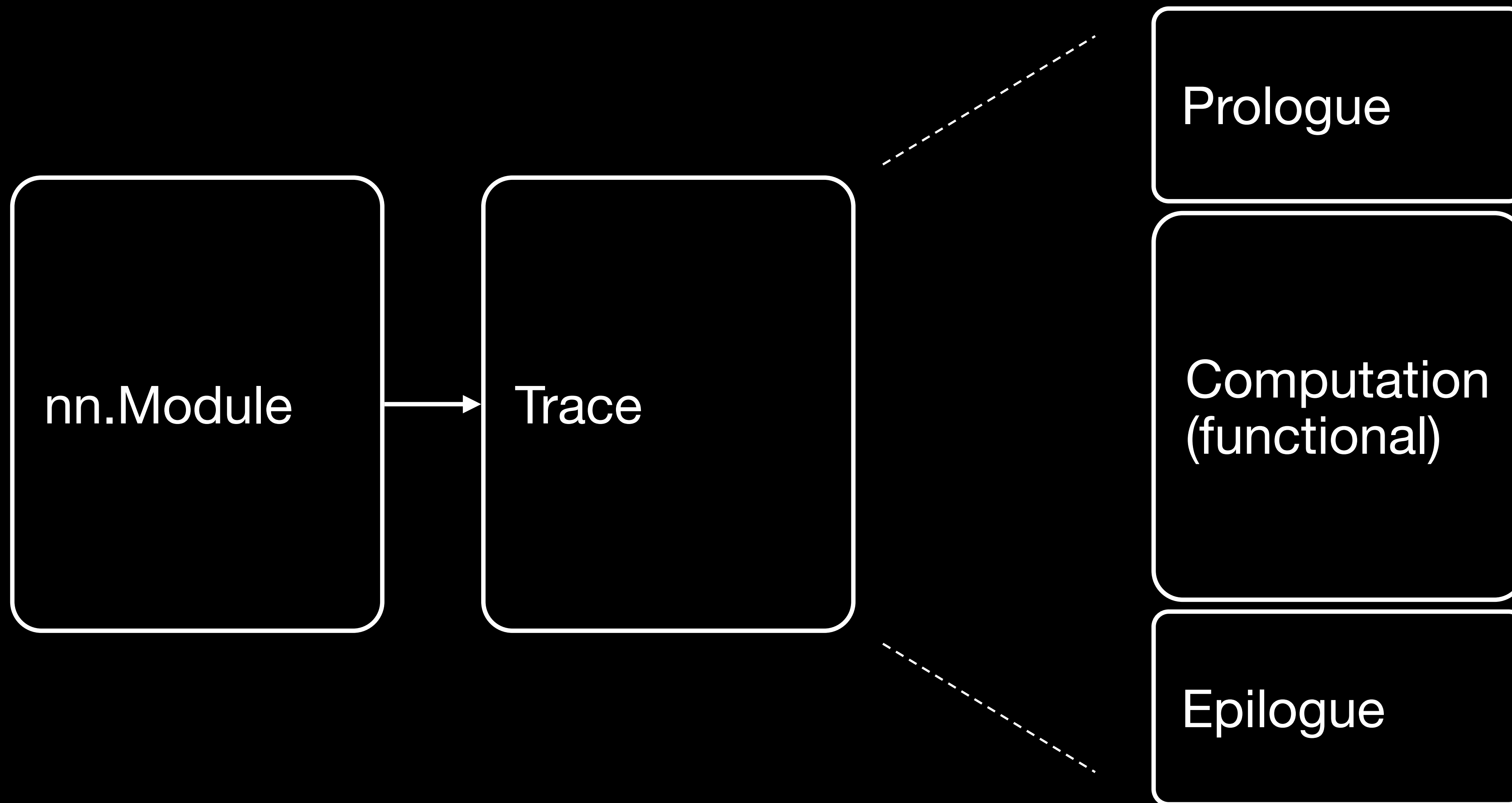




interpreter.py





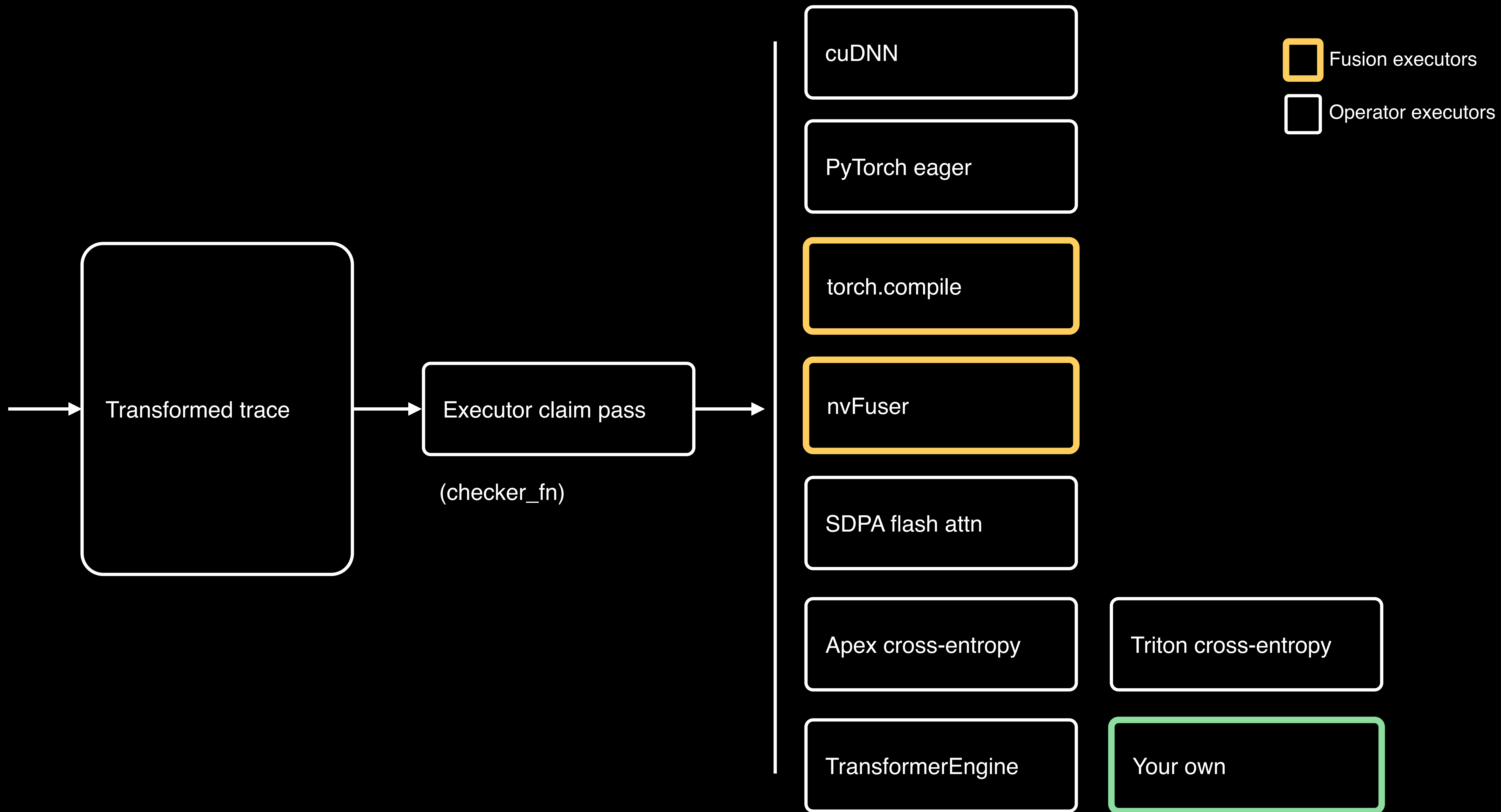


## Transforms:

- Grad
- Autocast
- Quantization
- Offloading
- Distributed (DDP, FSDP, TP)
- CUDAGraph
- ...



Composable



# Thunder with PyTorch

Can use `torch.compile` as an executor

Coming up:

available as a `torch.compile` backend

```
import torch
from thunder.dynamo import ThunderCompiler
backend = ThunderCompiler()

x = torch.ones(2, requires_grad=True)

@torch.compile(backend=backend)
def func(x):
    x = torch.sin(x)
    if x.sum() > 0:
        return x + 1
    else:
        return x - 1

out = func(x)
```



# The Thunder Sessions

Deep learning compilers and  
how the sausage is made



FRIDAYS @ 11AM EST





Lightning **AI**

