PyTorch CONFERENCE —NORTH AMERICA 2026—

Perch 2.0 & Perch-Hoplite in Pure PyTorch

An end-to-end, TensorFlow-free bioacoustics pipeline on NVIDIA Grace Blackwell

MBARI
Duane R. Edgington
Senior Software Engineer, Monterey Bay Aquarium Research Institute (MBARI)
PyTorch Conference North America 2026 · Poster

P1 The problem

Perch 2.0 (Google Research) classifies ~15,000 species and produces general-purpose audio embeddings widely used in conservation. It ships as TensorFlow.

On the newest NVIDIA hardware — the GB10 (Grace Blackwell, sm_121, CUDA 13, aarch64) — a working accelerated TensorFlow stack is effectively unavailable, while PyTorch runs natively. The framework dependency became a hard barrier to using an excellent model.

P2 The approach

Rather than wrap or convert the TF graph, I reimplemented Perch 2.0's embedding model from scratch as idiomatic `torch.nn.Module` code — a log-mel frontend and an EfficientNet-B3 embedder — loading weights extracted from the model's own graph.

Readable, hand-written PyTorch you can teach from, modify, and extend — not a mechanically converted graph.

P3 Reverse-engineered to get it exact

FRONTEND Perch 2.0 uses log scaling **0.1·log(max(mel,1e-5))** — not PCEN. (Perch 1.0 and some docs say PCEN — the source of much confusion.)

PADDING The convolutions' edge-padding scheme is undocumented and non-obvious; getting it wrong silently drops embedding cosine to ~0.82. (Specifics in the repo.)

QUIET AUDIO Deep-water hydrophone audio is ~1000× quieter than birdsong; per-window peak-normalization to 0.25 is essential for parity.

P4 Related work — what's different

Prior PyTorch routes convert the ONNX graph (onnx2torch). This is a hand-built, readable nn.Module reimplementation, verified against the reference TensorFlow model — relative L2 ~4×10⁻⁴, not just high cosine — carrying the marine low-amplitude finding and the full TensorFlow-free Blackwell pipeline.

P5 Headline results

~1.0
cosine vs reference TF
relative L2 ~4×10⁻⁴)

2.5x
faster than ONNX
~635 clips/s, on-device

100%
TensorFlow-free
embed → search → classify

P6 Architecture

The native PyTorch model — raw audio → log-mel frontend → EfficientNet-B3 → 1536-d embedding. FFT in float64; CNN in float32.

P7 Numerical parity

Path	Device	cosine	rel. error
Native embedder (ref. frontend)	CPU	0.9999999	~7e-7
Full pipeline (raw audio→emb.)	CPU	1.0000000	~1–5e-5
Full pipeline vs reference TF	**GB10 GPU**	**0.9999998**	**~4e-4**
ONNX bridge (cross-check)	CPU	—	~1e-9

Relative L2 ~4×10⁻⁴ (not just cosine ~1.0) confirms the embeddings match in magnitude, not only direction — a high cosine alone can hide a large error.

P8 Throughput on the GB10 (DGX Spark)

Path	b=1	b=4	b=8	b=16
ONNX bridge (ORT-CUDA)	102	200	207	211
Native PyTorch (eager)	177	245	227	209
Native + torch.compile	**350**	**635**	**607**	**533**

Native keeps the whole graph on-device; the ONNX bridge's in-graph DFT forces host↔device copies. torch.compile is ~2.5× the bridge at batch 4–16.

P9 Why it matters — the payoff

The whole **embed** → **search** → **classify** agile-modeling loop (perch-hoplite) now runs **100% TensorFlow-free** in one PyTorch environment on the box — replacing a Colab/TensorFlow round-trip.

Applied to MBARI's MARS cabled hydrophone (891 m deep, Monterey Bay National Marine Sanctuary): transfer-learning on frozen Perch embeddings to monitor a decade-scale deep-ocean soundscape. A retrain takes ~30 s — a working detector from hours, not months, of expert listening. And the port is fast enough that a pair of NVIDIA DGX Spark systems can embed and scan the entire 11-year MARS archive in about 1.5 months (roughly 3 months of audio processed per day) — decade-archive analysis becomes practical on-device.

P10 This project's linear classifier

A linear classifier on frozen Perch embeddings, over ~1.56M MARS embeddings (2018, 2020, 2026), four classes — Bigg's orca, Pacific white-sided dolphin, humpback, ship noise.

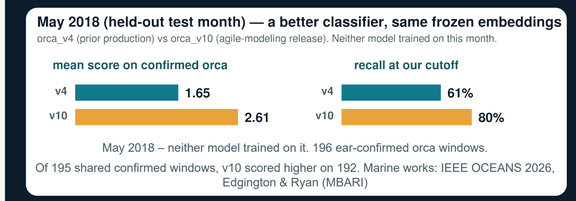
ROC-AUC 0.959. Cleanly separates orca from dolphin — the classic hard case.

P11 Code & details

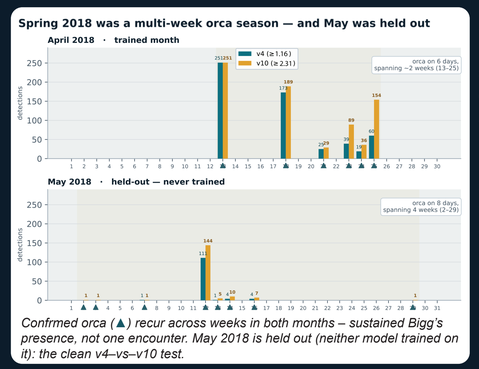
This port
`github.com/duane-edgington/perch2-pytorch-port`
Apache-2.0 · ships no weights

The science
`github.com/duane-edgington/perch-hoplite-orcas-MBNMS`
OCEANS '26 · orca detection

P12 The science it enables — making invisible biology visible

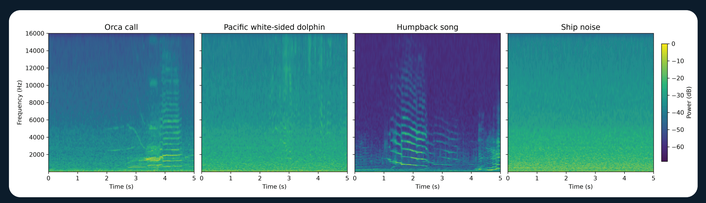


May 2018 is a permanently held-out test month — neither model trained on it. Against 196 ear-confirmed orca windows, the agile-modeling loop lifted the detector from the prior production model (v4) to v10, using the same frozen Perch V2 embeddings and only a better linear classifier.
Mean score on confirmed orca rose 1.65 → 2.61; recall at our operating cutoff rose 61% → 80%. Of 195 shared confirmed windows, v10 scored higher on 192

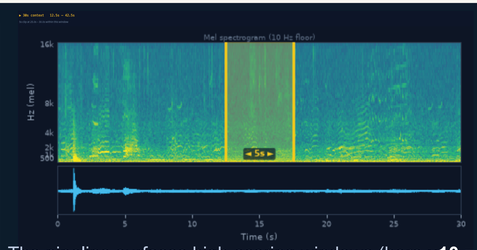


Confirmed orca (▲) recur across weeks in both months – sustained Bigg's presence, not one encounter. May 2018 is held out (neither model trained on it): the clean v4–vs–v10 test.

Spring 2018 was a multi-week Bigg's orca season — confirmed orca (triangles) recur across weeks in both months, sustained presence rather than a single encounter. On the same days, v10 (amber) detects more than v4 (teal). April is the trained month; May 2018 is held out (neither model trained on it) — the clean v4-vs-v10 test.



Orca calls and humpback song phrases share harmonic structure in the same band — the hard case that confuses other models. v10 separates them, even detecting orca calls in the midst of humpback song.



The pipeline surfaces high-scoring windows (here **v10 = 2.895**) in an in-browser labeling tool. This 2015 window opens with a **seal-bomb detonation** (a sea-lion deterrent charge; ~9,000/yr on this hydrophone — Simonis et al. 2020, incl. J. Ryan & Y. Zhang) **followed by orca calls**. A striking association pulled from the archive. (Association, not causation.)

A port, not a new model. Perch 2.0 architecture & weights © Google Research (Apache-2.0); ONNX export by justinchuby; EfficientNet by Tan & Le. This work redistributes no Perch weights.