Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

YAPAT — An Open-Source Platform for Active and Weakly Supervised Annotation of Passive Acoustic Monitoring Data

Thiago S. Gouvêa

thiago.gouvea@dfki.de

German Research Centre for Artificial Intelligence

Rida Saghir

German Research Centre for Artificial Intelligence

Prathmesh P. Doddanawar

German Research Centre for Artificial Intelligence

Novruz Mammadli

German Research Centre for Artificial Intelligence

Pratik P. Sitapara

German Research Centre for Artificial Intelligence

Hannes Kath

German Research Centre for Artificial Intelligence

Daniel Sonntag

German Research Centre for Artificial Intelligence

Article

Keywords:

Posted Date: July 31st, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10144323/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

YAPAT — An Open-Source Platform for Active and Weakly Supervised Annotation of Passive Acoustic Monitoring Data

Thiago S. Gouvêa[1,2,*], Rida Saghir[1,2,*], Prathmesh P. Doddanawar[1], Novruz Mammadli[1,2], Pratik Sitapara[1,2], Hannes Kath[1,2], and Daniel Sonntag[1,2]

[1]German Research Center for Artificial Intelligence (DFKI), Interactive Machine Learning, Oldenburg, Germany
[2]Carl von Ossietzky University of Oldenburg, Applied AI, Oldenburg, Germany
[*]Corresponding authors: {thiago.gouvea,rida.saghir}@dfki.de

ABSTRACT

Passive Acoustic Monitoring (PAM) enables non-invasive, large-scale assessment of animal biodiversity, but the resulting audio archives routinely exceed what domain experts can annotate manually. Existing annotation tools address parts of this challenge through spectrogram visualisation, automated inference, or collaborative review, but none support the full annotation cycle needed to produce high-quality, interoperable datasets at scale. We present YAPAT (*Yet Another PAM Annotation Tool*), a web-based platform for interactive machine learning applied to the curation of annotated PAM datasets. YAPAT integrates four capabilities that, in combination, are absent from prior tools: (i) ontology-grounded label spaces that link annotations to authoritative biological and environmental taxonomies (GBIF, ENVO), with explicit local concept support, supporting dataset interoperability and reuse; (ii) multi-label active learning that ranks unlabelled snippets by a composite informativeness score, reducing the expert effort required to produce a well-labelled dataset; (iii) two complementary cold-start mechanisms: upload-based vector similarity search over frozen BirdNET embeddings, enabling annotation before any classifier exists; and weakly supervised sound event detection via multiple instance learning, enabling classifier initialisation from sound collections labelled at the file level; and (iv) interactive 2D projection of BirdNET embeddings for exploratory annotation. The system is collaborative, browser-based, and open source, and produces reusable, ontology-grounded annotation datasets suitable for deposition and downstream analysis.

1 Introduction

Passive Acoustic Monitoring (PAM) has become a key method for assessing animal biodiversity, using acoustic signals to observe ecosystems at scale with minimal disturbance[1]. By deploying autonomous recording units, ecologists collect continuous audio data across large spatial and temporal extents but the resulting archives routinely exceed what a domain expert can review manually. Machine learning methods can automate species detection and classification, but they typically require labelled training data that must itself come from expert annotation, a bottleneck that, in practice, restricts machine learning applications to the subset of taxa and habitats for which training data or pre-trained models already exist.

Existing annotation tools address parts of this challenge. Desktop software such as Raven Pro[2] provides powerful spectrogram visualisation and manual annotation, with support for energy-based detectors and integration of externally trained classifiers through its Learning Detector interface. Tools such as BirdNET-Analyzer[3] and Koogu[4] support automated inference using pre-trained or custom classifiers. Web-based platforms such as ARBIMON[5] and Whombat[6] extend these capabilities to collaborative workflows. Concurrently, Dumoulin et al.[7] apply human-in-the-loop annotation with iterative retraining[8,9] to bioacoustics under the name *agile modeling*[10], combining precomputed foundation-model embeddings with vector similarity search, a linear classifier, and an active learning loop—sharing YAPAT's core technical paradigm, but as a research pipeline rather than a deployable collaborative platform. Importantly, none of the available tools support the full annotation cycle needed to produce high-quality, interoperable datasets at scale (Table 1): ontology-grounded labels, weak supervision, and the combination of active learning with vector similarity search are not jointly available in any existing tool.

We present YAPAT (*Yet Another PAM Annotation Tool*), a web-based platform for annotation of PAM datasets using interactive machine learning workflows. The primary challenge addressed is corpus-scale annotation: producing high-quality, ontology-grounded labels across archives spanning many hours, where manual review is intractable and target taxa may initially have no labelled examples.

YAPAT is distinguished by four capabilities that, in combination, are not found in prior work. First, labels are grounded to authoritative biological and environmental ontologies (GBIF, ENVO) at annotation time, with explicitly marked local concepts

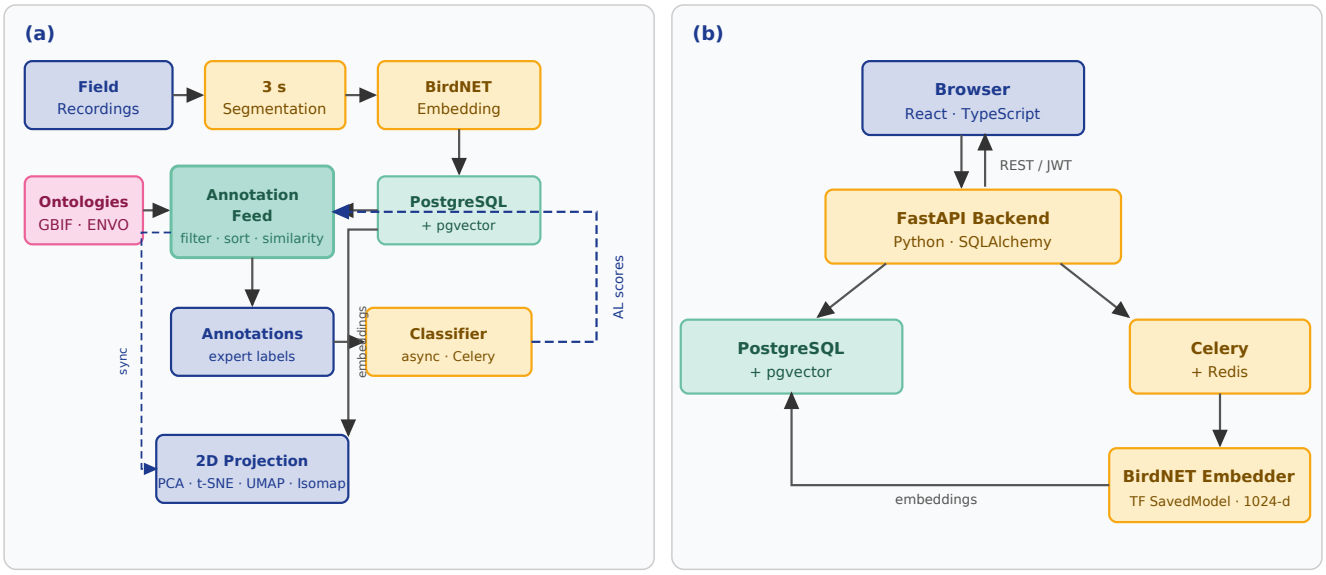


Figure 1. YAPAT system overview. (**a**) Pipeline from field recordings to a curated, ontology-grounded annotation dataset. Recordings are segmented into 3 s snippets and embedded with BirdNET; labels are grounded to GBIF and ENVO at annotation time. The annotator configures the feed using filter and sort controls, or queries the dataset by acoustic similarity; feedback triggers asynchronous retraining. (**b**) Component diagram showing the FastAPI backend, React frontend, and supporting services (PostgreSQL/pgvector, Celery task queue, BirdNET embedder). Colour code: blue—data and user-facing components; green—core YAPAT storage and annotation interface; orange—computation and ML processing; pink—external ontological resources (GBIF, ENVO).

as fallback, supporting interoperability and dataset reuse[11,12]. Second, multi-label active learning selects snippets for annotation based on a composite score balancing per-label uncertainty, representational diversity, and embedding-space density, enabling systematic reduction of annotation effort. Third, two complementary mechanisms address the cold-start problem: upload-based vector similarity search retrieves acoustically similar PAM snippets from a reference clip using frozen BirdNET embeddings, enabling annotation before any classifier exists; and weakly supervised sound event detection (WSSED) via multiple instance learning (MIL) initialises classifiers from sound collections annotated at the file level without requiring any pre-labelled PAM data. Fourth, an interactive 2D embedding space projection (using PCA, t-SNE, UMAP, Isomap) over BirdNET embeddings supports exploratory annotation; the projection view colour-codes snippets by annotation status and couples point clicks to annotation cards in the feed, following the design of Saghir et al.[13]. The data model is designed to accommodate additional embedding models without pipeline changes. These capabilities are integrated into a collaborative, browser-based interface deployable as a self-hosted web service via Docker, requiring no client-side installation beyond a web browser.

Component-level evaluations are reported in companion publications. Kath et al.[14] compared pre-trained feature extractors and showed that active learning with the best-performing representation substantially reduces annotation effort. Mammadli et al.[15,16] investigated MIL for weakly supervised sound event detection, generating temporally precise predictions from file-level annotations using public sound libraries. Saghir et al.[13] evaluated interactive embedding-space navigation for human-centric active learning, finding it improved annotator engagement while matching the performance of standard AL strategies. This paper describes the integrated system, its design rationale, and the scalability benchmarks relevant to practitioners choosing a deployment environment.

2 Results

2.1 System Overview

YAPAT is organised around *teams* and *datasets* (Fig. S1). A team corresponds to a group of collaborators sharing access to one or more datasets. Each dataset contains *recordings*—arbitrary-length audio files from field deployments—which are automatically segmented into fixed-length *snippets* (default: 3 s, non-overlapping). Snippets are embedded using BirdNET[3] (1024-dimensional vectors; 3 s windows at 48 kHz), with vectors stored in a PostgreSQL database extended with pgvector[17] for efficient cosine-similarity retrieval. The data model supports multiple embedding models, allowing future substitution of domain-specific alternatives without changes to the annotation pipeline. Annotations are stored at snippet level and linked to

identifiers from the GBIF Backbone Taxonomy[18] and ENVO[19,20] for biological and environmental terms respectively. Fig. 1 shows a scheme of the pipeline and component diagram. A detailed description of the system architecture, database schema, and REST API is provided in the Supplementary Information.

2.2 Annotation Interface

The primary interface is an *annotation feed*: a paginated, scrollable sequence of snippet cards, each displaying a mel-spectrogram rendered in the browser, an audio player, and a multi-label input field with GBIF and ENVO autocomplete search, with selected labels displayed as chips. The feed is configured through two orthogonal controls—*filter* and *sort*—that together determine which snippets are presented and in what order (Fig. 2).

Filter restricts the pool of snippets shown. Users can filter by annotation status (unannotated, annotated, or all), by recording metadata (monitoring station, date range, time of day), and by active learning scores (Section 2.4). Filters can be combined, supporting targeted workflows such as reviewing unannotated overnight recordings from a specific station where the predicted probability of occurrence of a target species exceeds a set threshold.

Sort determines the ranking of snippets within the filtered pool. To support common annotation workflows, four named presets are provided. *Validate* ranks by classifier confidence (descending), enabling systematic review of model predictions before they are committed as annotations, and is particularly useful for auditing WSSED initialisation output (Section 2.3). *Refine* ranks by the composite active learning score (descending), directing expert effort toward the most informative samples (Section 2.4). *Discover* ranks by diversity and density scores to surface snippets from underexplored regions of the embedding space, supporting efficient identification of novel species or sound classes[21]. *Custom* exposes all individual scores—confidence, uncertainty, diversity, density—for manual configuration. A *Random* order is also available for general-purpose annotation without model involvement.

Similarity search is available as an in-context discovery tool: the user selects any snippet in the feed and retrieves its nearest neighbours in embedding space, supporting rapid exploration of acoustically similar events during an active annotation session.

Embedding space projection. Alongside the annotation feed, YAPAT renders an interactive 2D scatter plot of the BirdNET embedding space using a user-selected dimensionality reduction method (PCA, t-SNE, UMAP, or Isomap). Each point corresponds to a snippet; points are coloured according to their label assignment, snippets that match the current feed filter but are unannotated are shown in dark grey, and filtered-out snippets appear in light grey. The selected snippet is highlighted with a visible outline across all views: miniature previews of dimensionality reduction methods run alongside the main plot, letting the user switch projection perspectives at any time while maintaining selection context. Clicking a point raises the corresponding annotation card in the feed, coupling the spatial view with the annotation workflow. This design is informed by Saghir et al.[13], who study interactive embedding-space navigation for human-centric active learning.

2.3 Cold-Start Mechanisms

A fundamental challenge in PAM annotation is the *cold-start problem*[22]: model-assisted recommendation of informative samples for annotation presupposes a trained classifier, yet the labelled PAM data required for training is precisely what the annotation effort aims to produce. YAPAT addresses this through two complementary cold-start mechanisms that differ in their prerequisites.

Similarity-based exploration requires only a reference audio clip. The user uploads an external clip—for example, a focal recording of a target species from a sound library—which is embedded on the fly using BirdNET and used to query the full dataset by cosine similarity. Results are returned as a standard annotation feed, allowing the user to confirm or reject candidate matches and accumulate an initial set of labelled snippets without any prior training. Vector similarity search over precomputed embeddings is a standard retrieval operation[17], and BirdNET embeddings have been shown to capture relevant structure in bioacoustic data[7,14,23].

Weakly supervised sound event detection (WSSED) via multiple instance learning (MIL)[24] is available when a collection of audio recordings with reliable file-level species labels exists; temporal localisation is not required. Focal recordings—short, species-directed clips from archival sound libraries such as Xeno-Canto or FNJV (Fonoteca Neotropical Jacques Vielliard)—are a natural source, but any labelled audio collection satisfies this requirement. Each clip is treated as a bag of 3 s segments: BirdNET embeddings are extracted per segment (transfer learning[14]) and passed to a linear classifier trained with binary cross-entropy loss and average pooling aggregation[16,24]. Because the BirdNET encoder is kept frozen, only the linear head is updated, enabling rapid training even from small collections.

The trained classifier is then applied to BirdNET embeddings of PAM recordings to produce segment-level predictions, bridging the domain gap between the clean focal recordings and the noisy, multi-species PAM environment. These predictions seed a confidence-ranked review using the *Validate* preset (Section 2.2): the annotator inspects top-ranked snippets and confirms or corrects the classifier output, bootstrapping the active learning cycle (Section 2.4, Fig. 4). The full pipeline is illustrated in Fig. 3.



Figure 2. Annotation interface. (**Left**) Interactive 2D embedding space projection of all dataset snippets (AnuraSet; PCA). Points are colour-coded by species annotation; unannotated snippets within the active filter appear in dark grey, and filtered-out snippets in light grey. The selected point (circled) is coupled to the snippet card shown in the feed. (**Right**) Annotation feed displaying one snippet card: mel-spectrogram (top), audio player, and multi-label annotation area with GBIF and ENVO label chips. The top navigation bar provides access to dataset selection and application settings; sort and filter controls in the left panel determine which snippets are shown and in what order.

Evaluation of the WSSED component under domain shift, including transfer from FNJV focal recordings to AnuraSet PAM data, is reported in Mammadli et al.[15,16].

2.4 Active Learning

YAPAT supports multi-label active learning across a configurable *label scope*—a subset of the full label set selected for a particular annotation session. Acquisition scores are computed with respect to the active scope rather than the full taxonomy.

After each retraining cycle, three acquisition scores are computed for every unlabeled snippet. *Uncertainty* is the mean per-label Bernoulli entropy over the label scope:

$$\mathrm{unc}(x) = \frac{1}{|\mathscr{S}|} \sum_{k \in \mathscr{S}} H(p_k(x)), \quad H(p) = -p \log p - (1-p)\log(1-p), \tag{1}$$

where $\mathscr{S}$ is the label scope and $p_k(x)$ is the predicted probability for label k.

Diversity is the distance from a snippet's embedding to its nearest labelled neighbour, encouraging coverage of underexplored regions of the embedding space.

Density is a k-nearest-neighbour density estimate within the unlabelled pool, down-weighting isolated outliers that are unlikely to be representative of the broader data distribution.

Uncertainty, diversity, and density are established active learning criteria[8], and their combination into a composite acquisition score has been previously proposed[25]. YAPAT adopts this approach; the *Refine* sort preset ranks snippets by composite score in descending order (Section 2.2).

For display, a multi-label confidence score is computed as the noisy-OR over the label scope: $\mathrm{conf}(x) = 1 - \prod_{k \in \mathscr{S}} (1 - p_k(x))$. This provides an intuitive summary of the model's overall belief that at least one label in scope is present.

Retraining is triggered automatically after a configurable number of new annotations accumulate, or on demand. The classifier is a linear head over frozen BirdNET embeddings, enabling retraining in seconds on CPU without blocking the annotation interface (Section 2.6). All inference and retraining jobs are dispatched asynchronously via Celery[26] and do not interrupt ongoing annotation sessions.

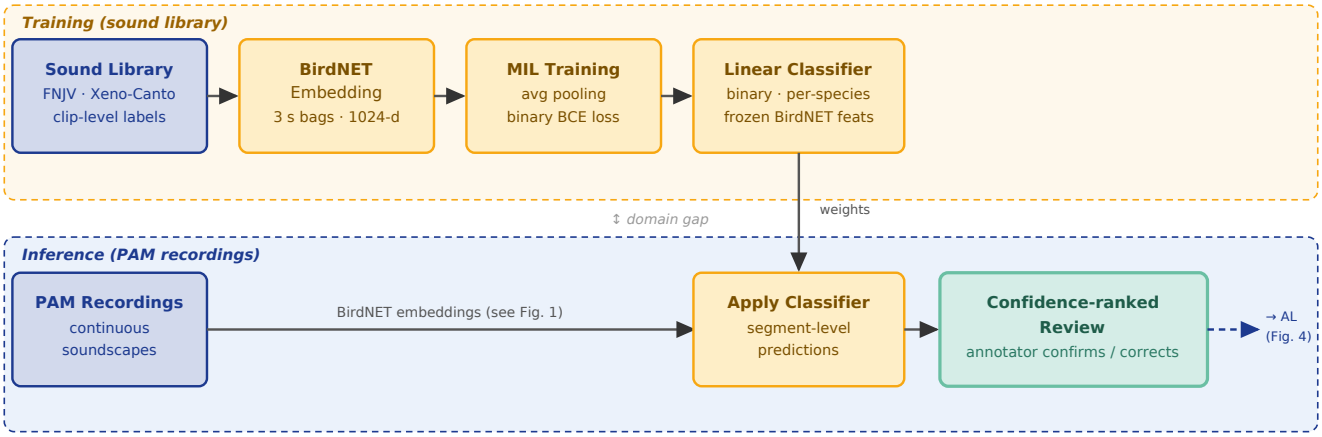


Figure 3. Weak supervision pipeline and cold-start initialisation. (*Training*, top) Focal recordings from archival sound libraries (FNJV, Xeno-Canto) carry file-level species labels. Each recording is segmented into 3 s windows and embedded with BirdNET; a linear classifier is trained under a multiple instance learning formulation using average pooling and binary cross-entropy loss. (*Inference*, bottom) The trained classifier is applied to BirdNET embeddings of PAM recordings (produced by the pipeline in Fig. 1), yielding segment-level predictions. The annotator sorts the annotation feed by classifier confidence and confirms or corrects the top-ranked snippets; these initial annotations bootstrap the active learning cycle (Fig. 4).

Component-level evaluation of the transfer learning and active learning pipeline on AnuraSet and two additional PAM datasets is reported in Kath et al.[14]. The interactive 2D embedding space projection described in Section 2.2 is derived from the representation-centric AL interface of Saghir et al.[13].

2.5 Comparison with Existing Tools

Table 1 summarises the feature landscape across representative PAM annotation tools. YAPAT is the only tool that combines ontology-grounded label spaces, active learning, weak supervision, vector similarity search, and interactive embedding space projection in a web-based, multi-user platform. The closest alternative, ARBIMON[5], is a web-based collaborative platform that supports multi-label annotation, similarity-based retrieval, and model training, but does not implement ontology-grounded label spaces, acquisition-function-guided query selection, or WSSED initialisation. The direct predecessor of YAPAT, described in Kath et al.[9], introduced a human-in-the-loop annotation interface with latent space visualisation, but was limited to single-label annotation, supported only uncertainty-based query selection without diversity or density scoring, and lacked WSSED initialisation. Where Kath et al. display a fixed two-dimensional VAE latent space, YAPAT applies user-selectable dimensionality reduction (PCA, t-SNE, UMAP, Isomap) to high-dimensional BirdNET embeddings, with point selection coupling directly to annotation cards in the feed. Dumoulin et al.[7] present the most methodologically similar external work: agile modeling shares YAPAT's pipeline of precomputed embeddings, similarity-based cold start, linear-head classifier, and active learning loop, and is evaluated on AnuraSet alongside real-world field deployments; it does not provide a collaborative web interface, ontology-grounded labels, or WSSED initialisation, and is available only as a notebook rather than a deployable platform.

2.6 Scalability

To characterise the hardware requirements of YAPAT, we benchmark operations that determine practical session size, using AnuraSet ($N = 30{,}687$ snippets). All backend benchmarks were conducted inside a Docker container running TensorFlow 2.21.0, deployed on a workstation equipped with an AMD EPYC 9135 16-core processor, 755 GB RAM, and four NVIDIA RTX PRO 6000 Blackwell GPUs (96 GB VRAM each); GPU benchmarks used a single GPU via TensorFlow device placement (`/GPU:0`). Frontend benchmarks were measured using Playwright headless Chromium on an Apple M4 Pro MacBook, reflecting client-side WebGL rendering performance on commodity hardware. Fig. 5 summarises the measured scaling behaviour.

Initial ingestion. Initial ingestion comprises three sequential steps. Audio discovery scans 1,612 recordings in 21.6 s; snippet generation segments all recordings into 30,687 three-second clips in 12.4 s. Extracting BirdNET embeddings is the dominant one-time cost, scaling linearly, $O(N)$, on both CPU and GPU (Fig. 5a). Below $N \approx 10{,}000$, GPU embedding is fast relative to CPU but the apparent speedup is inflated by initialisation overhead; above this threshold both curves scale linearly, with GPU providing a stable 3× advantage. On CPU, full-dataset embedding takes approximately 150 s; on GPU, approximately 52 s. Embeddings are cached and not recomputed unless the snippet set changes.

Cold-start similarity search. Upload-based vector similarity search executes a single cosine nearest-neighbour query

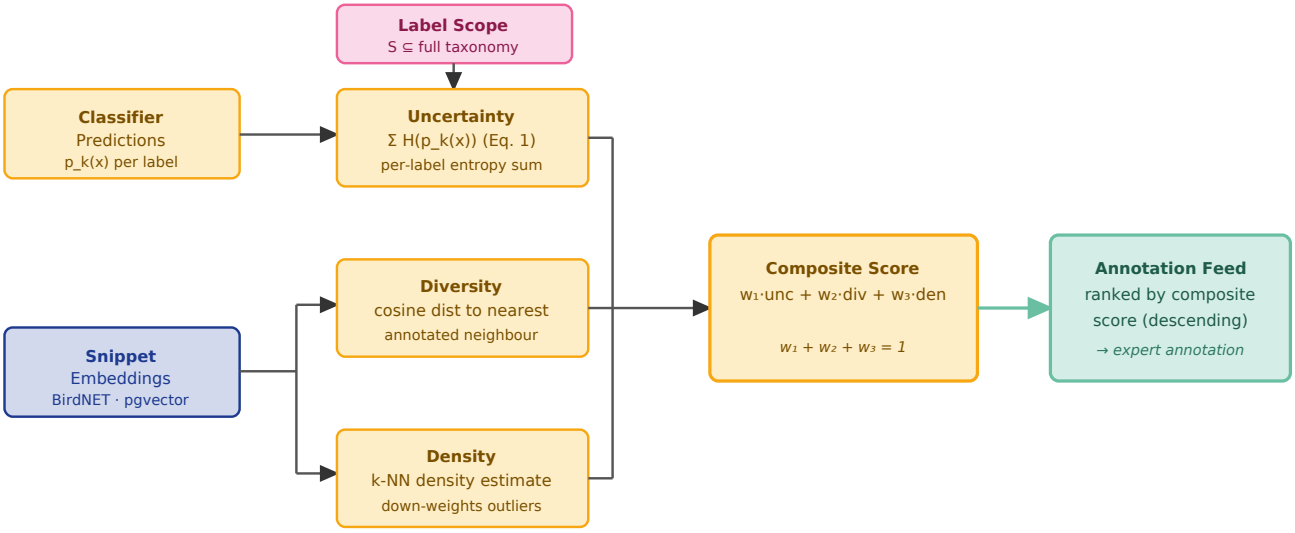


Figure 4. Active learning pipeline. After each retraining cycle, the model scores all unlabeled snippets on uncertainty (Eq. 1), diversity, and density. The composite score determines the AL feed order. Annotations accumulated through the feed trigger the next asynchronous retraining cycle.

against all N stored embeddings using the pgvector <=> operator. In practice, query latency is effectively constant at approximately 0.56 s across the full benchmarked range ($N = 5{,}000$–$30{,}687$), well within interactive response time for any realistic PAM deployment.

Per-cycle AL scoring. After each retraining, three acquisition scores are computed for every unlabeled snippet (Fig. 5d). Benchmarks were run on $N = 20{,}306$ unlabelled snippets with a fixed labelled set of $N_l = 10{,}381$. Uncertainty is the sum of per-label Bernoulli entropies over pre-computed model predictions, $O(N)$, and remains near-constant across the full range (0.03–0.12 s). Density uses k-nearest-neighbour search with an HNSW index[28], scaling as $O(N \log N)$ from 0.05 s to 1.6 s. Diversity, which compares each unlabelled embedding against the full labelled set, $O(N \cdot N_l)$, is the dominant cost, reaching 4.1 s at full scale; all three strategies remain within interactive response time for realistic annotation sessions.

Retraining. Classifier retraining updates only the linear head over frozen BirdNET embeddings, scaling linearly with the number of labelled snippets, $O(N_l)$ (Fig. 5c). GPU provides a 14–23× speedup across the benchmarked range; on CPU, retraining at $N_l = 10{,}381$ takes 44 s, while on GPU the same workload completes in 2.5 s. For typical early annotation cycles ($N_l < 2{,}000$), CPU retraining completes in under 9 s without blocking the annotation interface.

These benchmarks provide practical guidance for deployment. A CPU-only server is sufficient for datasets of similar size to AnuraSet (approximately 1,600 minutes), with embedding and retraining completing within minutes; GPU acceleration is recommended for larger datasets or workflows that require rapid annotation iteration. Because all retraining and inference jobs are executed asynchronously, the annotation interface remains responsive regardless of computation time.

Frontend rendering. The annotation feed uses scroll-based progressive loading (batches of 50 cards rendered incrementally via `IntersectionObserver`). The embedding space projection uses WebGL-accelerated rendering (`scattergl` via Plotly.js): measured render time is approximately 0.11 s and remains constant across the full dataset range ($N = 1{,}000$–$30{,}687$), consistent with GPU-side point rendering that is independent of dataset size. Switching between projection methods (PCA, t-SNE, UMAP, Isomap) takes approximately 0.21–0.23 s at $N = 30{,}687$. Initial page load at full dataset size takes approximately 2.2 s. Measured latencies for similarity search and frontend operations are summarised in Table 2.

3 Discussion

YAPAT embodies a deliberate trade-off: a web-based architecture eliminates installation barriers and enables collaborative annotation, at the cost of confining compute-intensive operations to the server side. The most demanding per-cycle operation is diversity scoring, which compares every unlabeled embedding against the growing labeled set and scales as $O(N \cdot N_l)$, setting the practical upper bound on interactive session size for large annotated corpora. Asynchronous execution via Celery ensures the annotation interface is never blocked; however, sessions on very large datasets may experience a delay between submitting feedback and receiving updated acquisition scores.

A second constraint is the dependency on BirdNET as the sole embedding model. BirdNET was trained on avian

Table 1. Comparison of YAPAT with representative tools for annotating bioacoustic recordings. ✓ = supported; × = not supported; ○ = partial or limited support. [a] BirdNET-Analyzer has no manual annotation interface; confidence-sorted review of automatic detections is assumed as a proxy workflow. [b] Raven Pro supports batch spectrogram correlation for pairwise similarity comparison; integration with annotation workflow is not documented. [c] Raven Pro's Learning Detector integrates externally trained models (e.g., via BirdNET-Analyzer or Koogu) but does not include built-in model training. [d] Whombat supports an export-label-retrain workflow but model training itself happens outside of Whombat.

Feature	**YAPAT** (this work)	Raven Pro[2]	BirdNET-Analyzer[3]	Koogu[4]	ARBIMON[5]	Kath et al. 2023[9]	Whombat[6]	Kaleidoscope Pro[27]
Deployment								
Web-based	✓	×	✓	×	✓	✓	✓	×
Multi-user / collaborative	✓	×	×	×	✓	×	✓	✓
Open source	✓	×	✓	✓	✓	×	✓	×
Annotation controls								
Multi-label annotation	✓	✓	×	×	✓	×	✓	×
Ontology-grounded labels	✓	×	×	×	×	×	×	×
Confidence-based triage	✓	✓	○[a]	×	✓	×	✓	×
Embedding space projection	✓	×	×	×	×	✓	×	×
Cold-start mechanisms								
Weak supervision cold-start (WSSED/MIL)	✓	×	×	×	×	×	×	×
Similarity search	✓	○[b]	×	×	✓	×	×	✓
Active learning & model training								
Model training	✓	○[c]	✓	✓	✓	✓	○[d]	✓
Active learning (query selection)	✓	×	×	×	×	✓	×	×
Uncertainty sampling	✓	×	×	×	×	✓	×	×
Diversity / density scoring	✓	×	×	×	×	×	×	×

vocalisations and may not generalise equally well to other taxa. In our deployment on AnuraSet (anurans), BirdNET embeddings support effective transfer, consistent with the findings of Kath et al.[14]. The cold-start quality of WSSED initialisation is bounded by the coverage of available focal recordings: taxa that are severely underrepresented in archival sound libraries cannot be reliably bootstrapped via weak supervision[16]. For such cases, YAPAT's similarity search can be used to manually seed a small initial set of confirmed positives without requiring focal recordings.

The current architecture fixes BirdNET as the embedder to reduce deployment complexity. A forthcoming version will support pluggable embedding models, enabling users to substitute domain-specific alternatives (e.g., models pre-trained on bat calls or marine mammal vocalisations) without changes to the annotation pipeline.

The primary deliverable of a YAPAT annotation campaign is the curated, ontology-grounded dataset, which can serve as training data for more capable classifiers, be deposited in public repositories, or reused in downstream analyses; trained linear-head classifiers can additionally be exported for standalone inference.

3.1 Limitations

Several limitations should be considered when deploying YAPAT. First, although the system is distributed as Docker containers, initial setup may require assistance from an IT professional, particularly for server configuration, network access, and storage mounting; it is not yet a point-and-click installation for non-technical users. Second, audio is segmented into fixed-length, non-overlapping windows (default: 3 s), which constrains both annotation precision and playback granularity: events that span window boundaries or occupy only a fraction of a window cannot be delineated temporally, and annotations cannot be restricted to a frequency band within a snippet. Third, the classifier is a linear head over frozen BirdNET embeddings. This design prioritises speed and deployability but precludes fine-tuning the encoder, data augmentation, or other advanced training strategies; classification performance is therefore bounded by the representational quality of the frozen embeddings for the target taxon. Finally, unlabelled snippets play no role in classifier training: only confirmed annotations are used to fit the linear head. The large proportion of unlabelled data in any realistic PAM archive therefore goes unused, whereas semi-supervised or

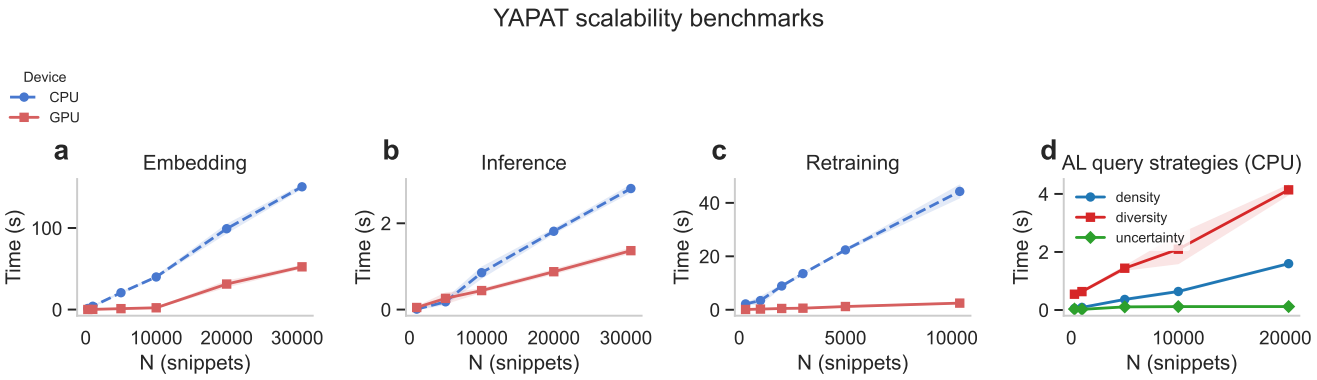


Figure 5. Scalability benchmarks on AnuraSet. Measured computation time as a function of dataset size N; shaded bands show ± 1 standard deviation over three runs. (**a**) BirdNET embedding extraction (one-time cost per dataset), $O(N)$, with CPU and GPU curves. GPU time is near-constant below $N \approx 10{,}000$ due to initialisation overhead; both curves scale linearly above this threshold. (**b**) Inference (forward pass of the linear head), $O(N)$; GPU overhead dominates at small N and provides approximately $2\times$ speedup at full dataset size. (**c**) Classifier retraining (linear head only), $O(N_l)$, measured against the number of labelled snippets N_l; GPU provides a 14–$23\times$ speedup across the benchmarked range. (**d**) Per-cycle active learning scoring for $N = 20{,}306$ unlabelled snippets and a fixed labelled set of $N_l = 10{,}381$: uncertainty ($O(N)$, entropy over pre-computed predictions), density ($O(N \log N)$ with HNSW index), and diversity ($O(N \cdot N_l)$).

self-supervised approaches—such as contrastive learning or pseudo-labelling over high-confidence model predictions—could exploit this pool to improve representation quality or classifier calibration. Integrating such methods with the active learning loop is a natural direction for future work.

4 Methods

4.1 Implementation

The YAPAT backend is implemented in Python using FastAPI[29] and SQLAlchemy, with a PostgreSQL database extended with pgvector[17] for vector similarity search. Long-running tasks—embedding computation, inference, and retraining—are managed by Celery[26] with a Redis message broker and do not block the annotation interface. The frontend is implemented in TypeScript using React 18, Vite, Tailwind CSS, Ant Design, and Redux Toolkit. Mel spectrograms are computed in the browser using a JavaScript mel filterbank implementation, avoiding server-side audio rendering.

BirdNET embeddings (version 2.4) are extracted via the TensorFlow SavedModel API (`call_endpoint="embeddings"`), producing 1024-dimensional `float32` vectors from 3-second audio segments resampled to 48 kHz. Batch embedding is supported for efficient initial processing of large datasets. Vector similarity search uses the `<=>` cosine distance operator

Table 2. Measured latency for similarity search and frontend operations on AnuraSet ($N = 30{,}687$ snippets). Values are mean and standard deviation over three to five runs; *render* was measured across multiple dataset sizes and is reported as an approximate constant.

Operation	N	Mean (s)	Std (s)
Similarity search (pgvector cosine)			
	5,000	0.561	0.004
	10,000	0.528	0.009
	20,000	0.560	0.006
	30,687	0.572	0.009
Frontend rendering			
Page load	30,687	2.207	0.137
Render (WebGL)	1,000–30,687	≈0.11	–
Switch to t-SNE	30,687	0.207	0.005
Switch to UMAP	30,687	0.228	0.029
Switch to Isomap	30,687	0.206	0.005

provided by pgvector.

The system is deployed behind an nginx reverse proxy with JWT-based authentication. Full architecture details, the database schema, and the REST API specification are provided in the Supplementary Information.

4.2 Datasets

Scalability benchmarks use AnuraSet[30], a publicly available multi-label PAM dataset from two Brazilian biomes (Cerrado savanna and Atlantic forest), annotated by experts for 42 species of anurans.

4.3 Code and Data Availability

YAPAT source code is available at https://github.com/yapat-app/yapat under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0) license. A demonstration instance is available at https://yapat.ni.dfki.de/. AnuraSet is available at https://zenodo.org/records/8056090.

Acknowledgements

We thank the bioacoustics team (led by Ivan B. Campos) in ICMBio's Monitora programme and the bioacoustics group (led by Simone Dena) of the Brazilian Team in the Xprize Rainforest competition for early testing of the tool and for generously sharing their experience and feedback. We thank Kanwar A. Ali for assistance with front-end development.

Funding

This research is part of the Computational Sustainability & Technology project area (https://cst.dfki.de/), and has been supported by the Lower Saxony Ministry of Science and Culture (MWK) in zukunft.niedersachsen program, the Federal Ministry of Research, Technology and Space (BMFTR) under grant number 16IW26002 (AMENABLE), and the Endowed Chair of AAI at the University of Oldenburg.

Author Contributions

T.S.G. conceptualised the study, defined system requirements and architecture, and wrote the original manuscript. R.S. developed the embedding visualisation and active learning interface, and contributed to the tool comparison. P.P.D. led software development and implemented the majority of the system components, and conducted all benchmarking experiments. N.M. designed the weakly supervised sound event detection component, and led the comparative analysis of existing tools. P.S. designed the label space and ontology integration, and contributed to the tool comparison. H.K. designed the active learning scoring pipeline. D.S. supervised the project. All authors reviewed and edited the manuscript.

References

1. Sugai, L. S. M., Silva, T. S. F., Ribeiro, J. W. & Llusia, D. Terrestrial Passive Acoustic Monitoring: Review and Perspectives. *BioScience* **69**, 15–25, DOI: 10.1093/biosci/biy147 (2019).
2. Cornell Lab of Ornithology. Raven pro (2024).
3. Kahl, S., Wood, C. M., Eibl, M. & Klinck, H. BirdNET: A deep learning solution for avian diversity monitoring. *Ecol. Informatics* **61**, 101236, DOI: 10.1016/j.ecoinf.2021.101236 (2021).
4. Madhusudhana, S. Koogu: A Python package for machine learning in animal bioacoustics (2022).
5. Aide, T. M. *et al.* Real-time bioacoustics monitoring and automated species identification. *PeerJ* **1**, e103, DOI: 10.7717/peerj.103 (2013).
6. Balvanera, S. M. *et al.* Whombat: An open-source audio annotation tool for machine learning assisted bioacoustics. *Methods Ecol. Evol.* **16**, 19–28, DOI: 10.1111/2041-210X.14468 (2025).
7. Dumoulin, V. *et al.* The search for squawk: Agile modeling in bioacoustics. *arXiv preprint arXiv:2505.03071* (2025).
8. Settles, B. Active learning literature survey. Tech. Rep. 1648, University of Wisconsin–Madison (2009).
9. Kath, H., Gouvêa, T. S. & Sonntag, D. A human-in-the-loop tool for annotating passive acoustic monitoring datasets. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence (IJCAI 2023) Demonstrations Track*, 6445–6447, DOI: 10.24963/ijcai.2023/835 (2023).
10. Stretcu, O., Du, J. *et al.* Agile modeling: From concept to classifier in minutes. *arXiv preprint arXiv:2302.12948* (2023).

11. Sitapara, P., Doddanawar, P., Gouvêa, T. S. & Sonntag, D. Structuring annotation label spaces by natural language concept elicitation and ontology grounding. In *Proceedings of the Joint Workshop on Generative AI and Knowledge Graphs and Knowledge Graphs & Agentic Systems Interplay (GenAIK-NORA 2026), co-located with IJCAI-ECAI 2026* (2026). To appear.

12. Sitapara, P., Gouvêa, T. S. & Sonntag, D. Grounded label space engineering for knowledge-centric annotation workflows. In *Proceedings of the Arduous Workshop, co-located with the 49th German Conference on Artificial Intelligence (KI 2026)* (2026). To appear.

13. Saghir, R., Gouvêa, T. S. & Sonntag, D. Visualizing and interacting with model representation space for human-centric active learning. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence (IJCAI 2026) Demonstrations Track* (2026). To appear.

14. Kath, H., Serafini, P. P., Campos, I. B., Gouvêa, T. S. & Sonntag, D. Leveraging transfer learning and active learning for data annotation in passive acoustic monitoring of wildlife. *Ecol. Informatics* **82**, 102710, DOI: 10.1016/j.ecoinf.2024.102710 (2024).

15. Mammadli, N., Gouvêa, T. S. & Sonntag, D. Multi-species mixing for weakly supervised SED under domain shift. In *Proceedings of the 49th German Conference on Artificial Intelligence (KI 2026)* (Springer, 2026). To appear.

16. Mammadli, N. *et al.* Making weak supervision interactive: Exploring transfer from sound libraries to passive acoustic monitoring data. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence (IJCAI 2026) Demonstrations Track* (2026). To appear.

17. Kane, A. pgvector: Open-source vector similarity search for Postgres (2021).

18. GBIF Secretariat. GBIF backbone taxonomy, DOI: 10.15468/39omei (2023).

19. Buttigieg, P. *et al.* The environment ontology: contextualising biological and biomedical entities. *J. Biomed. Semant.* **4**, 43, DOI: 10.1186/2041-1480-4-43 (2013).

20. Buttigieg, P. L. *et al.* The environment ontology in 2016: bridging domains with increased scope, semantic density, and interoperation. *J. Biomed. Semant.* **7**, 57, DOI: 10.1186/s13326-016-0097-6 (2016).

21. Kath, H., Gouvêa, T. S. & Sonntag, D. Novel class identification: A human-in-the-loop approach for efficient species recognition in bioacoustic data. *Methods Ecol. Evol.* (2026). To appear.

22. Schein, A. I., Popescul, A., Ungar, L. H. & Pennock, D. M. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '02, 253–260, DOI: 10.1145/564376.564421 (Association for Computing Machinery, New York, NY, USA, 2002).

23. Ghani, B., Denton, T., Kahl, S. & Klinck, H. Global birdsong embeddings enable superior transfer learning for bioacoustic classification. *Sci. Reports* **13**, 22876, DOI: 10.1038/s41598-023-49989-z (2023).

24. Wang, Y., Li, J. & Metze, F. A Comparison of Five Multiple Instance Learning Pooling Functions for Sound Event Detection with Weak Labeling. *ICASSP 2019 - 2019 IEEE Int. Conf. on Acoust. Speech Signal Process. (ICASSP)* 31–35, DOI: 10.1109/ICASSP.2019.8682847 (2019).

25. He, T. *et al.* An active learning approach with uncertainty, representativeness, and diversity. *The Sci. World J.* **2014**, 827586, DOI: 10.1155/2014/827586 (2014).

26. Celery Project. Celery: Distributed task queue (2009).

27. Wildlife Acoustics. Kaleidoscope pro analysis software.

28. Malkov, Y. A. & Yashunin, D. A. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis Mach. Intell.* **42**, 824–836, DOI: 10.1109/TPAMI.2018.2889473 (2020). Published online December 2018.

29. Ramírez, S. FastAPI (2018).

30. Cañas, J. S. *et al.* A dataset for benchmarking Neotropical anuran calls identification in passive acoustic monitoring. *Sci. Data* **10**, 771, DOI: 10.1038/s41597-023-02666-2 (2023).

S1 Supplementary Information

S1.1 System Architecture

YAPAT follows a modular client–server architecture deployed as a set of Docker services orchestrated with Docker Compose. The **frontend** is a single-page application implemented in TypeScript using React 18, Vite, Tailwind CSS, Ant Design, and Redux Toolkit. Mel spectrograms are computed directly in the browser using a JavaScript mel filterbank, avoiding server-side audio rendering; the embedding space scatter plot is rendered with WebGL-accelerated Plotly.js (`scattergl`). The **backend** exposes a RESTful API implemented in Python with FastAPI and SQLAlchemy, fronted by an nginx reverse proxy with JWT-based authentication. Computationally intensive operations—BirdNET embedding extraction, model inference, retraining, and active learning scoring—are dispatched as asynchronous tasks via Celery with a Redis message broker and result backend, so the annotation interface remains responsive at all times. A dedicated Celery worker handles the PAM active learning queue (retraining and scoring) separately from general processing tasks. All persistent data are stored in a PostgreSQL 15 database extended with pgvector[17] for efficient cosine-similarity retrieval over BirdNET embedding vectors. BirdNET embeddings are extracted via the TensorFlow 2.21 SavedModel API. The complete system is brought up with a single `docker compose up` command; a GPU override file (`docker-compose.gpu.yml`) enables GPU-accelerated embedding and inference without changes to the application code.

The system architecture is illustrated in Fig. 1 (main text).

S1.2 Data Model

The data model is organised around five core concepts.

Teams and users. A *team* is a group of registered users sharing access to one or more datasets. Team members hold one of two roles—*owner* or *user*—and can be invited via time-limited, optionally email-restricted invitation tokens.

Datasets and recordings. A *dataset* is a named collection of audio files located at a server-accessible path, typed as either PAM (field recordings) or focal recordings (sound library clips for WSSED training). Each dataset belongs to a team. Individual audio files are registered as *recordings*; a recording may be of arbitrary length.

Snippets. Each recording is segmented into fixed-length *snippets* (default: 3 s, non-overlapping). Snippets are grouped into a *snippet set* that represents one segmentation configuration; a dataset may have multiple snippet sets (e.g. at different window lengths), with one designated as the default. Snippets carry a start time and end time relative to their parent recording, allowing any annotation to be mapped back to an absolute timestamp in the original file.

Annotations. Each annotation links a snippet to a taxon identifier and a user. Taxon identifiers follow the pattern `namespace:key`—for example, `gbif:2435099` for a GBIF Backbone entry, `envo:<id>` for an ENVO environmental term, or `custom:<uuid>` for a locally defined concept. A resolved name snapshot is stored at annotation time to preserve human-readable labels independently of future changes to external taxonomies. The data model enforces that each (snippet, taxon) pair is annotated at most once per user.

Model checkpoints and predictions. Each retraining cycle produces an *AL model checkpoint* associated with a dataset, storing the path to the saved linear-head weights on disk and a reference to the parent checkpoint from which it was fine-tuned. Per-snippet predictions (label probabilities) and derived acquisition scores (uncertainty, diversity, density) are stored in the database and updated after each retraining cycle. For WSSED, a parallel set of tables records the trained species models, their training jobs, and the segment-level predictions applied to PAM recordings.

The entity relationships are summarised in Fig. S1.

S1.3 Deployment

YAPAT is distributed as a set of Docker images built from a single repository. Deployment requires Docker and Docker Compose; no other dependencies need to be installed on the host. The `HOST_DATA_ROOT` environment variable points to the directory containing audio recordings, which is mounted read-only inside the containers. A separate `HOST_MODELS_AL` directory is mounted read-write to persist model checkpoints across container restarts.

For GPU-accelerated embedding and inference, the override file `docker-compose.gpu.yml` is applied alongside the base compose file. GPU support requires the NVIDIA Container Toolkit on the host; the application selects the first available GPU via TensorFlow device placement (`/GPU:0`) and falls back to CPU automatically if no GPU is detected.

Hardware requirements scale with dataset size. Based on the benchmarks reported in the main text (AnuraSet, $N = 30{,}687$ snippets), a CPU-only deployment on a standard server is sufficient for datasets up to tens of thousands of snippets, with embedding extraction and retraining completing within minutes. GPU acceleration reduces embedding time by approximately $3\times$ and retraining time by 14–$23\times$, and is recommended for larger datasets or faster annotation iteration cycles. The annotation interface and all API endpoints remain responsive regardless of computation time, as all ML jobs are executed asynchronously.

Source code, Docker images, and deployment instructions are available at https://github.com/yapat-app.

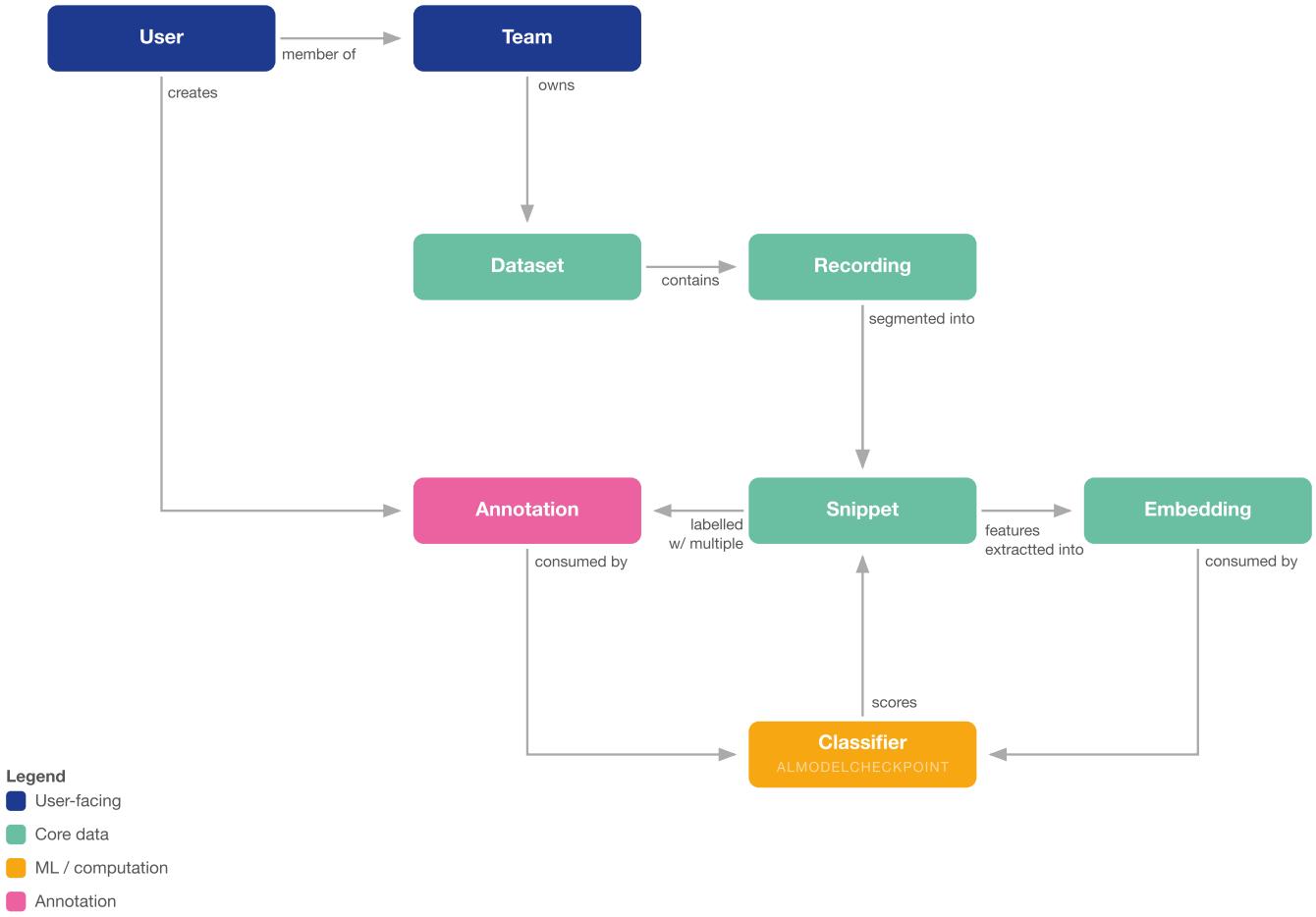


Figure S1. Simplified entity–relationship diagram. Core entities and their relationships; the WSSED species model and associated ML-specific tables (predictions, WSSED jobs) are omitted for clarity.