A CONFIDENCE-ROUTED HYBRID SYSTEM FOR MULTICHANNEL BIOACOUSTIC ALIGNMENT

Technical Report

Xin Guo[1]*, Xusen Yu*[1]*, Chunrui Zhao*[1]*, Shanzheng Guan*[2]*, Gongping Huang*[1]

[1]School of Electronic Information, Wuhan University, Wuhan, China
[2]Yichang Testing Technique R&D Institute, Yichang, China
xinguo19@whu.edu.cn, xusenyu@whu.edu.cn,chunruizhao@whu.edu.cn
gshanzheng@foxmail.com, gongpinghuang@whu.edu.cn

ABSTRACT

This report describes our submitted systems for BioDCASE 2026 Task 1: Multichannel Alignment. The task requires estimating the Channel 1 timestamp corresponding to each provided Channel 0 keypoint in temporally desynchronized stereo bioacoustic recordings. We formulate the task as time-delay estimation and submit three systems based on GCC-PHAT, neural delay regression, and confidence-based hybrid routing. Two submitted systems use the same confidence-routed hybrid framework, with one system applying an additional trajectory-level post-processing stage and the other using the raw selected delay trajectory directly. All systems predict Channel 1 timestamps by adding the estimated delay to the given Channel 0 keypoint.

Index Terms— BioDCASE 2026, multichannel alignment, time delay estimation, GCC-PHAT, neural regression, post-processing

1. INTRODUCTION

Multi-channel bioacoustic recordings are commonly collected by independent devices or by multiple recording channels that are expected to capture the same acoustic scene. In practice, these recordings may not be perfectly synchronized. Clock mismatch can introduce constant offsets as well as nonlinear drift, and real-world recordings may also suffer from intermittent signal overlap, background noise, and domain-dependent acoustic conditions. As a result, analyses that rely on relative timing across channels, such as localization or multi-recorder event comparison, can become unreliable without post-hoc temporal alignment [3, 4].

BioDCASE 2026 Task 1 addresses this problem by requiring systems to predict the corresponding Channel 1 timestamps for a sequence of given Channel 0 keypoints [1]. A simple global shift is not sufficient because the drift may vary over time. Classical correlation-based methods such as GCC-PHAT can be effective when the same event is clearly observed by both channels [2], but they may fail when the correlation peak is weak or ambiguous. In contrast, neural networks can learn more robust patterns from data, but they may over-smooth the delay trajectory or behave unreliably when the signal has an easily identifiable correlation peak that a traditional method can estimate directly.

Our submissions are built around these complementary properties. We submit three systems: a neural delay regression system, a confidence-routed hybrid system without additional post-processing, and a confidence-routed hybrid system with trajectory-level post-processing. The hybrid systems use a GCC-PHAT branch for recordings whose correlation evidence is reliable and a neural CV-Peak-BLSTM branch for recordings whose correlation evidence is weaker. The routing decision is made by an average GCC-PHAT peak confidence score.

The main components of our approach are as follows. First, we construct frame-level two-channel STFT magnitude features and align delay targets to the STFT frame grid [5]. Second, we estimate a dense delay curve with a sliding-window GCC-PHAT branch. Third, we train a cost-volume neural model that compares the two channels over candidate temporal lags and uses BLSTM context to regress a continuous delay trajectory [8, 9, 10, 11]. Finally, for the post-processed hybrid system, we apply a trajectory-level post-processing stage to the neural-branch output to improve temporal consistency while preserving useful local variations.

2. PROBLEM FORMULATION

Let a stereo recording be denoted as

$$\mathbf{x} = \{x_0[n], x_1[n]\}, \tag{1}$$

where $x_0[n]$ and $x_1[n]$ are Channel 0 and Channel 1, respectively. For each recording, the task provides a sequence of Channel 0 timestamps

$$\mathcal{T}^{(0)} = \{t_i^{(0)}\}_{i=1}^{N}. \tag{2}$$

The goal is to estimate the corresponding Channel 1 timestamps

$$\hat{\mathcal{T}}^{(1)} = \{\hat{t}_i^{(1)}\}_{i=1}^{N}. \tag{3}$$

We model the temporal relation between the two channels by a time-varying delay function $\Delta(t)$. For a given Channel 0 timestamp, the corresponding Channel 1 timestamp can be written as

$$t_i^{(1)} = t_i^{(0)} + \Delta(t_i^{(0)}). \tag{4}$$

Therefore, the system predicts the delay value at each requested keypoint and obtains

$$\hat{t}_i^{(1)} = t_i^{(0)} + \hat{\Delta}(t_i^{(0)}). \tag{5}$$

The official evaluation compares the predicted Channel 1 timestamps with the ground-truth timestamps using mean absolute error (MAE) in milliseconds and mean squared error (MSE) in sec^2.

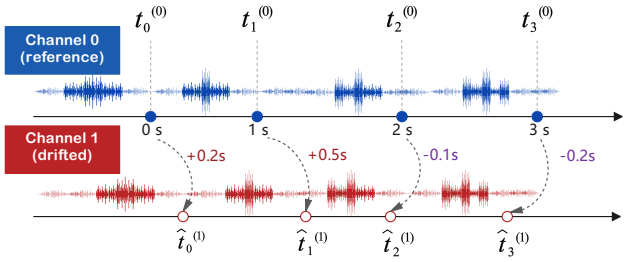

Figure 1: Illustration of the multichannel alignment problem. Channel 0 provides reference keypoints, while Channel 1 contains the corresponding acoustic events with time-varying drift. The task is to predict the Channel 1 timestamps from the given Channel 0 timestamps.

3. METHOD

All submitted systems formulate multichannel alignment as time-delay estimation. Given a stereo recording and a set of Channel 0 keypoints, the system estimates a delay trajectory from the waveform and predicts the corresponding Channel 1 timestamps by adding the estimated delay to the Channel 0 timestamps.

Figure 2 shows the confidence-routed hybrid framework used by our two hybrid submissions. It combines a GCC-PHAT branch and a neural CV-Peak-BLSTM branch using a recording-level confidence router. The neural-only submission uses only the neural branch. The post-processed hybrid submission further applies trajectory-level post-processing to the neural-branch output, as described in Section 3.5.

3.1. Feature and target construction

All audio is resampled to 16 kHz. For the neural component, we compute two-channel STFT magnitude features using a 2048-point FFT, a Hann window of 2048 samples, and a hop length of 800 samples. At 16 kHz, the hop length corresponds to 50 ms. The DC frequency bin is removed, so the neural model takes a feature tensor with shape $2 \times 1024 \times T$ as input.

During training, ground-truth timestamp pairs are converted into frame-level delay annotations. The model predicts one delay value per STFT frame, and only frames with valid targets contribute to the masked regression loss. At inference time, the frame-level delay predictions are linearly interpolated to the requested Channel 0 keypoints.

3.2. GCC-PHAT delay estimation

The GCC-PHAT component estimates a delay trajectory directly from the waveform. We use 1 s Hann-windowed segments with a 20 ms hop to compute local cross-correlation peaks. Very low-energy windows are ignored because their peaks are unstable. The lag search range is limited to ± 1 s. After local peak lags are collected across the recording, the resulting delay trajectory is interpolated to the requested keypoints and smoothed with a median filter of kernel size 5.

In our implementation, the GCC peak lag follows the convention $\tau(t) \approx t^{(0)} - t^{(1)}$. Therefore, the delay added to the Channel 0 timestamp is computed as

$$\hat{\Delta}_{\mathrm{gcc}}(t) = -\frac{\hat{\tau}(t)}{f_s}. \tag{6}$$

3.3. Neural delay regression

The neural component predicts frame-level delays from two-channel STFT magnitude features. The model first encodes the two channels with a shared frequency encoder. It then compares the encoded features over candidate temporal lags using a groupwise lag cost volume. Cost-volume representations are widely used in learned correspondence estimation [8, 9], and here they are used to describe how well the two audio channels match under different time shifts.

After cost-volume construction, lag-time convolutional blocks and peak-aware lag pooling are used to summarize the lag response at each time frame. A bidirectional LSTM models the temporal evolution of the delay trajectory [10, 11]. Finally, an MLP regression head outputs delays in seconds. The predicted delay trajectory is linearly interpolated to the requested Channel 0 keypoints.

3.4. Confidence-based hybrid routing

The hybrid system combines the GCC-PHAT and neural components using a recording-level confidence score. The router computes an average GCC-PHAT peak confidence using 1 s windows with a 50 ms hop. If the average confidence score is larger than 0.04, the GCC-PHAT branch is used; otherwise, the neural branch is used:

$$\hat{\Delta}(t) = \begin{cases} \hat{\Delta}_{\mathrm{gcc}}(t), & q_{\mathrm{gcc}} > 0.04, \\ \hat{\Delta}_{\mathrm{nn}}(t), & q_{\mathrm{gcc}} \le 0.04. \end{cases} \tag{7}$$

This routing strategy is designed to use GCC-PHAT when the cross-channel correlation peak is reliable and to use the neural model when the correlation structure is weak or ambiguous. In this way, the hybrid system avoids forcing all recordings into a single estimator.

3.5. Trajectory-level post-processing

For the first submitted system, we apply an additional trajectory-level post-processing stage to the neural-branch delay trajectory after the neural branch shown in Fig. 2. The GCC-PHAT branch is kept unchanged in both hybrid systems. Therefore, the difference between System 1 and System 3 only lies in whether the neural delay trajectory is post-processed before being used in the confidence-routed hybrid output.

The post-processing estimates a smooth global trend from the raw neural delay trajectory. Frame-level confidence is used to assign larger weights to more reliable delay estimates. The final neural trajectory is obtained by combining the fitted global trend with the residual component of the raw prediction. This keeps the output close to the original neural estimator while reducing unstable local fluctuations.

This post-processing is enabled for System 1 and disabled for System 3. In both systems, recordings routed to the GCC-PHAT branch use the same GCC-PHAT delay estimation procedure described above.

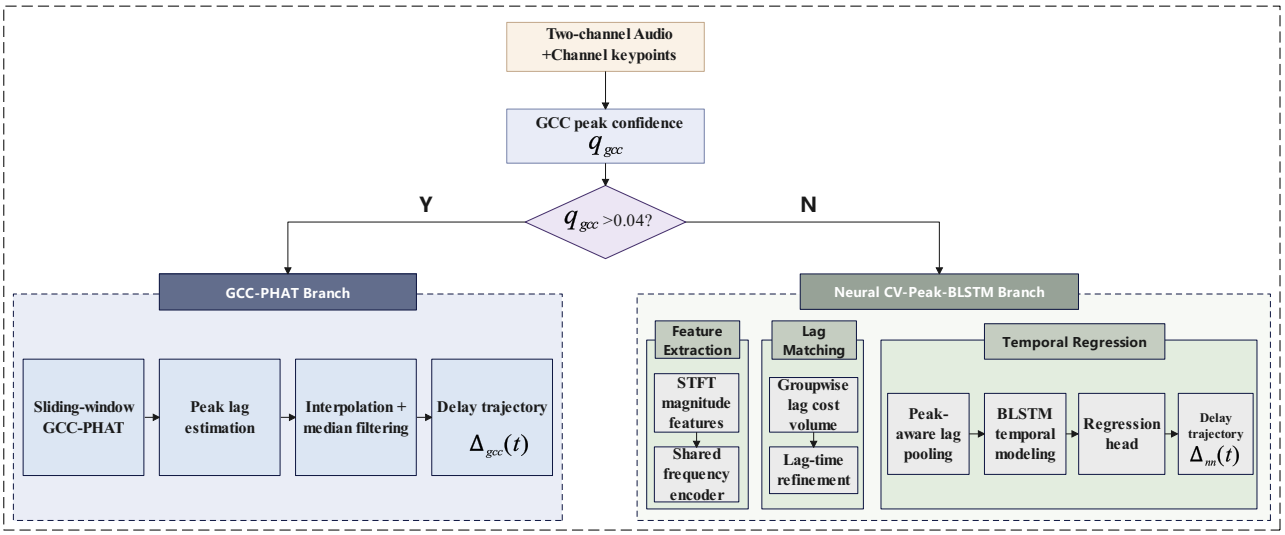

Figure 2: Overview of the confidence-routed hybrid framework used by the two hybrid submissions. One hybrid system further applies trajectory-level post-processing to the neural-branch output.

Table 1: Overview of the submitted systems.

System	Label suffix	Variant	Configuration
1	task1_1	Hybrid + PP	Confidence-routed hybrid system with neural-branch trajectory-level post-processing
2	task1_2	Neural	CV-Peak-BLSTM frame-level delay regression system
3	task1_3	Hybrid	Confidence-routed hybrid system without additional post-processing

4. SUBMITTED SYSTEMS

We submitted three systems for Task 1. All systems follow the same input-output format: given a stereo recording and a set of Channel 0 keypoints, each system estimates the corresponding Channel 1 timestamps. The submitted systems differ mainly in the delay estimator and whether trajectory-level post-processing (PP) is applied. All system labels share the prefix Huang_WHU_. Table 1 reports the label suffixes and corresponding configurations.

The hybrid framework shown in Fig. 2 is used by System 1 and System 3. System 1 applies the trajectory-level post-processing described in Section 3.5, while System 3 directly uses the delay trajectory produced by the selected branch. System 2 uses only the neural delay regression model. Each system produces separate prediction files for the required evaluation subsets. All output files follow the official CSV format with the columns "Filename", "Time Channel 0", and "Time Channel 1". The metadata files specify the corresponding system labels, system descriptions, authors, validation results, and external data usage.

5. TRAINING AND INFERENCE SETUP

The neural branch is trained on the official development data. The task contains two acoustic domains, aru and zebra_finch. The two domains have different recording conditions and different alignment difficulty, so we report validation results separately for both datasets. No external data is used in our submitted systems.

The neural model is optimized with masked mean absolute error:

$$\mathcal{L}_{\mathrm{MAE}} = \frac{\sum_{b,t} M_{b,t} |\hat{\Delta}_{b,t} - \Delta_{b,t}|}{\sum_{b,t} M_{b,t} + \epsilon}, \tag{8}$$

where $M_{b,t}$ is the target mask. We use AdamW with an initial learning rate of 1×10^{-4} [14]. Training is run with early stopping based on validation MAE, and the checkpoint with the lowest validation MAE is selected for inference. The neural models are implemented in PyTorch [15].

The main preprocessing and inference settings are shared across the submitted systems. All audio is resampled to 16 kHz. The neural branch uses STFT magnitude features and predicts frame-level delays in seconds. The GCC-PHAT branch estimates local delays with sliding windows, and the hybrid systems use a fixed GCC confidence threshold of 0.04 for branch selection. The trajectory-level post-processing is enabled only for System 1. No external data is used.

At inference time, all submitted systems predict delay values and convert them into Channel 1 timestamps according to Eq. (5). For the hidden evaluation data, the same inference procedure is applied to all required subsets, including Greater flamingo, Hadada ibis, and Red-billed quelea. No species-specific manual adjustment is used in the final submitted outputs.

6. RESULTS

We evaluate the submitted systems on the validation set using MAE in milliseconds and MSE in sec^2. Results are reported separately for aru and zebra_finch. The average is computed as the unweighted mean over the two validation subsets.

The validation results show that System 1, the post-processed hybrid system, achieves the lowest average MAE among our submitted systems. System 3 uses the same confidence-routed hybrid framework without the additional neural-branch post-processing,

Table 2: Validation set performance comparison. MAE is reported in ms and MSE is reported in sec^2.

Method	aru		zebra_finch		Average	
	MAE (ms)	MSE	MAE (ms)	MSE	MAE (ms)	MSE
Nosync	116.2	0.019	2297.4	5.734	1206.8	2.8765
GCC-PHAT	122.4	0.020	2457.4	7.995	1289.9	4.008
Official DL baseline	164.3	0.040	757.1	1.218	460.7	0.629
System 1 (Hybrid + PP)	**0.654**	**0.000087**	**416.14**	**0.262**	**208.397**	**0.131**
System 2 (Neural)	118.4	0.020	422.91	0.262	270.637	0.141
System 3 (Hybrid)	0.654	0.000087	422.91	0.263	211.782	0.131

and System 2 provides a neural-only reference. The comparison between System 1 and System 3 indicates that the trajectory-level post-processing mainly improves the zebra_finch validation results while keeping the aru results unchanged.

7. CONCLUSION

This report summarizes our submitted systems for BioDCASE 2026 Task 1. The systems formulate multichannel alignment as time-delay estimation and generate Channel 1 timestamps by adding the estimated delay to each given Channel 0 keypoint. The submitted systems include a neural frame-level delay regressor, a confidence-routed hybrid system without post-processing, and a confidence-routed hybrid system with neural-branch trajectory-level post-processing. The validation results show that the hybrid framework performs well across the two validation subsets, and the additional post-processing further improves the final submitted system.

8. REFERENCES

[1] BioDCASE 2026 Task 1: Multichannel Alignment. [Online]. Available: `https://biodcase.github.io/challenge2026/summary`

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

[3] M. Brandstein and D. Ward, Eds., *Microphone Arrays: Signal Processing Techniques and Applications*. Berlin, Heidelberg: Springer, 2001.

[4] J. Benesty, J. Chen, and Y. Huang, *Microphone Array Signal Processing*. Berlin, Heidelberg: Springer, 2008.

[5] J. B. Allen and L. R. Rabiner, "A unified approach to short-time Fourier analysis and synthesis," *Proceedings of the IEEE*, vol. 65, no. 11, pp. 1558–1564, 1977.

[6] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[7] L. R. Rabiner, A. E. Rosenberg, and S. E. Levinson, "Considerations in dynamic time warping algorithms for discrete word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 6, pp. 575–582, 1978.

[8] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "FlowNet: Learning optical flow with convolutional networks," in *Proc. IEEE International Conference on Computer Vision*, 2015, pp. 2758–2766.

[9] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8934–8943.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5–6, pp. 602–610, 2005.

[12] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," arXiv preprint arXiv:1607.06450, 2016.

[13] Y. Wu and K. He, "Group normalization," in *Proc. European Conference on Computer Vision*, 2018, pp. 3–19.

[14] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[15] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.