Diagnosing a PyTorch Memory Leak on Apple MPS

> *How Stormlog helped us find and fix a subtle GPU memory leak that only appeared on Apple Silicon.*

AUTHOR
Prince Agyei Tuffour

PUBLISHED
May 15, 2026

READING TIME
2 min

IN THIS PIECE

01 The symptom
02 What made it hard to find
03 How Stormlog helped
04 The fix
05 What I took away

When we started testing Stormlog on Apple Silicon, everything worked as expected on CUDA. The memory timeline was clean, allocations were predictable, and the leak detector stayed quiet. Then we switched to the MPS backend.

01

The symptom

During a standard training loop with mixed-precision autocast enabled, GPU memory usage climbed steadily across epochs. Not dramatically; maybe 50MB per epoch on a 16GB M2 Max. Easy to miss if you're not watching closely. But over a long training run, it would eventually exhaust available memory and crash.

02

What made it hard to find

The leak only appeared under a specific combination of conditions: MPS backend, autocast enabled, and tensors moved between MPS and CPU within the same training step. Remove any one of those three conditions and the leak disappeared.

This is the kind of bug that survives manual testing because no single condition is unusual. Mixed-precision training is standard practice. Moving tensors between devices happens during data loading and metric computation. It's only the intersection that triggers the problem.

03

How Stormlog helped

Stormlog's allocation timeline made the pattern visible. Each training step showed a small residual allocation that was never freed: a cached autocast tensor that the MPS allocator held a reference to after the tensor was copied to CPU. The tensor-level breakdown tab showed the exact allocation site.

04

The fix

The fix was straightforward once we understood the cause: explicitly clearing the autocast cache after cross-device tensor operations. But finding the cause without a profiler that tracks individual allocations would have been significantly harder.

05

What I took away

Memory leaks in ML training are rarely dramatic. They're slow, conditional, and only surface under realistic training conditions. Building profiling tools that make these patterns visible is the kind of infrastructure work that compounds over time.

• • •

That's the piece. If it earned a margin note, I'd love to hear it.

prince.agyei.tuffour@gmail.com

Typeset from the live article at princeagyeituffour.com/writing/mps-memory-leak. The web version may be newer and carries the interactive parts.