Fish Classification with Wideband Hydroacoustics and Deep Learning

Osama Tarek Alshabani
Vianey Leos Barajas
osama.alshabani@kaust.edu.sa
Date: August 8, 2026
Code: https://github.com/MarcoGallegos34/fish_classification-summer2026

Contents

Introduction 2
Methodology 3
Method details 3
Evaluation Procedure 4
Results 4
Discussion 5
Limitations and Future Work 6
Conclusion 7

Introduction

- Ontario's freshwater lakes support ecologically and commercially important fish populations, but once established, invasive Alewife (Alosa pseudoharengus) can outcompete native species such as Rainbow Smelt (Osmerus mordax). Reliable, non-invasive tools for tracking their relative abundance are essential for fisheries managers, and hydroacoustic (echosounder) surveys offer a scalable alternative to net-based sampling. An upward-looking, split-beam sonar deployed at two shore locations records two acoustic channels in parallel, a 70 kHz transducer and a 200 kHz transducer. Every fish that swims through the beam leaves behind an acoustic track, and each track consists of a sequence of individual pings, with each ping carrying a backscatter strength (in dB) measured across dozens of narrow frequency bins.
- The task addressed in this report is to classify each fish track as Alewife or Rainbow Smelt from its acoustic backscatter response, recorded across 91 frequency bins at 70 kHz and 201 frequency bins at 200 kHz, with pings grouped into trajectories by a track_id. Because the fish detected by the sonar are never physically caught or visually identified, species identity cannot be observed directly; it is instead inferred indirectly from the water temperature at the depth and time of each ping, interpolated from co-located CTD (temperature–depth) casts, under the assumption that Alewife occupies warmer, shallower water (temperature above 16°C) while Rainbow Smelt occupies colder, deeper water (temperature below 12°C).
- This labeling approach introduces three problems that shape the modeling pipeline described in this report. The first is **class imbalance**: Rainbow Smelt accounts for only 5–7% of labeled pings, with Alewife making up the remaining 75–85% at both frequencies. A model trained naively on this distribution defaults toward predicting the majority class, since doing so already yields high accuracy while almost never correctly identifying the minority species.
- The second is **uncertainty in intermediate labels**: pings whose interpolated temperature falls between 12°C and 16°C cannot be confidently assigned to either species and are left unlabeled, comprising 15–20% of all pings. This transitional band is not simply a labeling gap — pings within it span depths from under a meter to nearly 27 meters, overlapping substantially with the depth ranges characteristic of both species, so no fixed temperature or depth cutoff can resolve these tracks with confidence.
- The third is **potential data leakage** introduced by the splitting strategy itself. Models are evaluated on data split by track_id, which keeps every ping from a single beam pass within one split. However, a single physical fish can swim in and out of the sonar beam repeatedly, and each pass is assigned a new track_id, as though it were a different fish. If two passes made by the same fish land in different splits, a model can partially learn to recognize that individual fish's own acoustic signature rather than its species identity, inflating performance on data that is not genuinely held out. The primary dataset carries no persistent fish-identity field, so this risk cannot be verified directly on the data used for the classification task itself.
- This report presents the methodology developed to address each of these three problems and a comparative evaluation of three deep learning architectures — a one-dimensional residual convolutional network (CNN1D), a two-dimensional convolutional network built on a pretrained EfficientNet backbone (CNN2D), and a Vision Transformer (ViT-B/16) — trained and evaluated on a shared, track-grouped data split at both 70 kHz and 200 kHz.

Methodology

Method details

- **Data representation.** For every track, pings are ordered in time and, for the two-dimensional and attention-based models, stacked into a matrix whose rows are pings and columns are frequency bins. Backscatter values are clipped to the 1st/99th percentile computed from the training split only, rescaled to [0, 1], and passed through a colormap to produce an RGB spectrogram-like image. Variable-length tracks are converted to fixed-size images with a sliding-window scheme (window size of 9 pings, non-overlapping stride) rather than by resizing, which avoids the temporal distortion a simple resize would introduce. The frequency axis is kept at native resolution (91 px at 70 kHz, 201 px at 200 kHz) and is never resized.
- **Addressing class imbalance.** Two complementary techniques are used together. Inverse-frequency class weighting assigns each class a weight inversely proportional to its frequency and passes these weights into the loss function, so that a mistake on a rare Rainbow Smelt example counts proportionally more than a mistake on a common Alewife example; weights are recomputed after every self-training round as the class balance shifts. Self-training on the unlabeled, intermediate-temperature pool supplies additional minority-class examples: the current best model scores every unlabeled track, and only tracks predicted as Rainbow Smelt with confidence at or above 0.95 are converted into pseudo-labels and folded into training, while confidently predicted Alewife tracks from the same pool are discarded, since Alewife is already the majority class. This process runs for up to three rounds of fine-tuning.
- **Addressing uncertain intermediate labels.** The same self-training mechanism used to address class imbalance also resolves the intermediate-depth, intermediate-temperature ambiguity described in the Introduction: rather than applying a brittle, fixed temperature or depth rule, the classifier's own confidence is used to fold ambiguous tracks into the labeled set gradually and conservatively, only when its prediction exceeds the 0.95 confidence threshold.
- **Addressing potential data leakage.** Train, validation, and test splits are constructed by grouping on track_id using scikit-learn's StratifiedGroupKFold, so every image or ping derived from a given track lands in exactly one split, while stratification keeps the Alewife-to-Rainbow-Smelt ratio approximately constant across splits (target ratio 70/15/15, yielding approximately 66.7%/16.7%/16.7% in practice due to fold-count rounding). Because this grouping strategy cannot rule out leakage from a single fish generating multiple track_ids, it was stress-tested independently on a second, experimental dataset of tagged, individually identified fish, where true fish identity is recorded directly. The identical spectrogram-and-EfficientNet-B0 pipeline was run twice on this dataset, differing only in whether the split was grouped by the fish's true tagged identity or by track_id, to directly measure how much leakage the track-based grouping strategy permits.
- **CNN1D — residual one-dimensional convolutional network.** The simplest of the three architectures, CNN1D reads each ping's raw 91- or 201-bin frequency vector directly as a one-dimensional input, with no spectrogram or image step, testing whether a single ping's frequency response alone carries sufficient signal to separate species. A small residual network ("FishResNet1D") applies a convolutional stem followed by a stack of residual blocks — paired convolution, batch normalization, and ReLU layers with a skip connection, projected through a 1×1 convolution whenever the channel count changes — followed by global average pooling and a linear classification head. Hyperparameters (block channel widths, kernel size, learning rate, dropout, and

weight decay) were tuned per frequency with a 20-trial Optuna search using track-grouped, 3-fold inner cross-validation on the training split.

- **CNN2D — EfficientNet-B0 on spectrogram images.** CNN2D fine-tunes EfficientNet-B0, pretrained on ImageNet, on the window=9 spectrogram images described above, treating the ping sequence as an image so that the convolutional filters can exploit both the frequency structure within a ping and the temporal structure across consecutive pings jointly. EfficientNet-B0 was selected over larger EfficientNet variants for its comparatively small parameter count, leaning on transfer learning rather than a hand-designed architecture; its classifier head is replaced for the two-class task, and the backbone is fully unfrozen from the first training epoch.
- **ViT-B/16 — Vision Transformer.** ViT-B/16 replaces convolution with self-attention over image patches, applied to the same spectrogram images as CNN2D. It is a substantially larger, pretrained architecture (85.8 million trainable parameters, versus 4.0 million for EfficientNet-B0), included to test whether a state-of-the-art, attention-based vision backbone can outperform the smaller, purpose-built convolutional models. As with CNN2D, the entire backbone is unfrozen from the first epoch and fine-tuned at a conservative learning rate.

Evaluation Procedure

- All three architectures are evaluated on one shared, fixed, track-grouped, species-stratified split, so that results are directly comparable across model families rather than each pipeline computing its own independent split. Performance is reported by track, using F1-score (macro), balanced accuracy, and the sensitivity and specificity of the minority class (Rainbow Smelt), at the best validation checkpoint per run.
- Self-training is applied iteratively with a confidence threshold of 0.95; most configurations retain only minority-class pseudo-labels, while one configuration additionally retains confident majority-class pseudo-labels for comparison. Class weights are recomputed after every round to reflect the updated training distribution.
- Leakage is checked directly rather than assumed. Overlap between the unlabeled self-training pool and the train, validation, and test splits is audited both by track_id and by a ping-level fingerprint match, and the leakage stress test described above reports the fraction of fish, and of test images, shared across splits under each grouping strategy.

Results

- Grouping the tagged-fish dataset's split by track_id rather than by true fish identity let 62 of 64 fish (97%) appear in more than one split, and 98.8% of the held-out test images came from a fish already present in training (Table 1). The aggregate single-run F1-score moved in the opposite direction from what leakage would naively be expected to produce (0.708 under leakage versus 0.797 without it); repeating each condition over five random seeds clarified the picture, however: the leaky split's mean F1-score was both higher (0.721 versus 0.687) and approximately ten times less variable (± 0.009 versus ± 0.083), because the composition of the leaky test set barely changes across seeds when nearly every fish is already present in every split.

Table 1. Leakage stress test on the tagged-fish dataset: identical pipeline, split by true fish identity versus by track_id.

Metric	No Leakage (split by fish)	With Leakage (split by track)
Fish shared across splits	0 / 64	62 / 64 (97%)
Test images from an already-seen fish	0%	98.8%
Single-run F1 (macro)	0.797	0.708
5-seed mean F1 (macro)	0.687 ± 0.083	0.721 ± 0.009
Seen- vs. unseen-fish accuracy (pooled)	n/a	73.9% vs. 58.3%

- A companion three-species run and an independent overlap audit corroborate this finding from two additional angles. In the three-species run, per-class recall shifted substantially under leakage, in opposite directions for different classes (one class's recall rose from 32% to 57%; another's fell from 95% to 58%), even though the aggregate F1-score barely moved (0.630 without leakage versus 0.620 with it). Separately, an overlap audit of the primary pipeline's self-training pool found zero shared pings or tracks against the train, validation, and test splits at either frequency, confirming that step is not itself a source of leakage.
- Table 2 reports the best result achieved by each of the three models on the shared, track-grouped split. ViT-B/16, at 200 kHz with self-training, achieved the strongest overall result: F1-score (macro) of 0.929, balanced accuracy of 0.958, and Rainbow Smelt sensitivity and specificity of 0.929 and 0.987 respectively. CNN2D (EfficientNet-B0) reached its best result at 200 kHz with self-training (F1-score of 0.868), and CNN1D reached its best result at 70 kHz with self-training (F1-score of 0.867). Self-training contributed to the best-performing configuration for all three models, though its effect varied by architecture and frequency: it did not improve CNN1D at 200 kHz or ViT-B/16 at 70 kHz, where the supervised model was retained instead.

Table 2. Best result achieved by each model on the shared train/validation/test split, by track.

Model	Trainable Params	F1 (macro)	Best Configuration / Bal. Acc. / Sens. / Spec. (class 1)
CNN1D (ResNet1D)	964,130	0.867	70 kHz, self-trained — 0.867 / 0.750 / 0.983
CNN2D (EfficientNet-B0)	4,010,110	0.868	200 kHz, self-trained — 0.948 / 0.929 / 0.968
ViT-B/16	**85,800,194**	**0.929**	**200 kHz, self-trained — 0.958 / 0.929 / 0.987**

Discussion

- The leakage results demonstrate that data leakage in this setting is a real but subtle risk rather than a straightforward inflator of the headline metric. Aggregate F1-score barely moved under leakage, and in the single-run comparison it was even slightly lower — a result that, considered alone, might be mistaken for evidence that leakage does not matter here. The more informative evidence lies elsewhere: leakage produced a large, mechanistically consistent shift in per-class performance,

disproportionately helping classes the model otherwise struggles to generalize on, since memorizing an individual fish's acoustic signature pays off most where genuine species signal is weakest; and it made the leaky run's validation curves appear artificially stable, with training and validation loss tracking closely rather than diverging, purely because most validation fish were already present in training. A well-behaved validation curve under these conditions does not indicate better generalization — it indicates that the diagnostic itself has been compromised, which is arguably a more consequential harm than an inflated test score, since it also degrades the signal used for early stopping and model selection.

- Because the primary Alewife/Rainbow Smelt dataset carries no persistent fish-identity field, the leakage question could only be tested by proxy on the tagged-fish dataset, and remains open for the results reported directly on the primary task. The general implication is that a grouped split should not be trusted merely because a grouping argument is present in the code; the grouping key must be verified against the unit that genuinely needs to remain independent — in this case, the individual fish — through direct overlap audits.
- ViT-B/16 achieving the strongest overall result, but only at 200 kHz and only with self-training, indicates that self-training's benefit is architecture- and frequency-specific rather than universal, and should continue to be evaluated per configuration rather than applied as a fixed processing step.
- ViT-B/16 also required roughly twenty times as many trainable parameters as EfficientNet-B0 and nearly ninety times as many as CNN1D to achieve its margin of improvement, so the choice among the three architectures involves a genuine trade-off between predictive performance and computational cost, and the smaller CNN2D and CNN1D models remain competitive options where that cost matters.

Limitations and Future Work

- Whether track-grouped splitting is adequate for the primary field-collected dataset remains an open question, because that dataset provides no fish-identity field against which to check it directly. Should such a field become available — for example, through physical tagging or acoustic-signature matching across track_ids — every performance figure reported for the primary classification task should be re-audited against it before being used operationally.
- The results reported in Table 2 rely on a single run per configuration rather than a nested or repeated cross-validation protocol, so run-to-run variance due to which fish happen to fall in the test split is not yet quantified. Repeating each configuration across multiple track-grouped splits, as was done for the leakage stress test, would give a more robust estimate of each model's performance and of the margin between them.
- Self-training's inconsistent effect across model and frequency combinations is not yet well understood mechanistically. Distinguishing pseudo-label noise from frequency-specific signal quality or model-capacity effects, rather than treating self-training as a step to apply by default and keep only when it happens to help, remains unresolved.

Conclusion

- This report addressed three problems inherent to labeling fish species from hydroacoustic backscatter and CTD-derived temperature bands — class imbalance, uncertainty in intermediate labels, and the potential for data leakage introduced by track-based splitting — and evaluated three deep learning architectures under a methodology designed to mitigate each. Class imbalance and label uncertainty were addressed jointly through inverse-frequency class weighting and confidence-thresholded self-training on the unlabeled, intermediate-temperature pool. The risk of data leakage from track-based grouping was tested directly, using a dataset with verified fish identity, and found to be a genuine but subtle risk: it did not substantially inflate the aggregate F1-score, but it distorted per-class performance and compromised the validation diagnostics used for model selection.
- Comparing the CNN1D, CNN2D, and ViT-B/16 architectures on a single, shared, track-grouped split identified ViT-B/16 at 200 kHz, with self-training, as the strongest classifier evaluated, with an F1-score (macro) of 0.929 and a balanced accuracy of 0.958, ahead of CNN2D (0.868) and CNN1D (0.867). The central recommendation for continued work is to resolve whether track-based grouping is adequate for the primary dataset before placing further weight on architecture-level gains.