

How PyTorch Powers Open Foundation Models

PyTorch Conference | October 20, 2026

Songhao Jia
Software
Engineer, FAIR

Laura Gustafson
Research
Scientist, FAIR

Elisa Cascardi
Research Program Manager,
FAIR





CONFERENCE

—NORTH AMERICA 2026—

Supporting Muse Glimmer with ExecuTorch

Songhao Jia

What is ExecuTorch?

- PyTorch's native on-device inference stack



- Export PyTorch models with flexible quantization schemes.
- Optimize for CPUs, GPUs, NPUs and DSPs.
- Run in a lightweight, embeddable runtime.

Enabling Muse Glimmer in ET

- 30B text and vision model with up to 128K context.
- Model export and runnable examples on NVIDIA GPUs and Apple Silicon.
- Flexible quantization schemes for local memory budgets.
- PyTorch reference checks and automated export and runtime tests.

Quantization schema support in ExecuTorch

Quantization choices balance model quality, memory and speed.

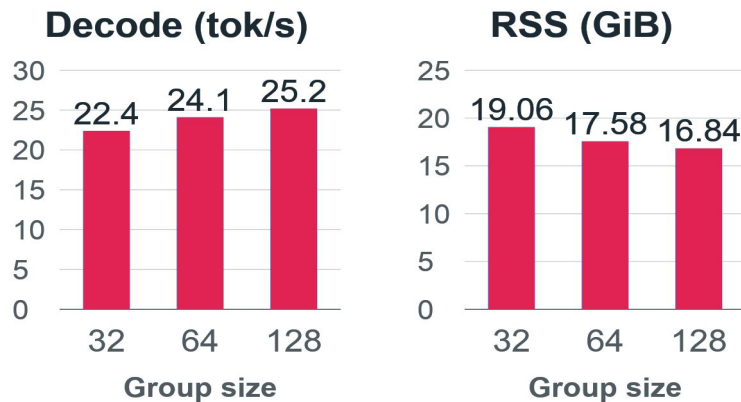
Quantization experiments

4-bit or mixed 4/6-bit / 4/8-bit.
Group sizes: 32, 64 and 128.

Quality and performance checks

Per-token quality vs bf16 PyTorch.
Prefill / decode speed and memory use.

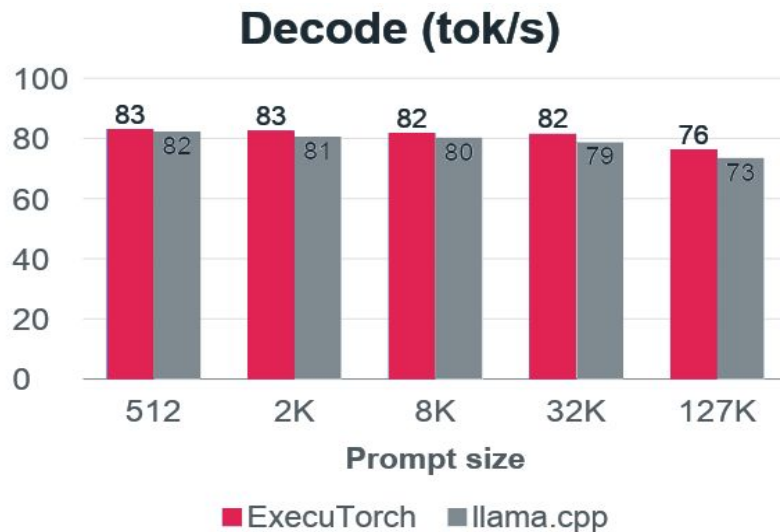
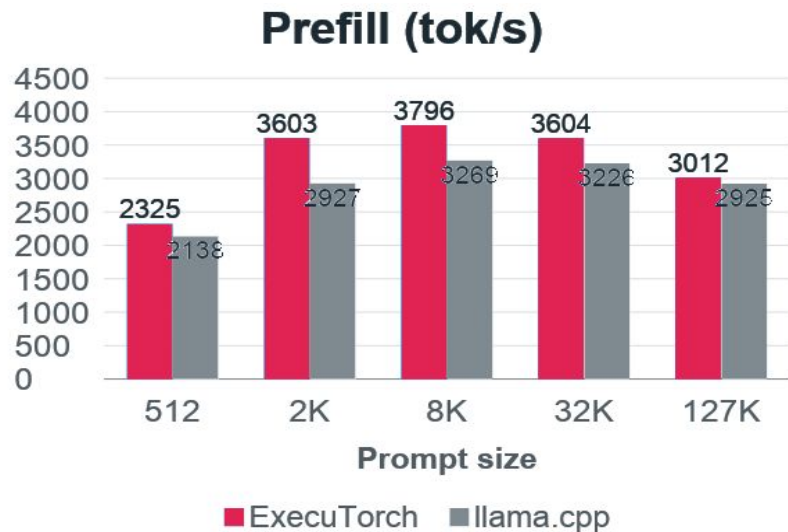
Controlled group-size sweep



M4 MacBook Pro (32 GiB), 512-token prompt. Sweep: mixed 4/6-bit weights.
RSS: resident memory. Larger groups reduce weight storage in this sweep.

Muse Glimmer solo performance: CUDA

RTX 5090

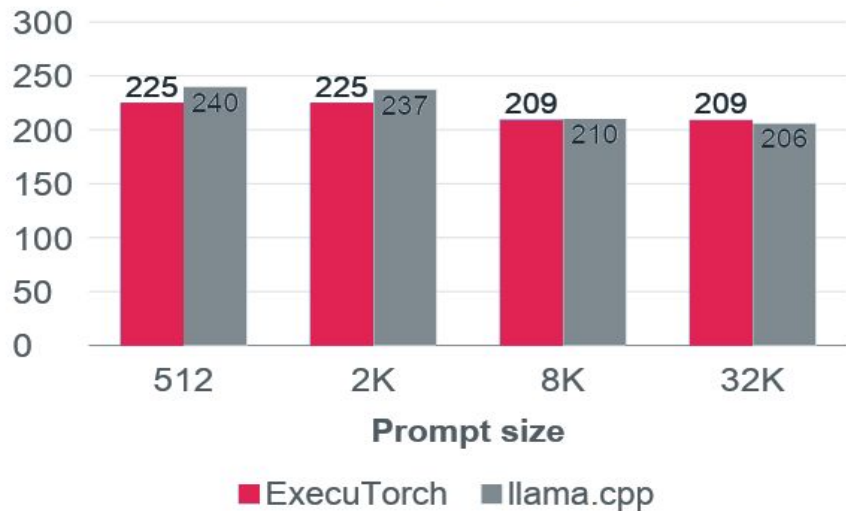


Prefill processes the prompt. Decode generates new tokens. Decode output length: 512 tokens.
Prompt size: 512–130,048 tokens. Mean of runs 3–7. Prefill activations: 16-bit.

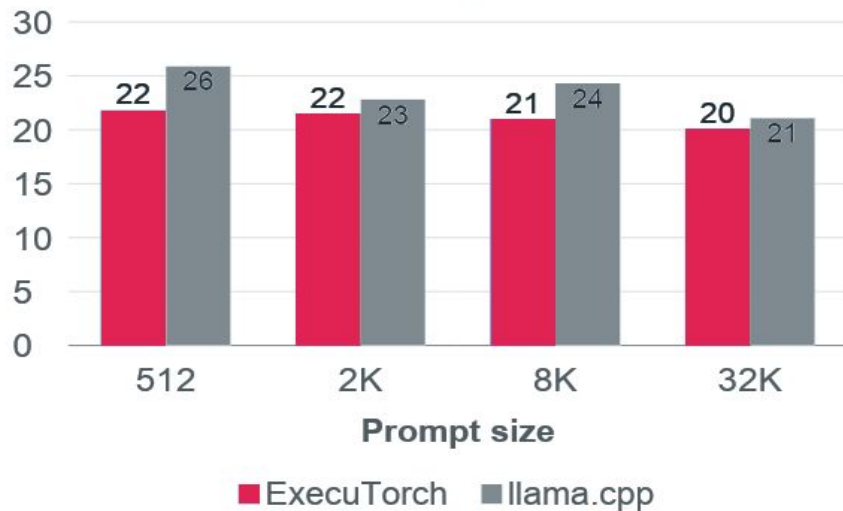
Muse Glimmer solo performance: MLX

MacBook Pro (M4)

Prefill (tok/s)



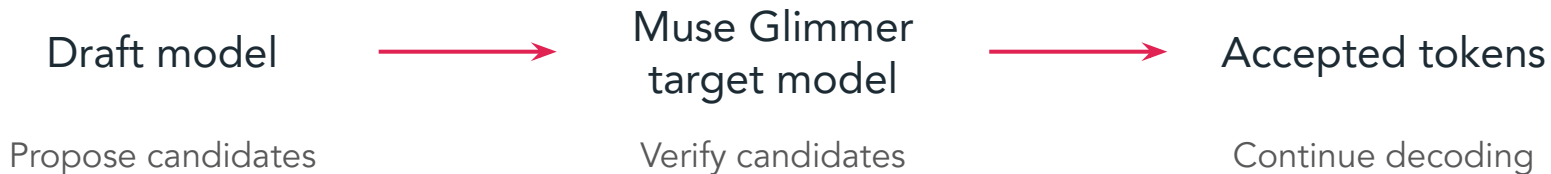
Decode (tok/s)



Prompt size: 512, 2K, 8K and 32K tokens. Decode output length: 512 tokens.

Runtime support for MG workloads

A draft model proposes several tokens. Muse Glimmer checks them together. ExecuTorch runs both models and manages verification and cache rollback.



DFlash

Muse Glimmer results

EAGLE3

Explored in ET

DFlash2

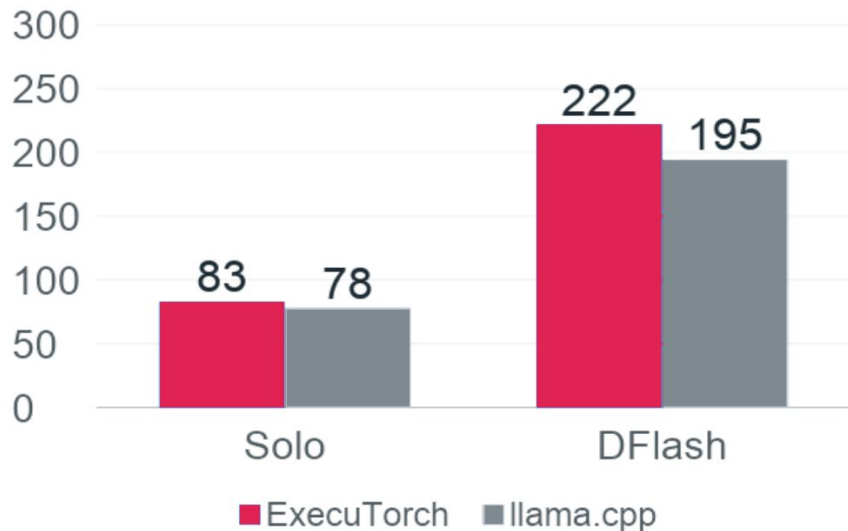
Support in progress

Open-source contributors are welcome to extend speculative decoding in ExecuTorch.

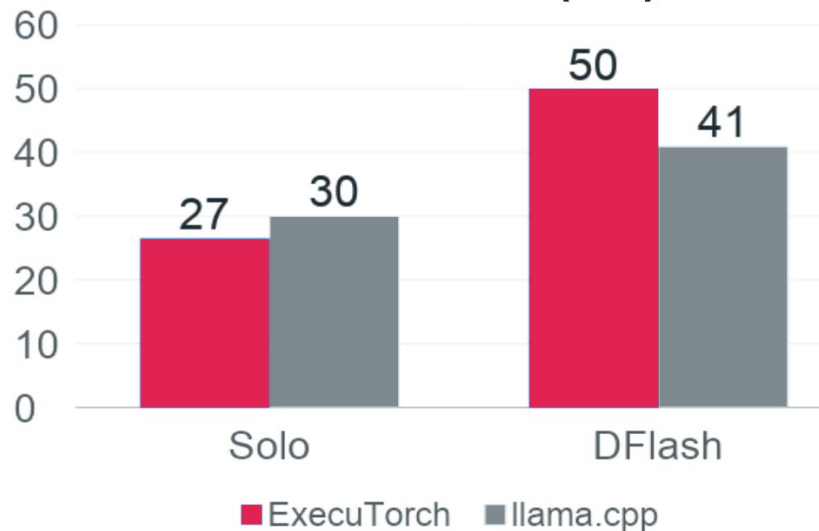
Muse Glimmer DFlash performance

Decode throughput (tok/s)

RTX5090



MacBook Pro (M4)



7-prompt mean, average of runs 3-7.

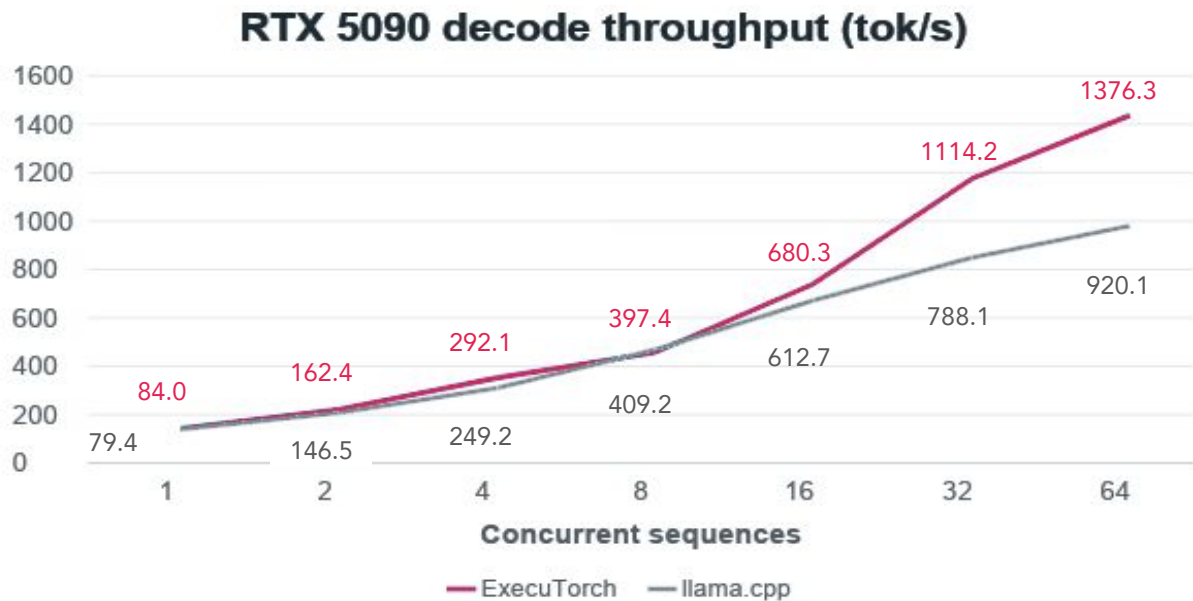
Runtime support for MG workloads

- KV cache stores the model's attention history across turns.
- ExecuTorch manages cache growth, rewind and reset at runtime.
- Prefix reuse avoids recomputing shared system prompts and tool schemas.

Runtime support for MG workloads

- Continuous batching combines requests from concurrent sessions.
- Each session keeps its own conversation state across agent turns.
- An OpenAI-compatible local server connects agents to Muse Glimmer.

Muse Glimmer batching throughput

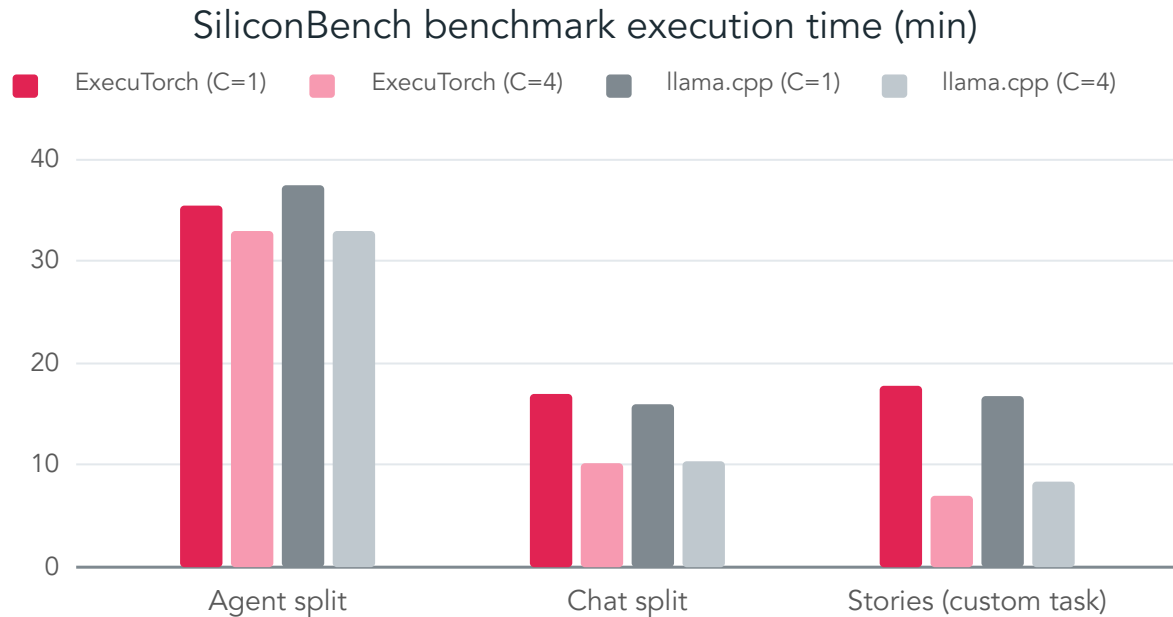


1.50x
vs llama.cpp
at 64 sequences

1–64 concurrent decode requests, 128-token prefix. Mean of runs 3–7.

Muse Glimmer batching performance: MLX

On M4 Max, ExecuTorch continuous batching performance scales comparably to llama.cpp across a variety of tasks



Lower execution time is better. C = concurrent requests.

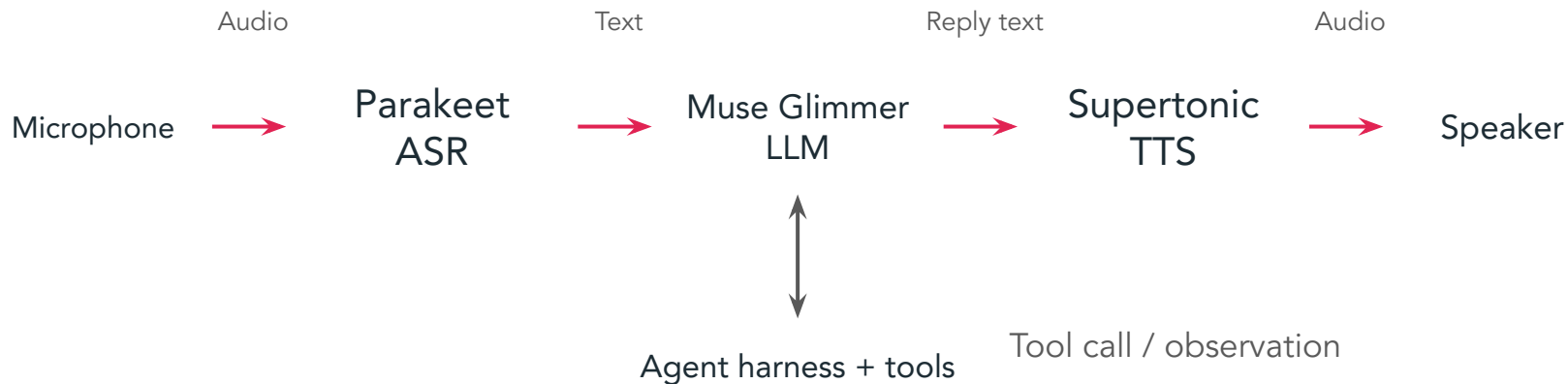
Muse Glimmer agent pipeline

Long context, repeated tool calls and concurrent sessions

Serving	OpenAI-compatible API and isolated agent sessions
Execution	Continuous batching and speculative decoding
State	Runtime KV cache control and prompt prefix reuse
Model	Model export, quantization and hardware acceleration

Demo

Local speech recognition, Muse Glimmer and speech synthesis run entirely on ExecuTorch.



Do

Takeaways

- ExecuTorch runs local models with reusable context and efficient execution.
- Muse Glimmer connects those capabilities to a complete local voice agent.

ExecuTorch Docs



GitHub Repository



Discord





CONFERENCE

—NORTH AMERICA 2026—

Segment Anything

Laura Gustafson

What is Segment Anything?

Open models to detect, segment and track any object in images and video

SAM 1 · 2023

Click a point,
get a mask



SAM 2 · 2024

Segment and track
objects in video



SAM 3 · 2025

Text prompts: find
every instance



SAM 3.1 · March 2026

Track 16 objects
in one pass



ModelAPI · Sept 2026

SAM 3.1, available via API

Open from day 1

Everything you need to build on SAM

93M+

model downloads

85K+

GitHub stars

#1

trending on Hugging Face

Weights + code

Checkpoints, inference and fine-tuning code under a permissive license

Data + benchmarks

SA-1B, SA-V and SA-Co datasets for the whole field to build on

Available via API

SAM 3.1 is available on Meta Model API: no deployment needed

Downloads and stars are SAM-family totals.

PyTorch made SAM fast

The PyTorch team sped up SAM using only native PyTorch features

8x

faster SAM 1 inference

torch.compile, SDPA, custom Triton kernels, int8 quantization, 2:4 sparsity and nested tensors, with no loss of accuracy

Segment Anything, Fast · PyTorch Conference 2023

Up to 13x

lower SAM 2 p90 latency

torch.export + AOTInductor for fast cold starts, reduced precision, batched prompts and GPU preprocessing on autoscaling cloud

Segment Anything 2: Fast and furious inference · 2024

The community builds on it too: HQ-SAM, EfficientSAM, MobileSAM, sam.cpp

From research to production

From a text prompt to tracked masks, fast enough to ship

SAM 3: real-time segmentation

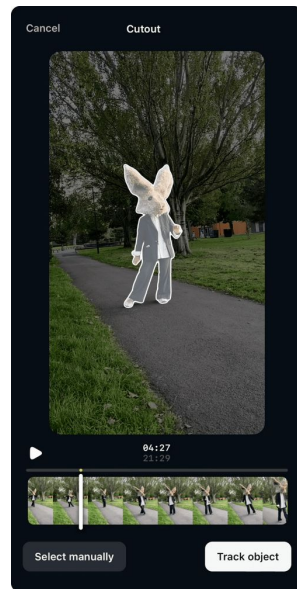
30 ms per image with 100+ objects on one H200;
near real-time video 30 FPS for ~5 objects

SAM 3.1: 2x video throughput

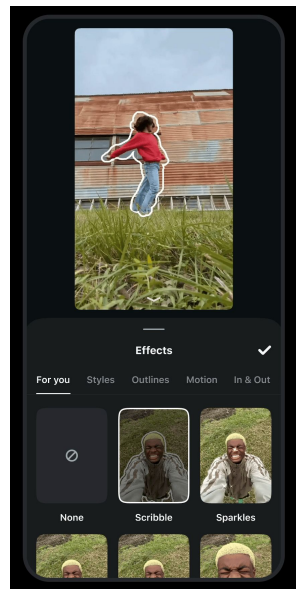
Object multiplexing tracks up to 16 objects in one
forward pass: 16 → 32 FPS on a single H100

In production, and open to everyone

Powers one-tap effects in Instagram Edits.



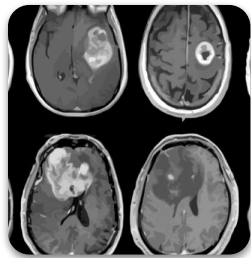
Edits · Cutouts (SAM 2)



Edits · Effects (SAM 3)

SAM is used across domains

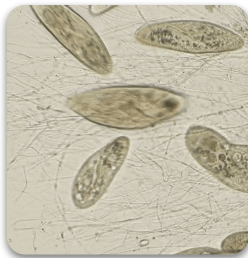
Zero-shot segmentation for image & video is now a standard building block



Radiology



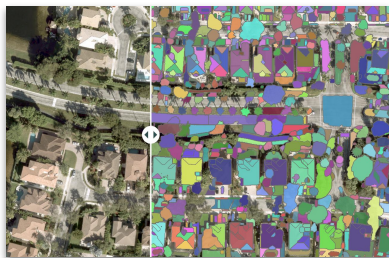
Autonomous driving & robotics



Cell biology



Live player tracking



Geospatial analysis

Science

SA-FARI: (with Conservation X Labs & Osa Conservation) 10K+ camera-trap videos of 100+ species

Collaborated with MBARI to add masks FathomNet database

Developers

Use the Meta Model API for SAM 3.1

Fine-tune and deploy on Hugging Face, Roboflow and AWS SageMaker.

Takeaways



SAM 3.1 Model



SAM 3.1 API

Open the full stack

Release weights, code and data,
and the community builds on it

Design for fast inference

Compile-friendly, sync-free code
pays off when you deploy

One model, research to product

SAM-3.1 ships in Edits, the Model API, and
is open-weight



CONFERENCE

—NORTH AMERICA 2026—

Open Source Isn't A Given, It's a Design Decision

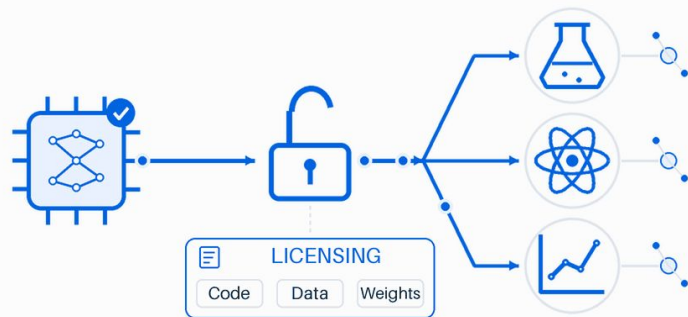
Elisa Cascardi
Research Program Manager, FAIR

Open source is our design decision from day 1

Model success, trapped by infra

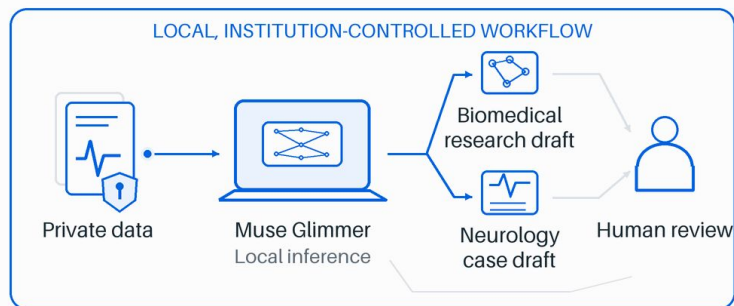


Open infra enables continued discovery



Open releases enable building at scale

MUSE GLIMMER



Locally-run Glimmer enables biomedical research and neurological case management with HIPAA requirements

SEGMENT ANYTHING 3 / SAM3

93M

SAM Family model downloads

1.90M

Last 30 days

[facebook/sam3](https://facebook.com/sam3) · Hugging Face
Snapshot: October 19, 2026

Each release becomes a starting point for the next breakthrough

How can we enable community breakthroughs?

01 BUILD WITH OPEN ARTIFACTS

MUSE GLIMMER

PyTorch, ExecuTorch,
TorchTitan

SAM3

PyTorch (Torch.compile,
torch.export)

02 RESEARCH, DEV & RISK MITIGATIONS

Model Development
and Testing + Risk
and Safety mitigations

Model Development
and Testing + Risk and
Safety Mitigations

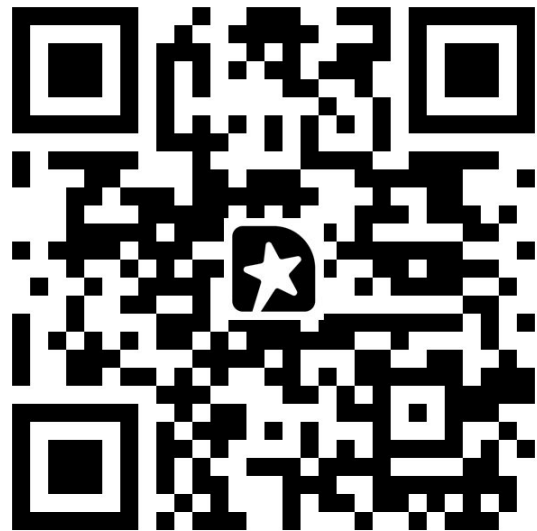
03 DEFINE TERMS & RELEASE

Fine-tuning/inference code, model weights
released
Commercially permissive licensing

Model weights artifacts released
Commercially permissive licensing

Thank you!

Feedback, ideas, suggestions welcome!



How PyTorch Powers Open Foundational Models – From Language to Vision

