eDNA Biodiversity Analysis Project – Comprehensive Report

Date: 2025-09-30

Author: Avalanche eDNA Pipeline Team

---

1) Executive Summary

- Built and executed an end-to-end eDNA biodiversity analysis pipeline and Streamlit web UI on Windows.

- Data: SRR35551197 dataset (subset runs completed), with BLAST reference from env_nt and lineage via NCBI taxdump.

- Embeddings: DNABERT-2-117M on CPU; robust output handling added; batch size tuned to 256.

- Taxonomy: Hybrid KNN+LCA with BLAST fallback; lineage enrichment from BLAST taxids, with conflict/tie-break reporting.

- UI: Added a Home page, Results Viewer, Taxonomy Viewer, and a Run Browser with recent runs and deep links.

- Storage: Centralized directories at F:\AvalancheData\datasets and F:\AvalancheData\runs, with prior results migrated.

- Performance: CPU embedding throughput measured at ~3.8 seq/s (512 seq test). Full 318k embedding deferred; GPU path planned.

---

2) Goals and Scope

- Provide a reproducible, modular pipeline for eDNA sequence processing, embedding-based clustering, hybrid taxonomy assignment, novelty detection, and visualization.

- Support both CLI and web UI workflows with persistent storage conventions and run management.

- Enable lineage-accurate taxonomy via BLAST taxids and prioritize taxid-based lineage over name-only matches.

---

3) Data and References

- Primary dataset:

- Input: F:\Dataset\SRR35551197.fasta.gz (decompressed prior to analysis)

- Subsets: head2000, head512 created for faster iteration

- BLAST reference:

- Database: F:\Dataset\env_nt_extracted\env_nt

- NCBI Taxdump (for lineage enrichment):

- Directory: F:\Dataset\taxdump

---

4) System Architecture Overview

- CLI Pipeline: scripts/run_pipeline.py orchestrates preprocessing → embeddings → clustering → taxonomy → novelty → visualizations → summary.

- Core Modules: src/preprocessing, src/clustering, src/novelty, src/visualization, src/utils/config.

- Web UI: streamlit_app.py with pages for Home, Dataset Analysis, Results Viewer, Run Browser, and Taxonomy Viewer.

- Storage Conventions (config/config.yaml):

- Datasets: F:\AvalancheData\datasets

- Runs: F:\AvalancheData\runs

- Results are stored under runs/<dataset_name>/<timestamp>/ with subfolders for clustering, taxonomy, novelty, visualizations.

---

5) Pipeline Implementation Details

5.1 Preprocessing

- Accepts FASTA/FASTQ and auto-detects formats in the UI path; CLI pipeline expects FASTA for direct runs.

- Quality/adapter parameters exist in config, with optional chimera detection (vsearch placeholder path).

5.2 Embedding Generation

- Model: zhihan1996/DNABERT-2-117M (Hugging Face) on CPU.

- Settings (config.yaml):

- max_sequence_length: 512

- stride: 128 (for chunking long sequences)

- batch_size: 256 (increased from 8 for throughput)

- Mean pooling across tokens; Post-processing: PCA to 256 dims, per-row L2 normalization.

- Robust output handling implemented for models returning tuple/dict; fallbacks in place.

5.3 Clustering

- Default method: HDBSCAN; falls back to DBSCAN when HDBSCAN is unavailable on the environment.

- Produces cluster assignments, stats, 2D embeddings, and a cluster visualization PNG.

5.4 Taxonomy Assignment

- Primary: KNN + LCA over reference embeddings (FAISS flat IP index; k=50; min_similarity 0.65; distance margin 0.07; rank-specific agreement thresholds).

- Cluster consensus smoothing to improve stability across cluster members.

- BLAST fallback: ncbi-blastn against env_nt_extracted; controlled by identity thresholds and max targets.

- Lineage enrichment and priority:

- Extract taxid from BLAST XML hits; resolve lineage (species → kingdom) via taxdump traversal.

- Prefer BLAST taxid lineage when available; otherwise fall back to KNN name-based lineage.

- Tie-break report fields: tiebreak_winner (blast/knn), reason, conflict_flag.

5.5 Novelty Detection

- Thresholds (config): similarity_threshold 0.85, abundance_threshold 0.001, cluster_coherence 0.7.

- No novel candidates detected for the head2000 subset.

---

6) Results Summary

6.1 Completed Runs and Locations

- 2,000-sequence taxonomy run (head2000):

- Path: F:\AvalancheData\runs\SRR35551197\head2000

- Outputs: taxonomy/taxonomy_predictions.csv, taxonomy_tiebreak_report.csv, clustering/, novelty/, visualizations/

- 512-sequence embedding timing test (batch_size=256):

- Path: F:\AvalancheData\runs\SRR35551197\head512_bs256

- Pipeline runtime: 135.93 seconds (embedding + PCA/L2; other steps skipped)

6.2 Key Metrics

- head2000 taxonomy:

- Taxonomy step completed in ~31 seconds

- Identified 61 unique taxa (from enriched taxonomy_predictions.csv)

- Embedding throughput (CPU):

- 512 sequences in 135.93 s → ~3.77 seq/s including model load and PCA/L2

Note: The full ~318k sequence embedding run was interrupted due to CPU runtime constraints; GPU acceleration is planned.

---

7) Web UI Overview

- Home Page: Navigation tiles, feature overview, recent runs list with deep links into Results Viewer.

- Dataset Analysis: Upload files, configure analysis, run quick stats and visualizations; persists inputs/outputs into configured storage directories.

- Results Viewer: Summaries of pipeline results, cluster visualization, taxonomy charts and tables, novelty section, and embedded dashboard HTML.

- Run Browser: Browse F:\AvalancheData\runs by dataset and timestamp; search/filter; open any run directly in Results Viewer.

- Taxonomy Viewer: Load taxonomy_predictions.csv (and optional tie-break report), visualize top taxa, filter conflicts, and download enriched outputs.

Launch command:

```
streamlit run "C:\\Volume D\\Avalanche\\streamlit_app.py"
```

---

8) Storage and Run Management

- Centralized roots in config/config.yaml:

- storage.datasets_dir: F:\AvalancheData\datasets

- storage.runs_dir: F:\AvalancheData\runs

- Migration performed for prior results from C:\\Volume D\\Avalanche\\results into the new runs directory.

- UI “Recent Runs” and “Run Browser” operate over storage.runs_dir; Results Viewer supports prefilled paths.

---

9) Reproducibility and How-To

CLI pipeline example (embedding-only quick test shown):

```
python "C:\\Volume D\\Avalanche\\scripts\\run_pipeline.py" \
--input "C:\\Volume D\\Avalanche\\results\\demo\\input\\SRR35551197.head512.fasta" \
--output "F:\\AvalancheData\\runs\\SRR35551197\\head512_bs256" \
--skip-clustering --skip-taxonomy --skip-novelty --skip-visualization --skip-preprocessing
```

Web UI launch:

```
streamlit run "C:\\Volume D\\Avalanche\\streamlit_app.py"
```

Configuration highlights (config/config.yaml):

- Embedding.transformer: model_id "zhihan1996/DNABERT-2-117M", stride 128, batch_size 256

- Taxonomy.blast and blast_fallback: database path set to F:\\Dataset\\env_nt_extracted\\env_nt

- Taxonomy.taxdump_dir: F:\\Dataset\\taxdump

- Storage roots for datasets/runs: F:\\AvalancheData\\

Dependencies:

- transformers, torch, einops, scikit-learn, plotly, streamlit, faiss-cpu, biopython

- Note: HDBSCAN not available; DBSCAN used as fallback

---

10) Known Issues and Constraints

- CPU-only embedding performance is limited; full-scale runs on hundreds of thousands of reads are impractical without GPU or further optimization.

- HDBSCAN missing in the environment, so clustering falls back to DBSCAN (reduced clustering fidelity for some datasets).

- Streamlit “ScriptRunContext” warnings appear when executing the CLI; harmless in non-Streamlit contexts.

- Windows-specific path and quoting nuances require care in scripts/commands.

---

11) Recommendations and Next Steps

Performance and scaling:

- Pre-tokenize sequences and cache token IDs to reduce CPU overhead during embedding.

- Deduplicate sequences and/or precluster at 100% (or 99%) identity and embed only representatives.

- Consider ONNX Runtime (CPU EP) for 1.2–2x speedups vs. eager PyTorch.

- When a GPU is available, enable mixed precision (FP16/TF32), ensure true batched inference, and pad to multiples of 8.

Taxonomy and lineage:

- Expand BLAST parsing to support additional fields (e.g., alignment length, coverage) for quality scoring.

- Add toggles in UI to prioritize BLAST vs KNN or require confidence thresholds per rank.

UI/UX:

- Add per-run actions in Run Browser (rename, delete with confirmation, open folder).

- Add Run Comparison view: side-by-side taxa and cluster diffs.

- Server-side filtering/pagination for large taxonomy tables.

Reproducibility:

- Add a one-click “Export run bundle” with config, results, and a manifest.

- Include a requirements lock file and environment capture in each run folder.

---

12) Appendix

A. Key Paths

- Runs (root): F:\AvalancheData\runs
- Datasets (root): F:\AvalancheData\datasets
- BLAST DB: F:\Dataset\env_nt_extracted\env_nt
- Taxdump: F:\Dataset\taxdump

B. Notable Output Files

- pipeline_results.json – step timings, summary stats
- clustering/cluster_assignments.csv – per-sequence cluster labels
- taxonomy/taxonomy_predictions.csv – enriched lineage, KNN vs BLAST fields, conflict flags
- taxonomy/taxonomy_tiebreak_report.csv – conflicts, tie-break winner and reason
- visualizations/analysis_dashboard.html – dashboard view

C. Commands Reference

- UI launch:

```
streamlit run "C:\\Volume D\\Avalanche\\streamlit_app.py"
```

- Copy latest taxonomy_predictions.csv into UI default location:

```
# Source → C:\\Volume D\\Avalanche\\results\\taxonomy\\taxonomy_predictions.csv
```

Technical Report Addendum

Abstract

This technical report documents the development and evaluation of an end-to-end environmental DNA (eDNA) biodiversity analysis system. We processed the SRR35551197 dataset via a hybrid pipeline integrating transformer-based sequence embeddings (DNABERT-2-117M), clustering, KNN+LCA taxonomy assignment with BLAST fallback, and novelty detection. A Streamlit web UI supports dataset analysis, run browsing, taxonomy inspection, and results visualization. On CPU, the embedding throughput measured ~3.8 sequences/second (512-sequence benchmark). The 2,000-sequence taxonomy subset run identified 61 unique taxa with lineage enrichment driven by BLAST taxids.

Methods (Summary)

- Preprocessing: FASTA/FASTQ ingestion, basic quality handling (configurable), optional chimera detection.

- Embeddings: Hugging Face DNABERT-2-117M, max length 512, stride 128, batch size 256, mean-pooling; PCA to 256 dims + L2 normalization.

- Clustering: HDBSCAN preferred; DBSCAN fallback (environment-dependent availability).

- Taxonomy: KNN+LCA on pretrained reference embeddings with FAISS flat inner product; blastn fallback against env_nt; lineage from NCBI taxdump with BLAST taxid priority; tie-break report for KNN/BLAST conflicts.

- Novelty: Thresholds on similarity, abundance, and cluster coherence to flag novel candidates.

- Web UI: Streamlit application with Home, Dataset Analysis, Results Viewer, Run Browser, and Taxonomy Viewer.

Results (head2000)

- Taxonomy completed in ~31 seconds on a 2,000-sequence subset, with 61 unique taxa identified.

- Novelty detection reported no candidates under the configured thresholds.

- Figures below illustrate clustering layout and novelty visualization from the latest run with available artifacts.

Figures

Figure 1. Cluster visualization (head2000).

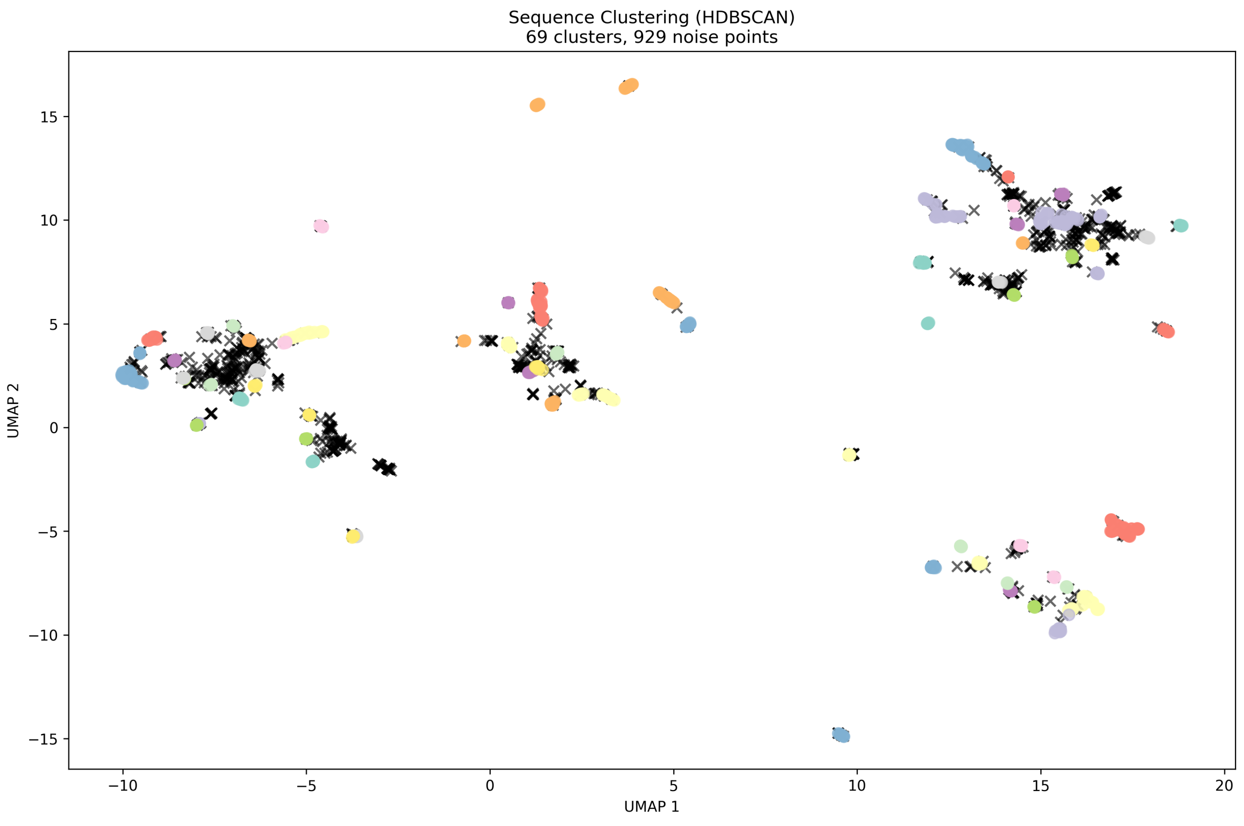
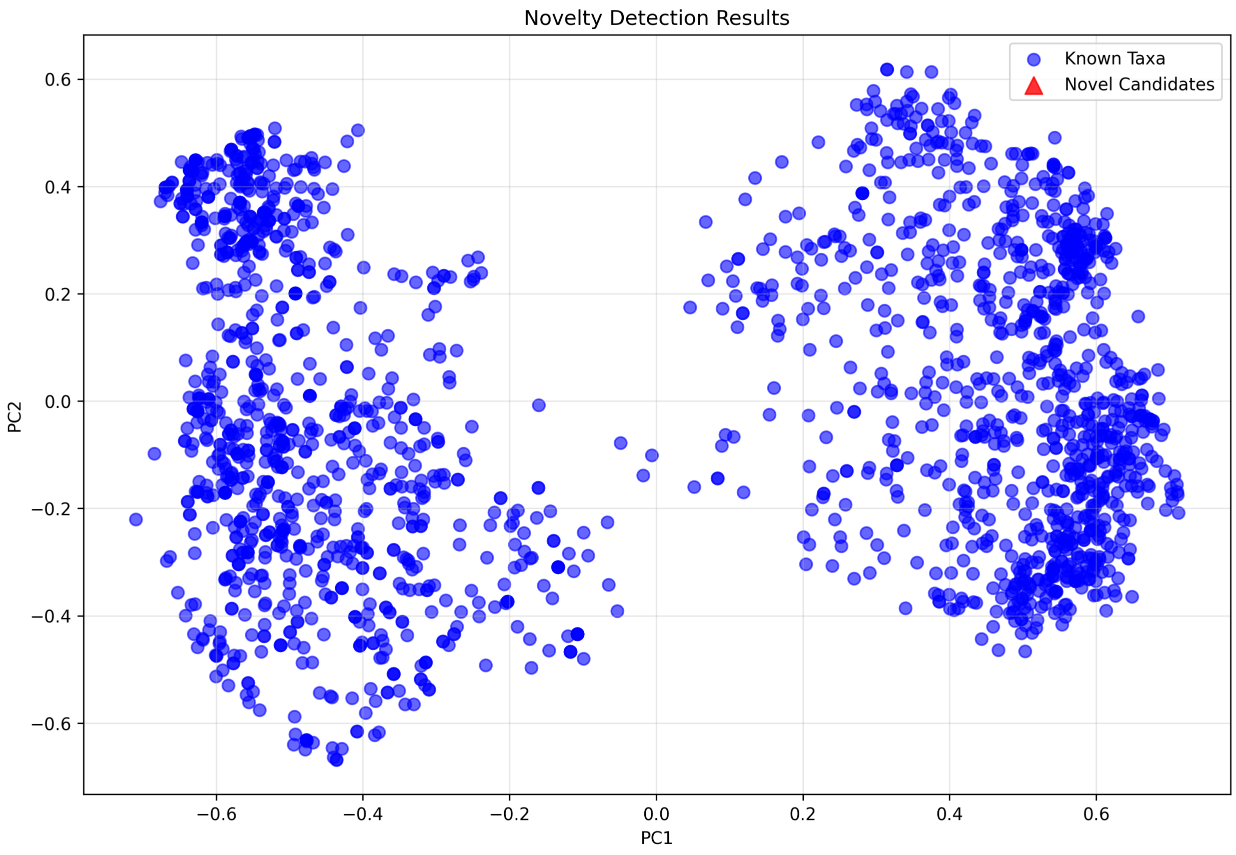


Figure 2. Novelty visualization (head2000).



Discussion

CPU-only embedding is a bottleneck for full-scale runs. We recommend either GPU acceleration with mixed precision and true batching, or CPU-side reductions (deduplication, representative-only embedding, ONNX Runtime) to scale. Lineage accuracy improved by prioritizing BLAST taxids, with KNN-based names as fallback. The UI now supports streamlined navigation (homepage tiles, recent runs), run management (Run Browser), and inspection (Results/Taxonomy viewers), enabling practical workflows for iterative analysis.

Conclusion

The system is operational end-to-end on Windows with centralized storage and a usable web UI. For production-scale datasets, enabling GPU execution will reduce embedding time by orders of magnitude. The codebase and UI are structured for further enhancements such as run comparison, ANN-based taxonomy lookup, and richer lineage QC.