Technical Documentation: AI-Enhanced eDNA Biodiversity Pipeline

Version: 1.0 (Hackathon Prototype)
Team: Anubhav
Date: September 12, 2025

1. Executive Summary

The vast biodiversity of Earth's deep oceans remains largely uncatalogued, primarily due to the limitations of traditional survey methods and the incompleteness of genetic reference databases. This project addresses a critical challenge posed by the Ministry of Earth Sciences: how to accurately assess biodiversity from environmental DNA (eDNA) when most deep-sea organisms are novel and absent from existing databases. Standard bioinformatics pipelines fail in this scenario, as their rigid reliance on database matching leads to misclassification, underestimation of diversity, and a failure to identify novel species.

We have developed an **AI-Enhanced eDNA Biodiversity Pipeline**, a novel end-to-end software solution that fundamentally inverts the traditional workflow. Instead of searching a database for every single one of the tens of thousands of DNA sequences, our pipeline first uses a custom-trained AI model to **discover** the underlying biological structure within the data itself. This "discover-first, label-second" approach enables us to:

1. **Identify Biologically Relevant Groups:** A Convolutional Neural Network (CNN) converts DNA sequences into image-like fingerprints and groups them into hundreds of distinct clusters without any prior knowledge.
2. **Annotate with High Efficiency:** By annotating only one representative sequence from each of the AI's clusters, we reduce the computational load on traditional tools like BLAST by over 95% while achieving robust taxonomic assignment.
3. **Quantify Novelty:** Our pipeline explicitly flags clusters that have no strong match in reference databases as "Potentially Novel," turning the database gap from a weakness into a powerful tool for discovery.

This prototype has successfully processed a dual-marker (18S and COI) deep-sea dataset, training two specialized AI models and populating a cloud-hosted PostgreSQL database with the final, annotated results. The entire system is modular, automated, and provides a clear framework for a production-ready tool that can accelerate deep-sea biodiversity research for institutions like CMLRE.

2. Problem Context & Background

Environmental DNA (eDNA) analysis is a revolutionary, non-invasive method for biodiversity monitoring. However, its application in deep-sea ecosystems is severely hampered by a critical bottleneck: the incompleteness of genetic reference databases (e.g., SILVA, BOLD).

Deep-sea organisms are vastly underrepresented, meaning standard bioinformatics tools (QIIME2, DADA2) that rely on direct sequence matching often fail. This leads to three primary problems:

- **Underestimation of Diversity:** Novel sequences that don't match the database are often discarded or binned into high-level, uninformative categories (e.g., "Unassigned Eukaryota").
- **Computational Inefficiency:** BLASTing millions of raw reads against massive databases is computationally expensive and slow, creating a significant bottleneck for routine monitoring.
- **Missed Discoveries:** The primary goal of deep-sea exploration—finding novel life—is inherently limited by a methodology that can only identify what is already known.

Our project was designed to solve this exact problem by reducing database dependency and using AI to extract biological meaning directly from the sequence data itself.

3. Solution Overview & Architecture

Our solution is a modular, end-to-end pipeline that transforms raw sequence data into a queryable, backend-ready database. The core innovation is the **"Discover-First, Label-Second"** workflow.

Architectural Components:

1. **Phase 1: Pre-processing (DADA2):** Standardizes and cleans raw FASTQ reads into high-fidelity Amplicon Sequence Variants (ASVs). This ensures the input to the AI is of the highest possible quality.
2. **Phase 2: AI-Powered Clustering (CNN + HDBSCAN):** This is the heart of our innovation.
    - **FCGR Conversion:** Each ASV is converted into a 2D "genomic fingerprint" image.
    - **CNN Autoencoder:** A GPU-accelerated neural network learns the structural features of these images, compressing each one into a meaningful 32-dimensional numerical vector.
    - **HDBSCAN Clustering:** A density-based algorithm groups these vectors, identifying clusters of related organisms and flagging unique sequences as "noise."
3. **Phase 3: Hybrid Annotation (BLAST):** Instead of blasting all ASVs, we only blast the most abundant representative from each AI-discovered cluster. The resulting taxonomy is propagated to all members of its cluster, dramatically improving efficiency.
4. **Phase 4: Database Population:** All processed data—sample metadata, AI cluster assignments, taxonomic annotations, and relative abundances—are loaded into a structured PostgreSQL database.
5. **Phase 5: Frontend Visualization:** A web-based dashboard (built with Streamlit/React) connects to the database via a backend API (Spring Boot) to provide an interactive user interface for data exploration.

4. Detailed Methods

4.1. ASV Generation (DADA2)

The DADA2 pipeline (v1.24.0) was used to process raw paired-end reads. The key steps, implemented in a resumable R script, were:

1. Quality filtering and trimming.
2. Error rate learning.
3. Sequence inference (denoising).
4. Paired-end merging.
5. Chimera removal (removeBimeraDenovo).
   The final outputs were a sequence-by-sample abundance table (seqtab.tsv) and a FASTA file of all unique ASVs (asv_sequences.fasta).

4.2. AI Clustering Workflow

FCGR Generation:
Each DNA sequence was converted to a 2D numerical array using a Frequency Chaos Game Representation (FCGR) algorithm with a k-mer size of 7, resulting in 128x128 pixel grayscale images.
CNN Architecture & Training:
We implemented a Convolutional Autoencoder in PyTorch. The goal of an autoencoder is to learn a compressed representation (latent vector) of the input by trying to reconstruct it.

- **Encoder:** Consists of three Conv2d layers that progressively downsample the image, followed by a fully connected layer that outputs a **32-dimensional latent vector**.
- **Decoder:** Mirrors the encoder architecture using ConvTranspose2d layers to reconstruct the original 128x128 image from the latent vector.
- Hyperparameters:

Parameter	Value	Justification
KMER_SIZE	7	Good balance of detail and VRAM usage for a 4GB GPU.
LATENT_DIM	32	Standard size for robust feature representation.
EPOCHS	20	Sufficient for loss to converge on this dataset.
BATCH_SIZE	16	Safe choice for a 4GB GPU to prevent memory errors.
Optimizer	Adam (lr=1e-3)	Standard, effective optimizer.
Loss Fn	Mean Squared Error	Measures reconstruction quality.

Clustering:
The 32-dimensional latent vectors were clustered using the HDBSCAN algorithm (min_cluster_size=5). HDBSCAN was chosen for its ability to discover clusters of varying shapes and densities without requiring the number of clusters to be specified beforehand, and for its explicit identification of noise points.

4.3. Hybrid Annotation Workflow

1. **Representative Selection:** For each AI-discovered cluster, the ASV with the highest

total abundance across all samples was selected as its representative.

2. **Database Preparation:** The DADA2-formatted PR2 (18S) and MIDORI2 (COI) databases were reformatted to be compatible with BLAST+ using a custom script to shorten FASTA headers. The makeblastdb command was then used to create indexed databases.
3. **BLAST Execution:** The representative sequences were searched against the appropriate BLAST database using blastn with a strict e-value of 1e-5 and max_target_seqs=1.
4. **Annotation & Confidence Scoring:** The taxonomic lineage from the best BLAST hit was assigned to all ASVs within that cluster. A confidence_level was assigned based on the percent identity of the BLAST hit:
    - **High:** >= 97% identity.
    - **Potentially Novel:** < 97% identity.
    - **Noise:** Applied to all unclustered ASVs from HDBSCAN.

5. Results & Proof-of-Concept

Our pipeline was successfully executed on both the 18S and COI datasets.

Key Performance Metrics:

Marker	**Total ASVs**	**AI-Discovered Clusters**	**Unclustered (Noise) ASVs**
18S	141	15	18
COI	17,255	692	9,756

Computational Efficiency (Simulated Example): The hybrid annotation strategy provides a dramatic reduction in computational time.

Task	**Traditional (BLAST all)**	**Our Hybrid Approach**	**Improvement**
Sequences to BLAST (COI)	17,255	692	**96% Reduction**
Est. Runtime (CPU hours)	~24 hours	~1 hour	**24x Faster**

Novelty Detection: The pipeline successfully identified numerous clusters with low identity to the reference databases, particularly in the COI dataset, flagging them as "Potentially Novel." This provides a direct, data-driven list of candidates for further scientific investigation, fulfilling a core requirement of the problem statement.

6. Reproducibility

The entire pipeline is reproducible using the provided conda environment, scripts, and a cloud-hosted PostgreSQL database. The key steps are:

1. Set up the conda environment using the provided command.
2. Run the DADA2 R script to generate ASVs.
3. Run the run_ai_clustering.py script for each marker.
4. Run the run_hybrid_annotation.py script for each marker.
5. Run the populate_database.py script to load the results.
6. Launch the app.py Streamlit dashboard to visualize the live data.

A full runbook with a Dockerfile is provided in a separate document.

7. Limitations and Future Roadmap

Limitations:

- The prototype does not yet implement the planned machine learning-based bias correction (Phase 4), which requires a separate mock community dataset.
- The epa-ng phylogenetic placement for novel clusters was not implemented in this prototype but is a clear next step.

6-Month Roadmap (MVP):

1. **Backend Finalization:** Complete the Spring Boot API to serve all necessary data for the frontend.
2. **Frontend Development:** Build out the full React + D3.js dashboard with all planned interactive components.
3. **Implement Bias Correction:** Integrate the XGBoost model for more accurate abundance estimations.

1-Year Roadmap (Production):

1. **Scalability:** Deploy the pipeline on a scalable cloud architecture (e.g., AWS Batch, Kubernetes) to handle hundreds of samples.
2. **User Authentication:** Add user accounts and project management features.
3. **Integrate Phylogenetic Placement:** Fully integrate epa-ng to provide interactive visualizations of novel taxa on the tree of life.

Appendix: Missing Inputs & Verification

- **Verified from Chat History:**
  - Successful execution of DADA2 pipeline for 18S and COI.
  - Successful training of two distinct CNN models on 18S and COI data.
  - Successful annotation and database population.
  - Generation of all key data artifacts (seqtab.tsv, cluster_map.csv, annotated_clusters.csv).
- **Could Not Verify (Not Provided):**

- **Mock Community Data:** This was required for Phase 4 (Bias Correction). The implementation in the technical documentation remains a conceptual plan.
- **Reference Tree for epa-ng:** A curated reference tree is needed for phylogenetic placement.
- **Primer Sequences:** The exact primer sequences used in the cutadapt step were not provided and are based on common examples.