A SHARED RECIPE FOR MULTICHANNEL BIOACOUSTIC ALIGNMENT

Technical Report

Colm Ennis

London, England

ABSTRACT

This paper describes a single shared recipe for multichannel bioacoustic alignment that reduces MSE by 97–98% over the strongest baselines across two distinct acoustic regimes: pairs of autonomous recording units (ARU) and an ARU paired with a bird-mounted logger (Zebra Finch, ZF). The recipe pairs a frozen pretrained audio encoder (BEATs) with a small MLP head that predicts whether two one-second windows are aligned; a constrained affine search then estimates a global offset and slope for each file. The architecture, optimiser, and loss are identical across both regimes. Between datasets, the only deliberate design choice is the BEATs layer; search ranges and negative-sampling offsets follow directly from the known clock-drift bound. Under a five-fold median-score ensemble, the system reaches 0.0004 s^2 MSE on ARU and 0.039 s^2 on ZF (95% CI $[0.022, 0.059]$), reductions of 98% and 97% relative to the strongest challenge baseline on each sub-dataset.

***Index Terms*—** Multi-channel alignment, BEATs, wildlife monitoring

1. INTRODUCTION

Acoustic monitoring is a key tool in wildlife conservation, giving a non-invasive way to study animal behaviour in both wild and captive settings. Because animals move, studying them often means deploying several recorders across a scene: comparing the same event across microphones at known positions reveals where a source is and how it moves, information a single channel cannot give. Combining these channels requires synchronisation, which is difficult because each device runs on its own clock and the offset between their recordings drifts non-linearly, amounting to seconds over the course of a recording. This drift must be corrected before the channels can be analysed jointly.

BioDCASE 2026 Task 1 asks for methods that perform this alignment automatically. Each file is annotated at one-second keypoints, each a pair (k_0, k_1) of timestamps marking the same physical instant on Channel 0 and Channel 1; systems are given the Channel-0 times k_0 and must predict the Channel-1 times k_1, scored by MSE (s^2) and MAE (ms).

The task spans two regimes: the similar-channel **ARU** and the dissimilar-channel **Zebra Finch** (ZF). ZF is the harder regime and the one that motivates our design; both are analysed in Section 2. The paper shows that a single shared recipe suffices across both regimes, identifies which components drive reliability on ZF through a cumulative ablation, and proposes a statistical framework suited to the small validation set size. Table 1 reports our results against the challenge baselines.

Table 1: Headline results on the official validation set; lower is better. †organisers' published values [1]. Bootstrap 95% CIs: ARU MSE $[0.0003, 0.0005]$, ZF MSE $[0.022, 0.059]$.

	ARU		Zebra Finch	
Model	MAE (ms)	MSE (s^2)	MAE (ms)	MSE (s^2)
`nosync`†	116.2	0.019	2297.4	5.734
`gccphat`† [2]	122.4	0.020	2457.0	7.995
`deeplearning`†	164.3	0.040	757.1	1.218
Ours	**14.7**	**0.0004**	**121.8**	**0.039**

2. ANALYSIS OF CHALLENGE DATA

2.1. ARU

The ARU sub-dataset pairs two autonomous recording units placed a fixed distance apart in the same forest scene [3]. Both units sit in the same soundscape, so their channels share most acoustic content; differences arise from sources whose proximity varies between the two recorders. The units' positions are fixed and the offset varies smoothly across the recording.

2.2. Zebra Finch

The ZF sub-dataset pairs an off-body ARU recorder with an on-body sensor carried by the bird [3]. The channels are dissimilar due to microphone placement rather than separation: the on-body sensor is dominated by the calls and movement of the single bird wearing it, while the off-body recorder captures the wider room – other birds and ambient sound that are heavily attenuated at the on-body microphone. Because many acoustic events appear on only one channel, the same physical instant often sounds highly dissimilar across the two recordings, and a similarity-based alignment method may score the true alignment poorly. The bio-logger's clock-drift and start-offset bounds are far wider than the ARU's (Table 2), requiring a correspondingly broader alignment search. We additionally find the stereo channel ordering to be inconsistent across training files: the bio-logger channel switches in about 47% of files.

2.3. Findings from the 2025 challenge

Results from the 2025 edition of this task [3] confirm that ZF is the harder regime. Task 1 drew five submissions from three teams. On Zebra Finch, only one submission (BEATsCA, MSE 0.45 s^2) outperforms the no-synchronisation baseline (1.84 s^2), with the deep-learning baseline (0.55 s^2) the only other system to best it; most systems were competitive on ARU. Zebra Finch thus dominated total error and proved challenging for most prior baseline methods. The

Table 2: Drift and offset statistics over the training set.

	ARU	Zebra Finch
Max clock-rate drift (excess slope)	2.5×10^{-3}	3.4×10^{-2}
Max start-time offset (s)	0.5	5.0
Best-fit affine floor, MAE (ms)	11	70
Best-fit affine floor, MSE (s^2)	0.0002	0.011
Official validation files	12	16

2026 recordings are distinct, but the data conditions are the same. We take ZF reliability as the primary design criterion throughout, removing components with inconsistent performance across folds and hyperparameter selection.

3. METHOD

We model the per-file relationship between the two channels' clocks as a single affine map: a slope for the clock-rate difference and an offset for the start-time difference. The best-fit affine floor lies roughly a hundred times below the no-synchronisation baseline on ARU and five hundred times below on ZF (Table 2), so an affine model captures most of the available improvement.

The pipeline has three stages:

- **Feature Extraction:** A frozen BEATs encoder extracts embeddings, pooled to 10 bins per second.
- **Pairwise Classification:** An MLP head with a single 32-unit hidden layer predicts alignment probability using a combined Binary Cross-Entropy (BCE) and within-batch contrastive loss.
- **Global Alignment:** A grid search over constrained affine transformations estimates a global offset and slope, stabilised by a five-fold cross-validation ensemble.

3.1. Architecture

The backbone is a frozen BEATs encoder [4]. We take intermediate layer embeddings and pool them to 10 bins per second. Maintaining temporal resolution is necessary to preserve the rapid, localised transients characteristic of zebra finch vocalisations. Because the pretrained encoder remains frozen, global pooling across a full one-second window would remove the sub-second acoustic structure required to distinguish fine-grained temporal offsets.

The head is an MLP with a single 32-unit hidden layer and a ReLU activation. Its input is the symmetric embedding $[e_0, e_1, |e_0 - e_1|]$ formed from flattened encoder features, and it outputs a single aligned/misaligned logit. The network is trained using a multi-task objective that combines a standard Binary Cross-Entropy (BCE) loss on the sampled training pairs with a dense, balanced pairwise BCE auxiliary loss ($\lambda = 0.5$) computed over an all-pairs batch matrix (detailed in Section 3.3).

3.2. Alignment search

At inference we search a single affine $(\text{offset}, \text{slope})$ map shared across the whole file, over a 50×50 grid of offset–slope pairs. The offset and slope ranges follow the drift and offset statistics of the dataset (Table 2), with the slope bounded at $1.5\times$ the largest training-fold drift. Each candidate map is scored by the sum of the head's logits over all keypoints, and we keep the highest-scoring map, with no silence gating applied.

Constraining the alignment to a single global affine map uses the full file context; summing logits across all keypoints mitigates localised classification errors. In contrast, local decoding strategies, such as the baseline's per-keypoint Viterbi search, resolve intervals independently and remain susceptible to local acoustic ambiguities.

A naive grid search embeds one Channel-1 window per keypoint per candidate, requiring $2500 \times N_{\text{kp}}$ forward passes per file. Instead embeddings are precomputed once on a dense 0.05 s temporal grid, and alignment candidates retrieve the required embedding via nearest-neighbour lookup. This leads to a 125x reduction in the number of BEATs encoder forward passes.

3.3. Training setup and augmentations

Random sub-second keypoint sampling. The ground-truth keypoints are spaced one second apart. Rather than training only on these fixed anchors, each example draws a random sub-second offset within an interval and linearly interpolates the corresponding Channel-1 timestamp. This has a regularising effect, increasing the number of useful training steps before overfitting.

Channel-flip augmentation ($p = 0.5$). To handle the inconsistent ZF channel ordering, we swap (x_0, x_1) with probability 0.5 per example, training the head to be channel-order invariant. We prefer this to a channel detector: a single mis-routed file would incur seconds of error and raise ensemble MSE sharply on ZF, whereas augmentation is immune to detector failure.

Endpoint trim. We drop each file's first and last keypoints. Ground-truth timestamps are constrained to lie within the audio file boundaries, so labels near the file edges may be clipped to the boundary rather than reflecting the true affine relationship. Dropping both boundary keypoints increases training stability across configurations.

Negative sampling. Misaligned (negative) training pairs are drawn with a minimum absolute offset of $0.3 \times$ max_error (0.15 s on ARU, 1.5 s on ZF), chosen to give clear temporal separation between positive and negative examples, and up to the full max_error, so the head never sees a "negative" that is in practice a near-aligned positive.

Pairwise BCE auxiliary loss. To prevent convergence collapse when cross-channel similarity is low (ZF regime), we add an auxiliary loss over all B^2 pairings of the B positive Channel-0 windows against the B positive Channel-1 windows in the batch. Each pair is scored by the same MLP head; the B diagonal pairs carry a positive label (aligned) and the $B(B-1)$ off-diagonal pairs carry a negative label (misaligned). Standard BCE with a positive class weight of $B-1$ is applied across the full matrix, and the auxiliary term is mixed with the main BCE loss at λ=0.5. Unlike InfoNCE, there is no softmax denominator: each cell is an independent binary prediction.

Shared hyperparameters. Both datasets share one schedule: lr $= 1 \times 10^{-4}$, weight decay 1×10^{-4}, batch size 128, 250 epochs, and an MLP head with a 32-unit hidden layer.

Stability. We observe that some seeds collapse at initialisation, with BCE fluctuating around $\ln 2$ chance value. We detect this at epoch 10 and restart training with a new seed, up to three times.

3.4. Evaluation and statistical framework

We apply five-fold cross-validation to the training set, keeping the official validation set as an unbiased held-out measure. A single training split would give a noisy estimate of accuracy given the

small training set size; cross-validation instead scores every training file once in its held-out fold, and we pool these predictions into one metric over all 36 (ARU) and 108 (ZF) training files. This pooled metric is the basis for our design decisions, and the five fold models are combined into the final ensemble (Section 3.5).

Statistical conventions. Inference is at the fold level (n=5): interval estimates use a two-stage cluster bootstrap (resampling folds, then files) and paired comparisons use a fold-level sign test; differences within fold noise are reported as inconclusive. We distinguish two views throughout: the *CV-holdout* view, in which each of the 108 ZF / 36 ARU training files is scored in its held-out fold, and the *official* view, in which the deployed five-fold ensemble is evaluated on the 16 ZF / 12 ARU validation files.

3.5. Submission aggregation

The five fold models share the same affine candidate grid, so we can ensemble before committing to a single alignment. For each file we take the median across the five folds of every candidate's score and select the single affine candidate with the highest median score. Ensembling the scores rather than the predicted keypoints keeps the submission outputs affine, while the median over five folds tolerates up to two folds converging on a spurious peak. Because the challenge metric is the mean per-file MSE, a single high-error file strongly affects the score; taking the median vote across folds is designed to suppress these per-fold outliers.

3.6. Comparison to challenge baseline

The challenge baseline uses separate architectures and inference algorithms for each dataset; our approach uses a single recipe for both. Table 3 summarises the key differences.

Table 3: Component comparison: challenge baseline versus our recipe. Per-dataset differences are the BEATs layer (Section 4.1) and the search and sampling ranges (Table 2).

Component	Baseline (ARU)	Baseline (ZF)	Ours
Temporal pooling	8 bins/s	1 bin/s	10 bins/s
Contrastive loss (λ)	0	0.5	0.5
Learning rate	3×10^{-4}	3×10^{-4}	1×10^{-4}
Alignment	Stride-8 + Viterbi	5×5 affine	50×50 affine
Neg. sampling	Curriculum (8 ep.)	Curriculum (8 ep.)	0.3–1.0× max offset
Silence gating	√	√	—

Our 50×50 affine grid is ten times finer than the baseline's 5×5 on ZF; the additional candidates are cheap because precomputed embeddings reduce each evaluation to a nearest-neighbour lookup rather than a forward pass (Section 3.2). Silence gating provided no benefit alongside the stronger recipe.

4. RESULTS

Table 1 compares our submitted ensemble against the three challenge baselines on the official validation set. On ARU both active baselines perform worse than no-synchronisation (GCC-PHAT MSE 0.020, deep-learning 0.040 versus nosync 0.019), making nosync the strongest available reference on that sub-dataset. The results demonstrate that a unified architecture and training recipe can handle radically different acoustic regimes.

We report headline MSE for comparability with the baselines, but base our development decisions on two diagnostics that are more stable under the error distribution we now describe. Per-file alignment errors are *bimodal*: a file either locks on (the top-scoring grid cell lands within tens of ms of the affine floor) or is high-error (the search selects a spurious peak seconds from the truth). Mean MSE is then dominated by a few outliers, masking gains across the rest of the set. Ablation decisions therefore rest on:

- **Conditional quality:** median MAE over successfully aligned files.
- **Reliability:** the magnitude-free *high-error rate* (MAE $>\tau$) and *collapse rate* (folds whose training never leaves chance).

We set $\tau = 500$ ms on ZF and 200 ms on ARU, chosen by inspection of the pooled per-file error distribution to fall in the wide empty gap between the two modes; any value in that gap yields the same file partition, so conclusions are insensitive to the exact choice. Statistical conventions – the CV-holdout versus official views, and the cluster bootstrap and fold-level sign test used throughout – are set out in Section 3.4.

Table 4: BEATs layer sweep (minimal recipe, 150 epochs). Cond. Med.: median MAE over locked-on CV-holdout files [cluster CI]. HE%: CV-holdout high-error rate (τ=500 ms ZF / 200 ms ARU). Bold: chosen layer per dataset.

	ARU		Zebra Finch	
Layer	Cond. Med. (ms)	HE%	Cond. Med. (ms)	HE%
1	17 [14, 21]	0	87 [45, 124]	25
4	**17** [14, 20]	**0**	72 [50, 114]	14
7	19 [16, 22]	0	**76** [56, 107]	**3**
12	38 [25, 62]	8	172 [135, 197]	22

4.1. BEATs layer sweep

We sweep across embedding layers $\{1, 4, 7, 12\}$ using a minimal recipe (Table 4). The results reflect the progressive generalisation of BEATs features [4]: lower layers preserve the fine temporal detail required for alignment, whereas deeper layers represent recording-agnostic, high-level concepts. On **ZF**, layer 7 minimises the holdout high-error rate relative to layer 4; layer 12 drops in reliability and its conditional quality degrades to 172 ms, over twice the 70 ms affine floor, through loss of fine timing information. For **ARU**, because layers 1, 4, and 7 are indistinguishable at the 11 ms floor, we default to the midpoint layer 4. The chosen layer is used in subsequent ablations.

4.2. Zebra Finch cumulative ablation

Table 5 tracks reliability and quality across each ablation stage; all per-fold counts below refer to that table. Throughout, none of these stages shifts alignment *quality*: the conditional median MAE stays near the 70 ms affine floor at every step. What changes is reliability.

Training with BCE loss alone is highly unstable: low embedding similarity between channels produces noisy gradients at initialisation, triggering the early-stopping criterion in all five folds, with two folds failing to converge despite three re-initialisation attempts. Adding the contrastive loss eliminates this collapse entirely ($0/5$ folds) and halves the mean high-error count ($7.2 \rightarrow 2.8$) – yet, counter-intuitively, raised mean MSE on the CV set. Whilst the majority of files produce lower error, the system chooses more spurious

Table 5: ZF reliability and quality across ablation stages (τ=500 ms). Cond. Med.: median MAE over locked-on CV-holdout files (n=108). HE/fold: mean high-error count on official validation (n=16). HE-ens: five-fold median-score ensemble on official validation.

Component added	Folds collapsed	Cond. Med. (ms)	HE/fold	HE-ens
Head (BCE, λ=0)	5/5 (2 non-conv.)	78	7.2	3/16
+ Contrastive (λ=0.5)	**0/5**	76	2.8	3/16
+ Random sub-second	0/5	78	0.4	1/16
+ Channel-flip (p=0.5)	0/5	**78**	**0.4**	**0/16**

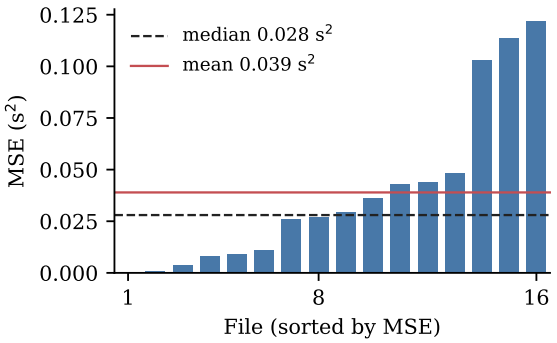

Figure 1: Per-file MSE on the Zebra Finch validation set (16 files), sorted, for the five-fold median-score affine ensemble. All 16 files remain below the high-error threshold (τ=500 ms) that individual folds fail to clear; a few tail files still raise the mean above the median, but no file reaches the outlier levels present in single-fold predictions.

peaks on a small subset of hard files. The five-fold median-score ensemble resolves this failure mode on the validation set: selecting the affine candidate with the highest *median* score across folds suppresses the confident but spurious single-fold peaks, recovering the reliability gain without the MSE penalty.

Random sub-second sampling is the dominant reliability gain, cutting the per-fold high-error count from 2.8 to 0.4. Channel-flip augmentation is retained to guard against the 47% channel-switching rate in ZF training files (Section 2.2), not because the validation set distinguishes it: HE/fold is unchanged at 0.4, and the one-file difference in HE-ens ($1 \rightarrow 0$ of 16) is within sampling noise for a 16-file held-out set. Figure 1 shows the final ensemble.

4.3. ARU regime

ARU performance is very consistent across training configurations. Two of five folds collapsed during training under BCE, but both train cleanly after re-initialisation. Every configuration converges on all 36/36 CV-holdout files with zero high-error files, and the conditional median MAE stays near the 11 ms affine floor (Table 2). The components that drive reliability on ZF leave this untouched: the contrastive auxiliary does not shift the conditional median. We retain the shared recipe for simplicity. The ZF-specific augmentations are benign here: because both ARU channels use the same recording method, channel-flip introduces no semantic contradiction for the MLP, and sub-second sampling provides no additional signal when the encoder's embeddings are already highly discriminative.

5. DISCUSSION AND CONCLUSION

The ARU regime consistently settles near its best-fit affine floor, reaching an official validation median MAE of 14 ms against an oracle target of roughly 11 ms. The pipeline components added to stabilise ZF impose no penalty on ARU: the channel-specific additions (random sub-second sampling and channel-flip augmentation) have negligible impact, while the contrastive auxiliary costs ARU almost nothing in quality (Section 4.3).

Remaining ZF error is concentrated in isolated tail files rather than spread broadly across the set. The deployed ensemble's median per-file MSE sits close to the affine floor, while the mean is skewed upwards by a few challenging recordings (Figure 1). The five-fold median vote suppresses these localised errors, achieving 0/16 high-error files on the validation set (Table 5).

The recipe achieves a final validation MSE of 0.039 s^2 on Zebra Finch and 0.0004 s^2 on ARU – reductions of 97% and 98% relative to the strongest challenge baseline on each sub-dataset. By eliminating redundant components and deriving parameters directly from known clock-drift bounds, the pipeline provides a robust, out-of-the-box framework that minimizes the need for extensive hyperparameter tuning.

Residual ZF error remains bounded by the affine modelling floor; a piecewise affine map or coarse dynamic time warping could reduce it further, though the low oracle floor (Table 2) suggests the gain would be modest.

6. REFERENCES

[1] A. Bhattacharjee, "BioDCASE 2026 task 1 baseline system," https://github.com/chymaera96/biodcase_2026_task1_public, 2026.

[2] C. H. Knapp and G. C. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 24, no. 4, pp. 320–327, 1976.

[3] D. Stowell, E. Vidaña-Vila, I. Nolasco, B. McEwen, *et al.*, "BioDCASE: Using data challenges to make community advances in computational bioacoustics," *bioRxiv*, 2026.

[4] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen, and F. Wei, "BEATs: Audio pre-training with acoustic tokenizers," in *Proc. Int. Conf. Machine Learning (ICML)*, 2023.