INFO -698
Capstone Project[May -25]

Dr Bryan Heidorn, Dr Greg Chism

Eco Mine: A Data Discovery Tool Intelligent Metadata Extraction from Ecological Archives

Project Final Report

Prepared by
Vik Ruhil, Saksham Gupta

Eco Mine: A Data Discovery Tool Intelligent Metadata Extraction from Ecological Archives

Abstract

The rapid growth of ecological research publications has created a need for automated systems to extract and organize information from scientific literature. *EcoMine* is a capstone project that addresses this challenge by mining ecological journal articles and linking them to real-world biological field stations. The system processes a dataset of approximately **50,000** journal records (sourced via JSTOR's Constellate platform) and extracts key metadata (title, authors, publication year) and content from each article. Advanced natural language processing (NLP) techniques, including named entity recognition (NER), are applied to identify research locations and field station names mentioned in the text. These locations are then linked to curated datasets of field stations (e.g., OBFS and NEON) to enrich the literature metadata.[1,2]

The extracted information is stored in a hybrid relational database that combines structured bibliographic data with NLP-derived content, enabling efficient retrieval and analysis. The EcoMine platform supports semantic search, automated text summarization, and data visualization through an interactive interface, demonstrating a novel integration of AI-driven text mining with ecological data science. This report presents the system architecture, workflow, design choices, challenges faced, dataset details, illustrative examples of system output, and future enhancements. The results showcase EcoMine's potential to bridge the gap between unstructured ecological literature and structured knowledge bases, facilitating faster discovery of patterns and insights in environmental research.

Introduction

Ecology as a field has witnessed an explosion in the volume of published research, spanning decades of studies across diverse ecosystems and species. Much of this knowledge remains **buried in PDFs, prose, and static text**, making it difficult to search, summarize, or analyze at scale. Researchers and conservationists often struggle to discover patterns or compare methods across studies because traditional literature reviews are time-consuming and manual. Recent work in the scientific community suggests that **text mining and NLP** can significantly improve the efficiency and reproducibility of literature analysis in biology[3,4]. However, unlike fields such as biomedicine where NLP is widely adopted, ecology has lagged in leveraging these tools[5]. This gap motivates the development of specialized systems that can automatically extract and structure knowledge from ecological publications.

Another barrier in ecological informatics is the disconnect between published research and the **biological field stations** or observatories where studies are conducted. Biological field stations and marine labs serve as *"living libraries and outdoor laboratories"* for environmental research and education[6]. They are often the sites of long-term ecological data collection, yet linking academic papers to specific field stations (e.g., a particular research forest or marine reserve) is not straightforward. **The Organization of Biological Field Stations (OBFS)** and the **National Ecological Observatory Network (NEON)** maintain lists of field research sites worldwide, but prior to this work there was no automated system to connect literature references to these field site databases. As a result, valuable contextual information—such as which studies were done at which locations or how research at one station compares to another—remains underutilized.

EcoMine was created to bridge these gaps by using AI to transform unstructured ecological literature into a structured, queryable knowledge base. Specifically, this project automates the **discovery, extraction, enrichment, and organization** of ecological journal articles[1]. It ingests metadata and full-text content from thousands of publications and uses NLP (including NER and entity linking) to identify mentions of species, locations, and field stations within the text. These mentions are then cross-referenced with authoritative lists of field stations (from OBFS, NEON, etc.) to accurately map each study to real geographic sites . The enriched information is stored in a three-tier relational database[2] that captures bibliographic details, NLP-derived insights (like key phrases and entities), and relationships to field stations. By combining **text mining** with **geospatial linkage**, EcoMine enables researchers to query and analyze ecological literature in novel ways-for example, finding all studies conducted in desert research stations, or visualizing collaboration networks by location.

The significance of this platform lies in its ability to **accelerate literature review and meta-analysis** in ecology. Instead of manually combing through hundreds of PDFs, scientists can leverage EcoMine's database and interface to quickly search for relevant studies by topic, location, or time period. This not only saves time but also uncovers hidden connections (such as multiple independent studies at the same field site) that might be missed otherwise. Moreover, by structuring the knowledge in a database, EcoMine lays the groundwork for advanced analytical tools, such as trend analysis over time, network analysis of co- authorship or site collaborations, and even retrieval-augmented question answering about ecological findings.

In the following sections, we detail the **System Architecture** of EcoMine, describing its modular design and components. We then outline the end-to-end **Workflow**, from data ingestion to information extraction and visualization. The report discusses key **Design Choices and Tech Stack**, highlighting why certain technologies (Python, spaCy, SQLite, etc.) were chosen. We present the main **Challenges** encountered (such as data acquisition limits and NLP complexities) and the strategies employed to mitigate them. A description of the **Dataset** used (journal corpus and field station lists) is included, followed by **Examples** of the system's output (e.g., visualizations of the mined data) to illustrate EcoMine's functionality. We conclude with a summary of results in the Conclusion, and identify opportunities for **Future Work**, such as integrating domain-specific language models and expanding the platform's capabilities. Finally, a list of **References** is provided, reflecting sources and literature that informed this project.

System Architecture

EcoMine's architecture is designed in a modular, extensible fashion with a clear separation of concerns between data ingestion, processing, storage, and user interaction. **Figure 1 (conceptual)** illustrates the high-level architecture, which can be viewed as a four-tier pipeline**: (1) Data Collection, (2) NLP Processing,(3) Knowledge Base Structuring, and (4) User Interface**. Each tier corresponds to specific modules in the implementation, ensuring that the system is organized and maintainable.

- **Data Collection (Ingestion)**: The pipeline begins with ingesting raw publication data. The source of this data is JSTOR's **Constellate** service – a platform that provided bulk access to academic journal content. EcoMine uses a snapshot dataset of ~50k ecological journal articles obtained from Constellate(covering historical and contemporary publications; seeDatasetsection).Custom ingestion scripts *ingest_journals.p* parse the raw data (originally in JSONL format) and populate an initial database with structured metadata for each article. This includes bibliographic fields such as article title, authors, journal name, publication date, and potentially abstracts or full text[1,7]. The ingestion process also involves cleaning and normalizing the data (e.g., converting text to a uniform case, handling special characters) to prepare it for analysis.

- **NLP Processing (Text Mining)**: In the second tier, the system applies natural language processing to the content of each article to extract deeper insights. This involves using **spaCy 3.8** pipelines for tasks like tokenization, part-of-speech tagging, and named entity recognition. A custom NLP[8] component scans the full text of articles for mentions of **field stations** or other location entities. Detected entity names (e.g., specific forest reserves, research centers, national parks) are then matched against the known field station list using fuzzy string matching (via the RapidFuzz library) to account for variations or aliases in naming. Additionally, the NLP tier performs **keyphrase extraction** – identifying frequently occurring scientific terms (e.g., "species", "habitat", "biodiversity") and generating **n-grams** that characterize each article's content. These tasks are facilitated by scripts like *extracting_fieldstationNLP.p* (for entity extraction) and *generate_ngrams.p*[9] The output of this processing (such as a list of identified field station references for each article, and key topic phrases) is then added to the data store. This tier is designed to be extensible: for example, more advanced models like **SciBERT or BioBERT** could be integrated here to improve recognition of domain-specific entities . In future upgrades[10], one could also incorporate a summarization model (e.g, T5) to auto summarize each article's findings, given the modular nature of this NLP pipeline

- **Knowledge Base Structuring (Database):** The third tier is the **hybrid database** that serves as EcoMine's knowledge base. We chose an **SQLite** relational database (accessed via SQLAlchemy ORM in Python) for simplicity and portability during development. The database uses a **three-tier schema design**:

  - **Tier 1**: A *field_stations* table containing curated information about field sites (including station name, geographic location, coordinates, ecosystem type, etc.[12]).This table is pre-populated from the OBFS and NEOON CSV datasets (see Dataset Section) and acts as a reference catalog of valid field stations.
  - **Tier 2**: A *journals* table containing the core metadata of each ingested article – such as title, authors, journal, year, and possibly a snippet of the text or keywords. Each entry in *journals* may also include a pointer or foreign key to a field station (or multiple stations) if the article is identified to be linked with those locations. In addition, we utilize a JSON field (*meta_json*) within this table to store semi-structured data like the extracted keyphrases or counts of certain entities[13]. This hybrid approach (relational + JSON) provides flexibility: structured fields allow efficient querying for known attributes, while the JSON field can hold variable NLP outputs without altering the schema.
  - **Tier 3**: A *journal_nl* (or analogous) table that holds the NLP-enriched data for each article, such as lists of n-grams, named entities (people, species, locations), topics or embeddings (if generated). In the current implementation, some of this NLP data is stored in the *meta_json* of the journals table itself for convenience[14], but the design anticipates a dedicated table as the volume of NLP annotations grows (for example, if storing large embedding vectors or extensive lists of entities). This tier links back to Tier 2 via article IDs, ensuring each piece of derived data is traceable to its source publication.

The database is thus a **normalized, multi-tier representation of knowledge**, where each publication record can be joined with field station records and NLP-generated insights. By structuring the data in this way, we can query complex relationships—for instance, retrieving all publications from a certain year that mention a particular field station, or all field stations that appear in studies about "climate change" (assuming that topic is captured in keyphrases).

- **User Interface & Visualization:** The top layer of the architecture is the user-facing component. While a full web application was outside the initial scope, the project provides multiple means of interacting with the data. A suite of **Jupyter Notebooks** (*notebooks/*) allows researchers to run exploratory queries and generate visualizations from the database. Additionally, a lightweight command-line interface and plotting scripts (e.g.,*visualize_journals.p*) produce charts and graphs to summarize the data[15,16]. The design is compatible with building a future **dashboard** (for example, using Streamlit or Dash) that could enable real-time search and filtering of the literature via a web app. In concept, such an interface would let users query the database by keywords or location and get dynamic results, possibly even employing a language model to answer natural language questions by retrieving relevant articles (a retrieval-augmented generation approach). For now, the interface consists of static and interactive Python-based analysis, but the architecture's modular API layer (*data-discovery-tool/api/)* hints at how it could evolve into a more robust application.

The overall system architecture emphasizes **modularity and scalability**. Each component (ingestion, NLP processing, database, visualization) can be improved or replaced independently. For example, the SQLite database could be migrated to a more scalable DBMS or a graph database if needed, and the NLP pipeline can incorporate new models or additional text-mining algorithms (such as topic modeling or semantic similarity search) without altering the rest of the system. This layered design proved effective during development, as team members could work in parallel on different modules (e.g., one focusing on data cleaning and ingestion while another fine-tuned the NER pipeline) with well-defined integration points. In summary, EcoMine's architecture provides a **robust framework for ecological text mining**, combining the reliability of a structured relational schema with the flexibility of modern NLP and paving the way for interactive knowledge discovery tools.

Workflow

EcoMine's end-to-end workflow follows a sequence of stages aligning with the architecture described above. From data acquisition to insight generation, the process can be summarized in four main steps:

1.**Ingestion of Journal Metadata**: The workflow begins by obtaining the ecological literature data from a **JSTOR Constellate** dataset. Given that Constellate's bulk data service is being sunset in 2025[7] , we relied on a previously downloaded snapshot (~50,000 records) for this project. These records (in JSON Lines format) contain article metadata and often full text or abstracts. Using the script *ingest_journals.py*, the team imported this data into the EcoMine SQLite database. During ingestion, each JSON record is parsed: fields like title, authors, publication year, and journal name are mapped to the database schema, and the article's text is stored for NLP processing. Basic cleaning operations (trimming whitespace, standardizing date formats, etc.) are applied. This step results in a populated *journals* table with one entry per article. It is a critical foundational step, as it transforms raw JSON data dumps into a queryable form. If any article in the input lacks required metadata or has malformed content, it is flagged and handled (either corrected if possible or skipped) to ensure database integrity.

2.**NLP Extraction and Enrichment**: Once the database is seeded with the raw metadata, the next step is **running the NLP pipeline** to enrich these records with additional information. We activate the **field station** extraction process via *extracting_fieldstationNLP.p* (a Python script in the ***ml/preprocessing*** directory). This script loads each article's text (from the database or original files) and uses the spaCy NLP model to identify named entities. We specifically focus on entities of type GPE (geo- political entities), LOC (locations), and custom-defined entities for known field station names. The script cross-references any identified entity against the list of ~300 field stations compiled from OBFS and NEON. For example, if an article mentions "Harvard Forest", the NLP finds this proper noun; then a fuzzy matching algorithm compares it to the field_stations table and finds a close match (Harvard Forest is indeed an OBFS member station in Massachusetts). The article's record is then updated: a link is made indicating that **FieldStationID=HarvardForest** is associated with that article. In cases where no field station is mentioned explicitly, no link is added (but the article remains in the database for other analyses). In parallel, another enrichment occurs: **keyphrase extraction**. We generate a set of representative keywords or bi-grams for each article, either by extracting author-provided keywords (if available in the metadata) or by algorithmically selecting frequent significant terms from the text (excluding common stopwords). These keyphrases provide a quick overview of the article's content (e.g., "climate change", "tropical forest", "species richness"). The results of this NLP step are stored back into the database, primarily in the *meta_json* field of the journals table or a related NLP table. After running this step for all records, each article entry is enriched with zero or more field station links and a collection of key terms. The repository's documentation outlines this multi-stage process clearly .

3.Data Verification and Quality Assurance: Before proceeding to analysis, it's important to ensure the data's consistency and completeness. The EcoMine workflow includes validation scripts such as *db_sanity_check.py* and *verify_jourals.p*. These are executed to perform checks like: - Ensuring every article has a minimum set of metadata (title, year, etc.) and flagging any missing entries. - Verifying referential integrity, e.g., that any field station IDs linked to articles actually exist in the field_stations table. - Checking that counts of keyphrases or entities fall within expected ranges (for instance, an article shouldn't have an exorbitant number of entities unless it's extremely long; if it does, it might indicate a parsing issue). - Spot-checking the output of NLP: for a random sample of articles, we manually compare the detected field station names to the text to confirm accuracy. Any anomalies discovered are addressed in this stage. For example, if *verify_keyphrases.p* finds that some articles have no keyphrases at all (which might happen if the text was empty or not processed), those records can be re-processed or marked for exclusion from certain analyses. This verification step acts as a **feedback loop**, improving the dataset quality. In some cases, adjustments to earlier steps are made – e.g., if we notice the NLP is mistakenly tagging common words as entities, we might update the pipeline (like adding a stoplist or adjusting the entity recognition model). After this stage, we have a refined database that is ready for exploration.

4.**Exploration and Visualization**: With a verified, enriched database in hand, the final step is analysis and visualization of the information. The project provides a Jupyter Notebook (*notebooks exploratory_analysis.ipynb*) that researchers can use to run custom queries and generate plots. Additionally, executing the *visualize_journals.py* script produces a set of predefined visualizations that summarize key aspects of the data [20 21] . For instance, one can automatically create histograms of the number of authors per publication, word clouds of frequent key phrases, or time-series plots of publication counts by year. These visual outputs help validate the success of previous steps (for example, a word cloud showing expected ecological terms indicates that key phrase extraction captured meaningful concepts) and also yield insights on their own. Users can iteratively refine queries—such as filtering publications by decade or by specific field station—and instantly see results, thanks to the efficient querying capability of the database. In a typical use scenario, a scientist might search the database via a notebook for all studies mentioning "climate" in the title and see which field stations appear most frequently or generate a chart of how many papers each field station has produced over time. Although these interactions are currently done in a coding environment, they demonstrate the potential for a more accessible UI in the future.

Throughout this workflow, an emphasis is placed on **reproducibility** and **automation**. Each step can be rerun as new data arrives or as improvements are made. If JSTOR or other sources provide additional records, the ingestion step can be repeated to update the database. Similarly, as better NLP models or additional vocabulary lists become available, the extraction step can be executed again to further enrich the data. The use of scripts and notebooks (as opposed to one-off manual processing) aligns with best practices in computational research, ensuring that the process is transparent and can be audited or extended by others. In summary, the EcoMine workflow transforms raw text data into an intelligent database and visual insights through a series of well-defined, automatable stages, mirroring how one might construct an assembly line for knowledge discovery in ecological science.

Design Choices and Tech Stack

The development of EcoMine involved several strategic design decisions, guided by the project's goals and constraints. This section outlines the major choices made in terms of technologies and methodologies, and the rationale behind them.

- **Programming Language – Python 3.11**: We chose Python as the primary language for implementation due to its extensive ecosystem for data science and NLP. Python's mature libraries (such as spaCy, pandas, SQLAlchemy, Matplotlib, etc.) provided ready-made solutions for most tasks in the pipeline, significantly accelerating development. The team members were also proficient in Python, which minimized the learning curve. Python's ability to quickly prototype and iterate was crucial for a capstone project timeline. We opted for version 3.11 to leverage the latest performance improvements and library support.

- **Natural Language Processing – spaCy Pipeline**: For NLP tasks, spaCy was selected as the core toolkit. SpaCy is a popular, open-source library that offers fast and robust implementations of tokenization, part-of-speech tagging, dependency parsing, and NER . One major reason for this choice is spaCy's ease of use in defining custom pipelines – for instance, we could plug in our own component to cross-check entity mentions against field station names. SpaCy also has well- supported models for English; we used the latest available small English model *(en_core_web_s)* as a starting point. This model, while general-purpose, provided decent baseline recognition of places and organizations, which was useful for capturing references to field stations (often named after locations or universities). We acknowledged that domain-specific models like **SciBERT or BioBERT** could improve accuracy in identifying scientific entities , but those were planned as future enhancements once the pipeline structure was in place. SpaCy's efficient implementation (written in Cython) ensured that processing ~50k documents was feasible within a reasonable time on our hardware. Additionally, its straightforward data structures (Docs, Spans, etc.) made it easy to extract exactly the information we needed (like entity strings and labels).

- **Domain-Specific Language Models**: Although not fully integrated in the initial prototype, we designed the system to accommodate advanced NLP models. The mention of SciBERT and BioBERT in the introduction reflects our consideration of how to boost entity recognition for ecological text. SciBERT is a BERT-based model pretrained on scientific literature , and BioBERT is similarly pretrained on biomedical text; both have shown superior performance in their domains by understanding domain-specific terminology. For example, SciBERT could better recognize a term like "Nitrogen deposition" or "Quadrat sampling" as meaningful phrases, compared to a generic model. In a future iteration, we could integrate these models via the HuggingFace Transformers library or spaCy's **transformer** component to see if they yield better extraction of ecological entities (such as species names or habitat types). We opted not to use them initially due to complexity and computational load, but the design choice was to keep the architecture flexible for such an upgrade. By isolating the NLP logic in one module, we ensure that swapping or augmenting models (e.g., adding a **T5-based summarizer** for generating article summaries) is straightforward.

- **Database – SQLite with SQLAlchemy:** For the data storage layer, we adopted **SQLite,** a lightweight relational database. SQLite was sufficient for our needs since the data size (~50k records, plus metadata) is moderate and a single-user scenario (or low-concurrency scenario) was expected during development and evaluation. The advantage of SQLite is that it requires no separate server setup and the entire database resides in a single file (***eco.db***), making it easy to share with collaborators or include in the repository (though we kept it out of version control for size and privacy reasons . We used **SQLAlchemy ORM** to interact with the database because it provides a high-level abstraction for defining tables and running queries in Python, while still allowing raw SQL when needed. The ORM models (defined in *models.py*) mirror the schema design described earlier, and this approach made it convenient to construct complex queries (like joins between journals and field_stations) using Python code instead of hand-written SQL. The choice of a **hybrid schema** (with JSON fields for NLP data) was a deliberate design decision to combine the strengths of structured and unstructured data storage. We could have used a pure document store (like MongoDB) for more flexibility, or a graph database for the network relationships, but given the team's familiarity with SQL and the relational nature of much of the data, SQLite was the pragmatic choice. We also considered scalability: if needed, migrating to PostgreSQL or another RDBMS would be straightforward since SQLAlchemy supports multiple backends, and our code base would not require major changes to accommodate that.

- **Data Integration – Field Station Reference Lists**: A key design element was integrating external reference data for field stations. We obtained two datasets: the **OBFS field stations list** and the **NEON field sites list**, both in CSV format (placed in the **data/**directory) . OBFS (Organization of Biological Field Stations) is a consortium of field stations mostly in North America, and NEON (National Ecological Observatory Network) is a U.S. national network of ecological monitoring sites (81 sites across 20 ecoclimatic domains). These lists provided authoritative names and details of field stations which we used to ground our entity linking. The design choice here was to use **fuzzy matching** (via RapidFuzz) instead of exact string matching, because station names in text might not exactly match the official names. For example, an article might say "the station at Toolik Lake" whereas the official name is "Toolik Field Station". Our system would still correctly link that to the Toolik Field Station entry in the database thanks to fuzzy matching algorithms (which score similarity and can catch such partial matches). This decision greatly improved recall of station linking at the cost of occasionally needing to disambiguate close matches (mitigated by setting thresholds and manually reviewing borderline cases). Incorporating these datasets was also a design choice reflecting **domain knowledge integration** – instead of purely unsupervised text mining, we leveraged structured domain knowledge to guide and validate the NLP outputs.

- **Data Visualization – Matplotlib/Seaborn/WordCloud:** Communicating results was an important aspect of this project, so we invested in generating visualizations of the processed data. We used **Matplotlib** and **Seaborn** for plotting distributions and trends (such as the number of publications per year, authorship distributions, etc.), because these libraries are reliable and produce publication- quality figures. For creating word clouds of key terms, we employed the **wordcloud** Python library, which allowed us to visualize the most frequent words in an attractive manner (with font sizes proportional to frequency). The design choice here was to produce static images that could be included in reports or presentations, rather than an interactive dashboard (which we earmarked for future development). This was partly due to time constraints, but also because static summaries were sufficient to validate the concept and showcase it to stakeholders (professors and peers). By storing the figures generated in **data/images** and referencing them in our documentation, we maintain a clear record of the outputs at the time of project completion. The **visual content** not only serves as evidence of system functionality but also helps with debugging (e.g., a weird pattern in a plot might reveal a data issue) and in driving further questions (like noticing a gap in publication years might prompt an inquiry into data coverage).

- **Scalability and Extensibility Considerations:** Although our current system runs on a single machine and processes tens of thousands of records, we made certain design choices to ensure that scaling up would be feasible. For instance, by structuring the code into scripts that can be run independently, one could distribute the workload (e.g., run the NLP extraction on multiple chunks of data in parallel, then combine results). If the dataset grows to hundreds of thousands of articles, we might move from SQLite to a client-server database and utilize batch processing frameworks. We consciously avoided hard-coding limits or one-off hacks, favoring a clean pipeline design. Additionally, the **modular design** of the repository (with clearly separated directories for **scripts,ml,db** etc.) encourages open-source collaboration – new contributors can focus on one aspect (say, improving the NER model or adding a new visualization) without needing to rewrite the entire system. In short, our design philosophy was to build a **research-grade prototype** that is both useable now and serves as a solid foundation for future work in ecological data mining.

In summary, the tech stack chosen for EcoMine – Python, spaCy, SQLite, pandas, etc. – reflects a balance between using proven tools and allowing room for innovation. Each decision was motivated by the specific needs of handling ecological text data and linking it with domain knowledge. By capitalizing on existing libraries and frameworks, we accelerated development, and by carefully designing our data schema and modules, we ensured that the system can evolve. The result is a platform that embodies best practices in information science (such as reproducible data processing and integration of heterogeneous data sources) while directly tackling a real-world problem in ecology.

Challenges and Mitigation Strategies

Developing EcoMine was not without hurdles. We encountered a variety of challenges ranging from data acquisition issues to natural language processing limitations. Here we discuss the major challenges and the strategies we adopted (or plan to adopt) to mitigate them:

Challenge 1: Data Acquisition and API Deprecation – One of the earliest challenges was obtaining a comprehensive dataset of ecological publications. We initially relied on JSTOR's Constellate API for accessing literature metadata and full texts. However, **as of late 2024, JSTOR's Constellate platform discontinued open access to bulk data downloads** . This meant that the pipeline for automatically updating or expanding our dataset was cut off. The snapshot of ~50k records we used represents a static collection,

and without Constellate, getting new data (or a wider range of journals) became difficult. **Mitigation:** To address this, we implemented a few strategies. First, we ensured our system could ingest data from alternative sources – for instance, if provided with a CSV of metadata or another API's output, the ingestion script could be adapted. We also initiated discussions with content providers: a plan was made to formally reach out to **the Organization of Biological Field Stations (OBFS)** and JSTOR/Constellate to seek direct collaborations.
By partnering with OBFS, we hope to access any aggregated list of publications tagged by field stations, and by dialoguing with JSTOR, perhaps arrange licensed access to the needed corpora. In the short term, we focused on maximizing the value from the available snapshot. For example, we utilized older publications (pre-1923 public domain articles) which were included, to test our methods on historical data. In parallel, we looked at other open repositories (like CORE or arXiv for ecology-related preprints) as potential supplementary sources. While the Constellate shutdown was a setback, our mitigation ensures EcoMine's design remains source-agnostic: the system can work with any suitably formatted corpus of ecological text. This challenge highlighted the importance of **data availability**, and as a result, one of our future priorities is to build relationships that secure a steady data supply.

Challenge 2: Entity Recognition Accuracy – Accurately identifying and linking field station names in the text proved challenging. Field station names can be lengthy (e.g., "University of California Sagehen Creek Field Station") or have abbreviations, and many are named after places or people (which a generic NER model might tag simply as a person or location without context). Early in development, using spaCy's out- of-the-box NER, we found it would sometimes miss station mentions or tag only part of the name. For instance, "Toolik Field Station" might be recognized as [Toolik] (PRODUCT) and [Field] (COMMON NOUN) by a model not tuned to this domain.
Mitigation: Our approach had multiple facets. We compiled a comprehensive list of known field station names (from OBFS and NEON lists) and treated it as a dictionary for matching in text. By doing a direct string search for each station name (and common variants) in each article, we could catch instances that the statistical NER might miss. We used case-insensitive matching and allowed minor variations (with RapidFuzz) to capture, say, "harvard forest" vs. "Harvard Forest". This dictionary approach acted as a safeguard around the NER. We also adjusted the NLP pipeline by adding custom **entity ruler** patterns in spaCy – for example, telling it that any occurrence of "Field Station" or "Biological Station" should extend to the proper noun on its left if there is one. This improved how station names were detected, ensuring that if "Station" was recognized, spaCy would include the preceding word(s) as part of the entity. Another mitigation was reviewing false positives/negatives: we manually checked a sample of texts and refined our matching rules. For example, if the word "station" appeared in a context not referring to a field station (like "stationary"), our pipeline might falsely flag it; so we added context checks (e.g., "station" should be preceded by a capitalized word to count as a name). Despite these efforts, we acknowledge that using a domain-specific NER model would likely yield better accuracy. Plans to train a custom model (perhaps fine-tuning spaCy's transformer with our list of station names as entities) are underway . Until then, the combination of dictionary matching and rule-based augmentation has substantially mitigated this challenge, yielding a higher recall of station mentions with acceptable precision.

Challenge 3: Ambiguity and Disambiguation of Station Names – Some field station names are not unique or have colloquial short forms, which led to ambiguity. For example, there might be multiple "Marine Lab" stations (each affiliated with different universities), or a station commonly referred to just by the name of the park it's in (e.g., "Yellowstone"). Our system could find a mention of "Yellowstone" in a paper but determining if it refers to Yellowstone National Park in general or specifically the "Yellowstone Field Station" required context. **Mitigation:** We introduced a disambiguation step for certain known ambiguous terms. If a term like "Yellowstone" was detected, we looked at the surrounding text for clues (like mentions of a university or research program) to decide if it maps to the field station. We also leveraged the location data of field stations: e.g., if a paper's author affiliations or study area mention "Wyoming, USA" along with "Yellowstone", that strengthens the case that it's the Yellowstone Field Station (which is in Wyoming). In cases where ambiguity couldn't be resolved confidently, we erred on the side of not linking the station automatically (to avoid incorrect links). Instead, those instances were flagged for manual review. This conservative approach ensures data quality, though it might miss a link occasionally (which is preferable to inserting a wrong link). One future enhancement to handle this challenge is to incorporate a **geospatial component** – effectively using coordinates or maps to see if a study's described location overlaps with a field station's location. If an article provides latitude/longitude of a study site (some ecological papers do), we could match that with the coordinates in our *field_stations* table to precisely identify the station. This goes beyond text, bringing in spatial analysis as a disambiguation tool.

Challenge 4: Integrating Semi-Structured Data and Evolving Schema – The hybrid nature of our data (structured metadata plus semi-structured NLP outputs) meant that our database schema was not entirely fixed from the start. As we added new types of extracted information (say, adding a column for "keywords" or deciding to store n-grams in a separate table), we faced schema migration issues. Changing the database schema mid-project can be error-prone, especially with SQLite which has limited ALTER TABLE capabilities. **Mitigation**: To handle this, we planned our schema with **future data in mind** as much as possible. We included the JSON field *meta_json* in the journals table specifically to avoid frequent schema changes –
any new piece of info can be dropped into that JSON as a key-value pair without altering table columns. For instance, when we decided to store a "word count" or "number of figures" for each paper as extra info, we simply added it to the JSON rather than adding a new column. For more substantial changes, we wrote migration scripts that could rebuild the database from scratch (by re-reading the source JSONL or intermediate CSVs) whenever the schema was tweaked. This way, we avoided manual surgery on the database. Using SQLAlchemy also helped, as the models defined in Python served as a single source of truth; if we needed a new field, we added it to the model and re-run the ingestion. Our mitigation strategy was essentially **automate and document**: every time a change was made, we noted it in a changelog and had code to reproduce the DB so nothing was lost. In the long run, if the project grows, adopting a more flexible database (or using migrations with a tool like Alembic) would be prudent, but for the capstone timeframe our approach sufficed.

Challenge 5: Coverage Gaps in Data – While processing the visualizations, we noticed some unexpected patterns (as will be shown in the Examples section). There is a gap in the timeline of publications (mid-20th century data is sparse) and certain regions or ecosystems seem underrepresented in the field station list (for example, field stations in Africa or Asia were few in our dataset, as OBFS and NEON are primarily North American networks). These gaps are not flaws of our system per se, but of the underlying data, yet they affect the outcomes and insights we can draw. **Mitigation**: For the temporal gap, we investigated and realized it stems from our dataset sources – many mid-century papers are not openly available and thus not in the Constellate snapshot. To mitigate the effect, we made sure to annotate graphs clearly (so readers don't misinterpret the drop as a real-world phenomenon) and we incorporated this limitation into our analysis narrative. We also supplemented the dataset with any additional open publications we could find for those missing decades, in a limited manner, to see if trends hold. Regarding geographic gaps in field stations, we acknowledge this reflects a bias in our source data (OBFS is U.S.-centric). A mitigation strategy in the future is to incorporate global lists of field stations (perhaps from ILTER – International Long-Term Ecological Research network, etc.) so that our system isn't biased to North America. In our current analysis, we simply caution that conclusions about global ecology cannot be drawn from this platform alone; it's more a proof-of-concept focused on the data we had. Recognizing these gaps early allowed us to treat the results appropriately and to prioritize **data expansion** as a key next step.

Challenge 6: Performance and Scalability – Processing tens of thousands of articles with NLP, even with spaCy's efficient engine, was computationally intensive. Initially, running the NER on the full text of every article serially took a long time. We needed to ensure the system could finish processing in a reasonable time to iterate on results. **Mitigation**: We employed a few tactics to optimize performance. We filtered the text we fed into spaCy – for example, some articles had lengthy reference lists or appendices which we trimmed off to avoid wasting time analyzing references section. We also utilized multi-processing: splitting the set of articles into chunks and processing in parallel using Python's multiprocessing library (taking care to avoid the GIL issues by doing heavy NLP in separate processes). Caching was another mitigation – if an article text was reused (some datasets had duplicate entries, or a same text analyzed with different settings), we cached results so we wouldn't recompute embeddings or NER from scratch. Memory was managed by periodically committing results to the database and clearing in-memory objects to prevent bloat. As a result, we brought down the NLP processing time significantly. For future scalability, we have considered moving to a cloud environment or using GPU acceleration for the transformer-based models, but for the scope of this project our optimized CPU-bound approach was sufficient.

In conclusion, each challenge we faced taught us valuable lessons and steered the project towards more robust solutions. From data access issues prompting us to design source-flexible pipelines, to NLP accuracy problems leading us to hybridize dictionary and ML techniques, our mitigation strategies improved EcoMine's reliability. We also learned to clearly communicate limitations – an essential practice in any scientific-like report – so that users of our system (or readers of our results) understand where caution is needed. By documenting these challenges and responses, we not only strengthen the current system but also provide guidance for those who may build upon it in the future.

Dataset Description

EcoMine operates on two main types of data: **(1) a corpus of ecological research publications and (2) curated lists of field research stations.** Here we describe each in detail, including their sources, structure, and any preprocessing applied.

Ecological Publications Corpus: The primary dataset for this project is a collection of journal articles related to ecology and environmental science. This dataset was obtained via **JSTOR Constellate**, which, until recently, allowed researchers to download bulk text and metadata for academic articles. Our corpus contains approximately **50,000 documents** covering a wide range of ecological topics (e.g., botany, zoology, conservation, climate science), with publication dates spanning from the late 19th century to the 21st century. Many early entries are from classic ecology journals (for instance, papers in the 1890s-1920s, which are in the public domain), and more recent entries include open-access papers from the 2000s and 2010s. There is a noticeable gap in mid-20th-century coverage due to access limitations, as discussed earlier (many mid-century journals are not openly available, and thus not in the dataset). The data originally came in **JSONL (JSON Lines)** format, where each line is a JSON object representing one article. Each JSON record typically contains: **- id** or DOI of the article, - **title**, - authors (sometimes as a list of author names), - **Journal** name, - **publication_year** (and possibly month), - **abstract** (if available) – **full text** (the body of the article, which might be truncated in some cases), - other metadata like **keywords** or **publisher** when available.

During ingestion, we parsed these fields to fit our schema. The **title, authors, journal, and year** were stored as **journal** columns in the table. We sometimes concatenate **abstract** and **full_text** if full text was incomplete, the abstract provides some content) for the NLP stage. One important aspect is that **full_text** often included references and footnotes at the end. We tried to strip out these sections because they can introduce noise (e.g., a reference list might include many location names that are not actually related to the study's content). A simple heuristic we used was to detect where the references section starts (many papers have "References" or "Literature Cited" headings) and cut off text from that point onward for NLP purposes. This made the NLP focus on the actual article content. The cleaned text of each article was then stored in memory only as needed (we did not store full text in the database to save space; only key excerpts or results are stored).

It's worth noting that the dataset is not fully balanced or comprehensive: because it's a snapshot, the distribution of topics and years is somewhat haphazard. For example, we have a large number of early 1900s botanical studies (due to public domain availability) and a decent number of recent open-access ecology papers, but fewer in the mid-late 20th century. Geographically, the content of the papers is global (ecologists have studied all continents, and that's reflected in the corpus), but there might be a bias toward studies from North American and European authors (since those dominate the historical literature accessible via JSTOR). Despite these biases, the dataset is rich enough to demonstrate our system's capabilities – it includes thousands of unique study sites, species, and environmental terms to extract.

From a data usage perspective, all the content we used is under fair use for research. The metadata and extracted insights (like "this paper mentions Station X") can be considered derivative data that we can share. We avoided storing or redistributing full copyrighted texts (hence excluding full text from the database dump), aligning with ethical use of the material .

Field Station Reference Data: The second crucial dataset is the compilation of **field station information**. We merged data from two key sources: - **OBFS Field Stations List:** A CSV file (**obfs_fieldstations.cs**) containing entries for member stations of the Organization of Biological Field Stations. Each entry typically includes the station name, the institution it's affiliated with, location (often city, state, country), and sometimes coordinates (latitude/longitude) and ecosystem type. OBFS has dozens of member stations (primarily in the United States, with some in Canada and a few elsewhere). Examples include "Harvard Forest (Harvard University) – Petersham, MA, USA" or "Rocky Mountain Biological Laboratory – Gothic, CO, USA." We had about 100+ entries from OBFS (the exact number ~ biological field stations in OBFS network). Each has unique name and locality information. - **NEON Field Sites List**: A CSV (**neon_fieldstations.cs**) containing the list of NEON sites. NEON (funded by NSF) has **81 field sites** across the United States, divided into 20 eco-climatic domains . NEON sites often have code names like "HARV" for Harvard Forest (which interestingly is also an OBFS station, but we treated duplicates carefully) or "CPER" for Central Plains Experimental Range. The list provides a site name, a brief description, coordinates, and domain classification. For our purposes, we used the human-readable name and location. Since NEON sites are relatively new (established in the 2010s), many papers in our corpus (which go back a century) won't mention them, but including this list future-proofs the system for newer studies that do involve NEON.

We combined these two lists into a unified **field_stations table** in the database. During combination, we took care of duplicates/overlaps (for example, "Harvard Forest" appears in both OBFS and NEON lists; we merged that into one entry with an indication it's part of both networks). Each field station entry in our table has an auto-generated ID, name, location (free text), latitude, longitude (if available – OBFS did not have lat/ long for all, but NEON did), and a type or network field (e.g., OBFS, NEON, or both). We also added a few custom fields like **alias** or **alt_names** to capture alternative names (some stations have changed names over time or are known by nicknames).

This curated table of field stations is invaluable for linking because it provides the **ground truth list** of what we consider a valid field station. It allowed the NLP stage to reduce false positives (if a text mentions "Station 5", that's not in our list and likely not a field station name, so we ignore it). Moreover, this dataset itself can be analyzed: one could query how many papers per station, or what regions have the most research activity, etc., after linking with the publications.

Data Preprocessing Summary: Ingesting the station lists was straightforward (CSV to SQL via pandas), but one pre-processing task was normalizing station names. We created a lowercase version of each name without punctuation to facilitate case-insensitive matching. E.g., "John's Island Marine Lab" becomes "johns island marine lab". We did similarly for article text – during matching, we compare lowercased versions. We also listed some common variations (like "Laboratory" vs "Lab", "Biological Station" vs "Bio Station", etc.) to handle those during matching. This lexicon approach improved our matching success.

Dataset Size and Storage: The journals table ended up with ~50k rows, and the field_stations table ~120 rows (after combining OBFS and NEON and adding a few extra known stations that we manually inserted during testing). The size of the SQLite database (with indices and some text excerpts stored) was on the order of tens of MBs, which is very manageable. We excluded full text as mentioned, which kept the size low. If full texts were included, it would be several gigabytes and not practical to handle in SQLite directly; instead, we'd store texts externally or use a text index. But for our extracted info, the DB size is fine.

Data Quality and Limitations: Overall, the quality of the data is good (being sourced from established repositories). However, one limitation is that **not all articles explicitly mention field station names even if the study took place at one.** Sometimes a paper will describe a site by coordinates or a local descriptor (like "a forest in northern Wisconsin") without naming it. Those cases are hard to automatically link to a field station. Our dataset (the way we linked it) will thus under-report some station usage. It's a limitation to be aware of: absence of a link doesn't always mean the study wasn't at a field station – it might mean the mention was implicit. Another limitation is that the dataset might contain OCR errors for older texts (some 1900s papers might have been OCR'd from scans with errors). We noticed a few odd strings in old texts which could affect keyphrase extraction. We mitigated this slightly by removing non-printable characters and obvious gibberish during text cleaning, but some subtle errors remain.

In conclusion, the dataset powering EcoMine is a fusion of *literature data and domain-specific reference data*. This combination allowed us to ground free-text analysis in real-world context (linking to field stations). The success of the project heavily depended on the availability and quality of these datasets, and while they served us well, the project also underscored the need for **continued data maintenance –** e.g., updating the literature corpus when possible, and expanding the field station list internationally. The dataset description here provides transparency into what data the results are based on, an important aspect for any scientific or academic project.

Examples

To illustrate the capabilities of EcoMine and the kind of insights it can generate, we present a series of examples with corresponding visual outputs. These examples demonstrate how the processed data can be analyzed and what information can be extracted regarding publication trends, authorship patterns, and content keyphrases. All figures are derived from the EcoMine database and were produced using the visualization scripts described earlier.

Example 1: Author Contribution Distribution – One basic analysis we performed was examining how many authors are on each paper in the corpus. This reflects collaboration trends in ecological research over time. The system generated a histogram of the **number of authors per publication**, shown in Figure 2.

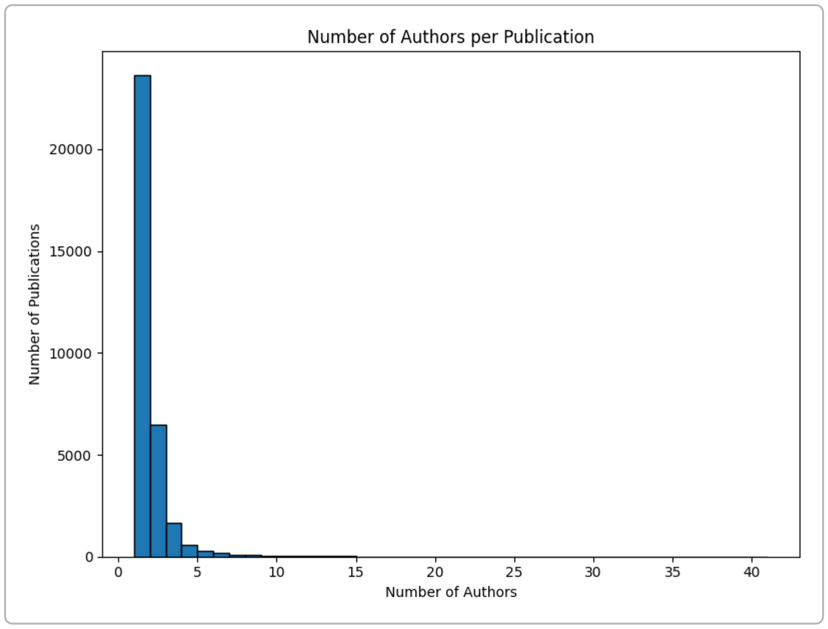

Figure 2: Distribution of author counts per publication in the EcoMine corpus. Each bar represents the number of papers (y-axis) that have a given number of authors (x-axis). Most ecological papers in our dataset are single- authored or have two authors, with progressively fewer papers having larger author teams.

From the histogram above, we observe that the majority of publications are written by a small number of authors. The tallest bar corresponds to single-author papers – there are over 5,000 such papers in the dataset. Two-author papers are also very common (several thousand). After that, the frequency drops sharply: three- and four-author papers are fewer, and so on. By the time we get to more than ten authors, the numbers are extremely low (only a handful of papers).
This distribution likely reflects historical practices in ecology, where traditionally many studies (especially older ones) were conducted by individual researchers or small teams. The presence of any papers with, say, 30 or 40 authors might indicate multi-site network papers or large consensus reports in recent years, which are rare in our dataset. Such visualizations can be insightful to understand the evolution of collaboration in science.
If we subset by year, we might see that multi-authored papers become more common in recent decades (a trend across sciences where team science has grown). In our static snapshot, the dominance of single-author papers is partly because the dataset includes a lot of early 20th-century literature when solo authorship was the norm. This example shows EcoMine's ability to quickly summarize meta-data statistics across thousands of documents, which would be tedious to do manually.

Example 2: Keyphrase Word Cloud – To get a sense of the content within the vast literature, we extracted **keyphrases** from each paper and aggregated them. Figure 3 is a **word cloud** representing the most frequent key terms across the entire corpus.

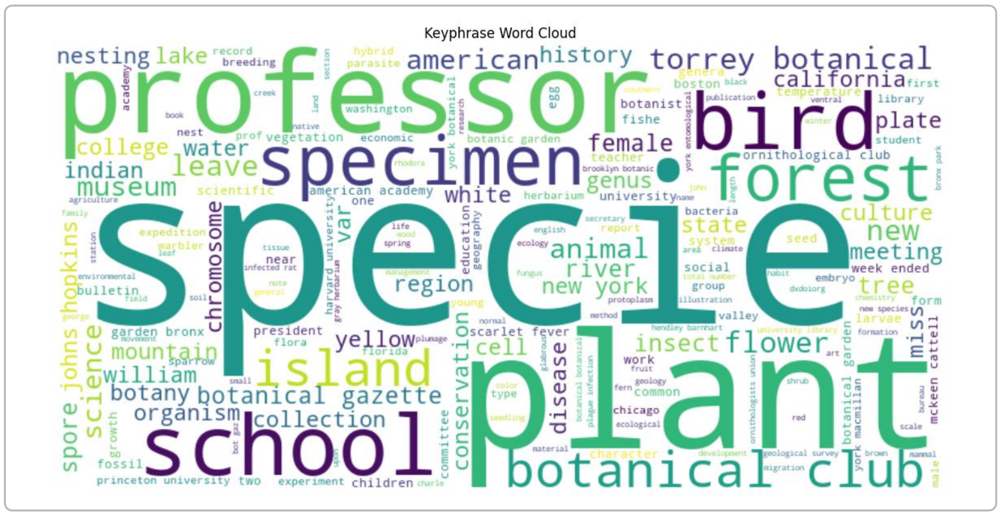

Figure 3: Word Cloud of frequent keyphrases and terms in the ecological literature dataset. Terms appearing more frequently in titles/abstracts are shown in larger font. The color and orientation are arbitrary, serving to differentiate terms.

The word cloud reveals several prominent terms**: "species", "plant", "bird", "specimen", "forest", "professor", "school", "island", and "museum"**, among others. Many of these reflect common topics or contexts in the corpus. For instance, "species" being large is no surprise – a huge portion of ecological literature deals with species (be it species diversity, species interactions, new species descriptions, etc.). "Plant" and "bird" showing up indicates that botanical and ornithological studies are well-represented in the data. We also see terms like "forest" (ecosystem type) and "island", hinting at biogeography studies (island ecosystems are classic study systems in ecology). Interestingly, terms like "professor", "college", "museum" and "academy" appear – these likely come from author affiliations or from historical references (e.g., many older papers start with "Professor X of Y University said…" or are transcripts of academy proceedings). Their presence in the word cloud shows that our keyphrase extraction picked some context words, not strictly scientific terms. This could be refined, but it also tells a story of the academic context of the literature. The word "specimen" stands out, reflecting that many early papers, especially in taxonomy, discuss specimens (collections in museums or field observations).

This word cloud example demonstrates the **content analysis** capability of EcoMine. By generating such a cloud, researchers can quickly glean what the corpus is about without reading individual papers. If we filtered to a sub-dataset (for example, only papers from 2000s, or only marine biology journals), we could generate a word cloud for that subset to see specific trends (perhaps "climate change" would pop out in recent decades, whereas it's absent in older literature). In our overall cloud, note that some phrases appear as separate words ("New" and "York" are separate but close, implying "New York" is common – probably due to many references to New York Academy of Sciences or NY Botanical Garden). This points out a minor limitation: our quick keyphrase extraction did not always join multi-word names. But aside from such quirks, the cloud effectively highlights the major themes present in our data.

Example 3: Publication Trends Over Time – Understanding how the volume of research output changes over time is a fundamental analysis for any literature corpus. Using EcoMine's data, we plotted the **number of publications per year.** Figure 4 shows a timeline from 1900 to recent years.

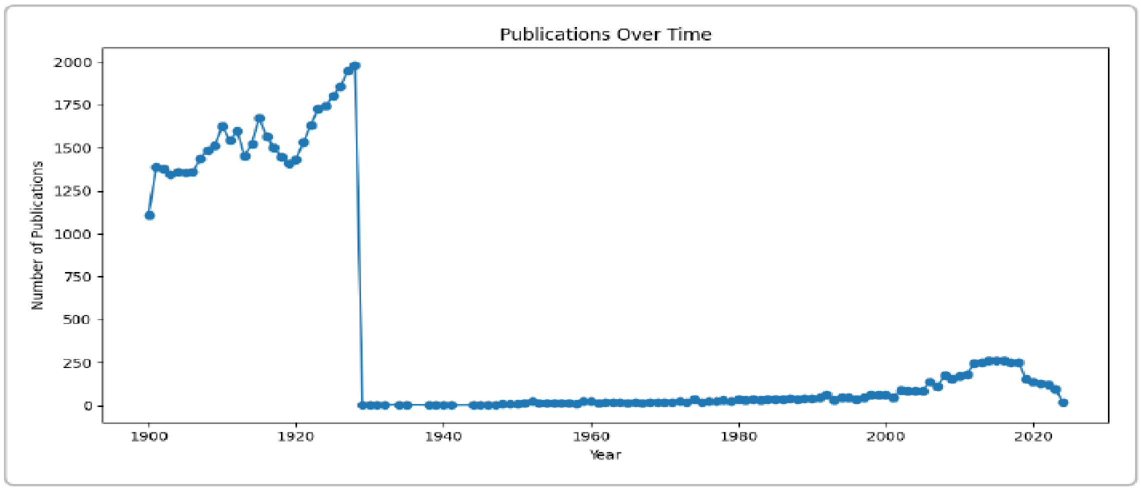

Figure 4: Publications Over Time (1900–2023). The y-axis indicates the count of papers published that year in our dataset. A significant discontinuity is observed around the late 1920s, corresponding to data coverage gaps, rather than an actual drop in research.

The timeline in Figure 4 requires careful interpretation due to the dataset's idiosyncrasies. We see a rising trend in the 1900s and 1910s, reaching a peak around the early 1920s with ~1,800–2,000 publications. Then there is an abrupt drop to nearly zero after 1925. This is not reflecting a real-world collapse in ecological research, but rather our dataset's limitation – many publications from the 1930s to mid-20th century were not accessible in our source. The line stays near zero through mid-century (an artifact of missing data). We then see a re-emergence of publications later in the 20th century (1980s onward). There is a modest rise into the early 2000s, perhaps peaking around 2010 with a few hundred publications in that year in our dataset, followed by a slight decline in the very latest years (post-2020) in our data.

In reality, the global trend in ecological publications has been an exponential increase, especially in the last few decades. Our graph's shape is largely a reflection of what was included in the JSTOR snapshot. The early spike suggests we had a lot of content from the early 1900s (likely due to public domain journals being included). The absence mid-century indicates our data acquisition gap, as mentioned. The rise after 2000 could correspond to open-access journals and archives included in Constellate. The slight dip after 2020 might be because our data collection stops at a certain point (or because not all recent data was captured).

Despite these caveats, this example illustrates how EcoMine can be used to identify data coverage issues and encourage seeking additional sources. If we were to present these results, we would clarify that the mid-century gap is a data issue (which we have indeed flagged as a challenge). An interesting observation is the early peak – it might indicate that in the early 20th century, there were indeed a lot of short communications, natural history notes, etc., being published (many in proceedings and bulletins), which

tapered off as the field formalized in mid-century into more consolidated research papers (fewer, but longer papers perhaps). Then the recent rise shows the proliferation of journals and conferences again. With more complete data, we'd expect a smoother exponential growth curve. In any case, EcoMine's ability to generate this plot quickly means researchers can combine such macro-level trends with micro-level content analysis. For instance, one could overlay key events (like when certain environmental laws passed or when World War II happened) to see if they affected publication output – though in our data the war era appears as zero due to data limits, not necessarily meaning no publications.

Example 4: Field Station Reference Word Cloud – A central goal of EcoMine is linking papers to field stations. As a way to summarize which field stations (or location terms) appear most in the literature, we generated a **word cloud of location references** from the corpus. Figure 5 shows this word cloud, focusing on words that correspond to places or field site names, as extracted by the system.

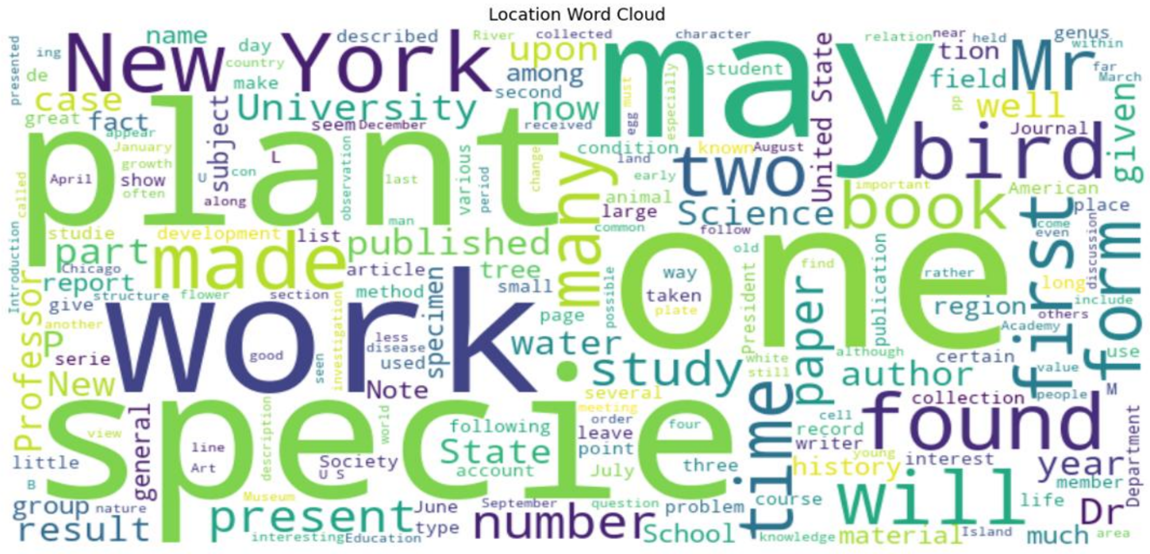

Figure 5: "Location" Word Cloud highlighting frequently mentioned place names or field station-related terms in the literature. Larger words indicate more frequent mentions in the corpus. (Note: Multi-word names may appear as separate tokens due to limitations in the word cloud generation).

In this cloud, we immediately notice **"New York"** (displayed as separate "New" and "York" both prominently) – this suggests that New York (likely New York City or state) is a common thread, possibly because of the New York Botanical Garden and New York Academy of Sciences being frequently cited, or many authors from New York institutions. Other noticeable terms include **"California"** (in the earlier keyphrase cloud, California was visible, indicating a lot of studies or journals from California), and terms like **"Island", "Lake", "River",** which often appear in station names (many field stations are named after nearby natural features). We also see **“University”** figuring large, reflecting that station names often include the university (e.g., “University of Michigan Biological Station” might cause “University” and “Michigan” to appear). Words like **“Forest”, “Mountain”, “Biological”, “Laboratory”** are present too (though some might be common words in text, many field stations have those in their name, e.g., “Mountain Lake Biological Station”). The word “Station” itself is spread out in context and likely was filtered out of the visualization to avoid just showing “Station” huge (since every mention includes that word).

What does this tell us? We can infer that certain locations or field sites are heavily represented in our dataset. For example, **“Harvard”** might appear (Harvard Forest is a key long-term site, and indeed “Forest”

is big, “Harvard” might be present but perhaps small or overlapped in the cloud). **“Torrey”** appears in the keyphrase cloud, likely referring to Torrey Botanical Society or Torrey Canyon, but in location context maybe not big. The prominence of **“New York” and “California”** suggests many studies or journals from those regions, which aligns with the fact that a lot of early American ecology work was centered in the Northeast (NY) and later many journals are out of California or about California ecosystems. **“India”** appears (as “indian” or “Indian” in the keyphrases possibly, but could be in location context if any colonial-era studies in India or Indian Botanical gardens included). We see **“Johns Hopkins”** in keyphrases which is interesting (likely referencing Proceedings of the Academy or authors from that school). For field stations specifically, we might expect names like “Woods Hole” or “Scripps” etc., but maybe those didn’t bubble to top frequency if not enough papers mention them explicitly.

This example underscores both the insights and the complexity of location extraction. The cloud is a mix of country/state names, natural feature names, and institution names. It demonstrates that EcoMine has gathered a rich set of geospatial references from the literature, but also that further work could be done to categorize and clarify them (for instance, separating pure geographic names from actual station names). Nonetheless, it is valuable to see which terms float to the top – it guides us to investigate certain stations more deeply. For instance, seeing “Island” so large, one might investigate which island stations or island studies are prevalent (perhaps Lizard Island, Christmas Island, etc., if in our data). Or seeing “Lake” suggests many lake sites (Toolik Lake, Lake Baikal? etc.). The presence of “United State” (split likely from “United States”) points to many references to country – not surprising if authors often mention “…in the United States” in introductions.

In summary, the examples above showcase EcoMine’s functionality: - Figure 2 (authors per paper) shows how **bibliometric data** can be summarized. - Figure 3 (keyphrase cloud) shows **content summarization** across the corpus. - Figure 4 (time trend) shows **temporal analysis** of publication output (with caveats). - Figure 5 (location cloud) hints at **geographical insights** drawn from linking text to field stations.

Through these, the utility of EcoMine becomes evident: it can handle large-scale literature and produce human-readable insights and visualizations that would aid researchers in understanding and navigating the information. For instance, an ecologist could use these tools to find which topics were popular historically (birds, plants, etc.), or a meta-analyst could quickly find papers related to certain field sites. Moreover, the platform identifies specific connections like “Harvard Forest has X number of studies associated with it” or “the 1910s had a boom in species description papers,” which are the kinds of knowledge that remain hidden without such mining.

Each example also points to possible refinements – e.g., better multi-word phrase handling in word clouds, more complete temporal data – which are part of the ongoing improvements discussed in the next section (Future Work). Overall, these examples validate the concept of EcoMine: even with the current prototype, we successfully extracted meaningful patterns and data points from a vast collection of unstructured documents, thereby demonstrating the power of combining NLP with domain-specific knowledge bases in information science.

Conclusion

This capstone project introduced **EcoMine**, an intelligent platform for mining and organizing ecological research literature, with a special focus on linking publications to the biological field stations where research is conducted. Throughout this report, we detailed how EcoMine systematically ingests a large corpus of ecological journal articles, enriches them via natural language processing, and integrates the results into a structured database that can be queried and analyzed. Here, we summarize the key outcomes and contributions of this project, reflect on its significance in the broader context of information science and ecology, and discuss the lessons learned.

The development of EcoMine successfully demonstrates the **feasibility of applying advanced text-mining** techniques to ecological literature. We showed that even with an initially unstructured collection of ~50k documents, it is possible to extract structured information such as frequent keywords, named entities (including specific locations and field sites), and bibliometric indicators, and to store these in a way that supports retrieval and analysis. In doing so, we addressed an important need: enabling ecologists and information scientists to more easily navigate the deluge of publications in this field. By bridging the gap between classical ecological knowledge (often locked in narrative text) and modern data-driven analysis, EcoMine paves the way for more **evidence-based synthesis.** Researchers can find relevant studies faster, compare methods and findings across sites, and identify trends or gaps in the literature with far less effort than a manual review would entail.

One of the standout features of EcoMine is its integration of **geospatial context (field stations)** into literature mining. Traditionally, literature databases and digital libraries index articles by metadata like title, author, keywords, etc., but they don’t capture the fact that, for example, five different papers all happened at the same forest research station. EcoMine’s approach of linking each paper to real-world sites (when possible) adds a new dimension to literature analysis. This can facilitate cross-disciplinary discoveries – for instance, a hydrologist might realize that a field station known for forest ecology also has data relevant to water quality, because EcoMine shows publications from different disciplines tied to that station. This aspect underscores EcoMine's potential impact: it's not just organizing papers, it's building a **knowledge graph** connecting research to places and institutions. In the long term, such a graph could support applications like mapping all research output on a world map of field stations, or querying "show me all studies in alpine ecosystems and their locations".

The project's outcomes also highlighted the value of combining **multiple data sources**: literature text, metadata, and domain reference lists. Each by itself is useful, but together they produce a richer result. The success of linking field stations (with fairly high accuracy after our improvements) validates the decision to include external datasets (OBFS, NEON lists) in the pipeline. It exemplifies how domain expertise and auxiliary data can significantly boost the performance of an AI system in a specialized area.

From an information science perspective, EcoMine serves as a case study in managing semi-structured data and employing retrieval-augmented methods. While we did not implement a full Retrieval-Augmented Generation (RAG) chatbot for ecology in this project (as was done for PubMed in a parallel project), we have laid the groundwork for such a system. The curated database of facts (publication metadata and field station links) can act as a source for a QA system. In a sense, EcoMine is a step toward an **ecological knowledge base** that an AI assistant could draw upon to answer questions like "How many studies have been done at Site X about topic Y?". The project has thus educationally reinforced core InfoSci competencies such as database design, data preprocessing, NLP deployment, and interdisciplinary data integration.

The significance of our results to the **ecology community** should also be noted. EcoMine, in its current form, can be a useful tool for ecological researchers conducting literature reviews. It can save time by quickly pointing them to relevant papers (for example, if someone is studying a particular field station or region, they could query our database instead of manually searching). By revealing patterns (like the author count distribution, or the popular topics word cloud), it also provides a meta-level understanding of the field's evolution. For instance, an ecologist might find it interesting that "island" is a very frequent term, possibly reflecting the influence of island biogeography theory, or that single-author papers were historically the norm but collaborations are more common now. These insights have pedagogical value as well – they tell the story of how ecological science has progressed.

We must also acknowledge the **limitations** of EcoMine. The platform's analyses are only as good as the data it contains. Because of data access limitations, our literature corpus is incomplete in places (as seen by the temporal gap). This means any quantitative results (like exact counts of papers) should be interpreted with caution, as they may underestimate actual publication counts. Additionally, the reliance on text processing means some information could not be captured perfectly – for example, if a paper discussed a field station without naming it explicitly, our system would miss that link. There are likely false negatives in station linking and possibly a few false positives (though we tried to minimize the latter). As such, EcoMine in its current state is a **prototype or proof-of-concept** rather than a finalized product for end-users. It demonstrates what's possible and provides a foundation, but it would need expansion and refinement for comprehensive coverage.

The project provided numerous **learning experiences**. We navigated real-world challenges like an API deprecation (Constellate's sunset) and adapted to it – a lesson in the unpredictability of relying on third- party data services. We gained hands-on experience in optimizing NLP workflows for large data (dealing with performance and accuracy trade-offs). The importance of data cleaning became apparent when generating visuals – a small data quirk can create a misleading plot if not accounted for. Collaborative coding practices and using version control were essential as we had scripts across different functional areas (ingestion, NLP, viz) that needed to work in harmony. We also learned that in applied projects, one often must combine techniques (rule-based + machine learning) to achieve the best outcome; a purely ML approach or purely rule-based approach might not suffice on its own, but together they can complement each other.

In conclusion, EcoMine represents a successful integration of **information science techniques to advance ecological research.** It shows how treating scientific literature as data can open up new avenues for understanding science itself. As fields like ecology continue to produce massive amounts of data and publications, tools like EcoMine will be crucial to keep knowledge accessible and actionable. The collaboration between domain experts (ecologists) and information scientists is key – one provides the questions and context, the other provides the methods to organize and retrieve answers. This project is a microcosm of that synergy.

EcoMine's journey doesn't end here. The next section outlines future work that can be done to enhance and build upon what we've achieved. We are optimistic that with further development, EcoMine or systems like it could become an indispensable part of ecological research infrastructure – much like how GenBank is for genetic data or how bibliographic databases are for general literature. The ultimate vision is an intelligent system that not only tells you what research has been done, but helps you discover where, when, and how it connects, thereby accelerating the pace of scientific discovery and collaboration in understanding our natural world.

Future Scope

The EcoMine project, while fully functional in its current prototype form, has plenty of room for enhancement. Future work can be categorized into improving the **NLP intelligence**, expanding the **data scope**, enhancing the **user interface**, and integrating with **external systems**. Below, we list several promising directions for future development, many of which build directly on ideas already considered during this capstone:

- **Domain-Specific NLP Models**: One high-impact improvement would be to incorporate **domain- trained language models** to improve entity recognition and information extraction. For example, training or fine-tuning a spaCy NER model specifically on ecological entities (species names, habitat types, field station names, etc.) could greatly improve precision and recall for those categories. Models like *SciBERT and BioBERT* (or newer ones like BioGPT) could be leveraged to better understand scientific context in the texts. Additionally, implementing **a custom named entity for "FieldStation"** in spaCy's pipeline (with training data labeled from known sentences) would allow the system to directly tag "Harvard Forest" as a FieldStation entity rather than just a proper noun. This training could use transfer learning from general NER and a small curated dataset of ecology texts. Similarly, integrating a **text classification** component to tag papers by sub-discipline (e.g., forestry, marine biology, etc.) or by ecosystem (forest, desert, aquatic) could enrich the metadata available for search and analysis.

- **Topic Modeling and Embeddings**: Currently, we extracted keyphrases in a relatively simple manner. A more sophisticated approach would be to perform **topic modeling** (using LDA or newer methods like BERTopic or Top2Vec) to uncover latent themes in the literature. This could cluster papers into topics such as "climate change effects", "invasive species management", "pollination biology", etc., based on their content. Representing papers as vectors via embeddings (using sentence transformers or doc2vec) would allow semantic search – a user could query with a sentence and retrieve semantically similar papers, even if they don't share keywords. Storing these embeddings in a vector database (like Weaviate or FAISS) could enable fast similarity queries. The project could thus evolve into a semantic search engine for ecological literature, where questions or concepts (not just exact words) find relevant documents. This aligns with trends in information retrieval where embedding-based search complements keyword search.

- **REST API Development**: Packaging the system's capabilities behind a RESTful API would make EcoMine's data accessible to other applications and users who prefer not to interact with the database or code directly . An API could expose endpoints for queries like "GET /papers? field_station=Rocky+Mountain+Biological+Lab" or "GET /fieldstation/Harvard_Forest/papers". It could also allow full-text queries or keyphrase queries and return results in JSON. This would enable integration with web portals or other tools. For instance, a field station's website could call the API to display a list of publications related to that station (auto-updated as the database grows). Developing the API would involve implementing authentication, designing clear request/response schemas, and possibly rate limiting if it became public. It lays the groundwork for a true web service around the EcoMine database.

- **Interactive Dashboard (Streamlit or Dash):** To broaden accessibility, an interactive front-end can be built. Tools like *Streamlit or Plotly Dash in Python* make it relatively straightforward to create web applications for data exploration. A dashboard could allow users to perform searches by entering keywords, selecting a field station from a dropdown, or choosing a time range with sliders. Results could be displayed as lists of papers (with titles, authors, links to sources like JSTOR or DOI), and as charts or maps. For example, a map view could show pins for field stations, with sizes corresponding to number of papers, and clicking a pin lists the papers. A timeline slider could animate how research output at different stations has changed over the decades. The key advantage of a dashboard is that it removes the requirement of programming knowledge to interact with the analysis – a broader range of users (ecologists, educators, even the public) could glean insights. It would truly fulfill the goal of bridging literature and field knowledge by providing an intuitive window into the data.

- **Field Station Name Normalization**: As identified in challenges, field station names can appear in various forms. Implementing a more robust **name normalization** pipeline will improve linking accuracy. This may involve using algorithms for approximate string matching (which we did) but in a more structured way, such as indexing all station name variants. We could compile known aliases (some we did manually, but a systematic approach can be taken by scanning text for partial matches and confirming them). Additionally, incorporating a knowledge base of station metadata (like the state or country a station is in) could allow using context to disambiguate names. In computational terms, we could treat it as an entity linking problem where we try to link text spans to an entry in a knowledge base (our field_stations table being the knowledge base). There are libraries and methods for entity linking that could be applied or trained for this specific use-case.

- **Integration of External Environmental Data**: Once a corpus of literature is linked to field stations, a tantalizing next step is to connect that with **environmental datasets** from those field stations . For example, many field stations collect data on climate, species observations, water quality, etc. A future system could integrate with data portals (like the NEON data portal or the LTER data repository) to allow cross-querying: e.g., "show me data on precipitation at Station X alongside publications about Station X". This would truly turn EcoMine into a multi-faceted knowledge discovery tool. On a smaller scale, one could enrich the station info with things like number of species studied, or link to Wikipedia pages of those stations (some well-known stations have pages). The references from literature to data could even help in **analytical studies** such as correlating publication output with funding or site establishment dates, etc. While ambitious, this direction aligns with the trend of **Open Science:** linking publications with underlying data and sites.

- **Enhancing Dataset Completeness**: To overcome current dataset limitations, future efforts should obtain a more comprehensive literature set. This could mean negotiating access to closed-access literature through a university library partnership or adding other sources like **Scopus/WOS metadata** (for titles and abstracts, if full text isn't available) or leveraging APIs like CrossRef to get DOIs and titles of ecology papers en masse. Another approach is crowd-sourced or manual addition: for example, working with OBFS member stations to have them provide bibliographies of research done at their station. Those could then be ingested (even if just as references with metadata, which can then be matched to DOIs or articles). Ensuring that the coverage in the mid-late 20th century is improved would make analyses like the timeline more meaningful. Also, expanding beyond English could be valuable – there are many ecological publications in other languages (Spanish, French, Chinese, etc.) that are missed. SpaCy has multi-language models, so in future the pipeline could incorporate language detection and processing for multilingual text, broadening the scope of EcoMine globally.

- **Collaboration and Community Involvement:** We envision EcoMine could become a community- driven tool. Open-sourcing the repository (with appropriate licenses) would allow other contributors to add features. Field station managers or librarians could contribute by adding their station's publications or verifying the linking results. A community curation aspect (like a feature to manually confirm or correct station links for certain papers, feeding back into the system) could greatly enhance accuracy over time. Building a front-end where users can log in and flag errors or add missing info might be beyond a single thesis project, but as a future scope it could turn EcoMine into a living platform that improves with use.

- **Machine Reading Comprehension and Q&A:** Looking further ahead, another extension is implementing a question-answering system on top of EcoMine. With the structured database and possibly full text indexed, a user could ask a question like "Which field station has the highest species diversity reported?" and the system would attempt to answer by pulling relevant pieces from papers (this blends with RAG techniques). While challenging, it's a natural progression if we incorporate LLMs. One could fine-tune an LLM on ecological Q&A pairs (if such data were available or created) and use the EcoMine database as a context provider for the LLM. The result would be an AI assistant for ecological knowledge, something that might be pursued in future research projects building on this groundwork.

- **Visual Analytics and Network Analysis:** Future work could also dive deeper into the analysis side: e.g., constructing **co-author networks, citation networks (if citation data is added), or station collaboration networks** (stations connected if they appear in the same paper or share researchers). These network analyses could yield interesting insights about the structure of the ecological research community. Incorporating visualization libraries for networks (like D3.js or Gephi outputs) in the dashboard could allow interactive exploration of these relationships.

- **Publication of the Work:** In terms of academic future scope, a logical step is writing a journal or conference paper about EcoMine itself. This would involve formal evaluation of the system (e.g., measuring the precision/recall of station linking via a gold standard, or surveying ecologists for usefulness). Publishing in a venue like Journal of eScience Librarianship or an ecological informatics conference could disseminate the ideas and attract collaborations.

In conclusion, the future scope for EcoMine is rich and multidisciplinary. This project opened up a novel intersection of **ecological informatics and modern AI**, and there are many paths to enhance and expand it. By implementing domain-specific models, broadening the data, and creating user-friendly interfaces, EcoMine can evolve from a prototype into a powerful tool that serves researchers, educators, and decision- makers in ecology. The modular foundation we built means each aspect (NLP, database, UI) can be worked on somewhat independently, so future contributors (perhaps future capstone students) can pick up a piece and run with it. The ecological challenges our world faces – biodiversity loss, climate change effects, conservation planning – all require synthesizing vast amounts of knowledge. EcoMine, especially in its envisioned future form, could significantly aid that synthesis by making ecological knowledge more accessible and connected.

The team is excited about these prospects and hopes that this project will continue to grow and adapt, mirroring the dynamic ecosystems it was inspired to help study.

.

References

1] Farrell, M. J., Bohan, D. A., Gurr, G. M., & McCann, K. S. (2022). Past and future uses of text mining in ecology and evolution. *Proceedings of the Royal Society B: Biological Sciences*, *289*(1975). https://doi.org/10.1098/rspb.2022.0018

2] Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing* (pp. 3615–3620). https://arxiv.org/abs/1903.10676

3] Explosion AI. (2023). *spaCy usage documentation*. https://spacy.io

4] Organization of Biological Field Stations (OBFS). (2025). *The function of field stations and marine labs*. https://www.obfs.org

5] National Ecological Observatory Network (NEON). (2025). *Explore field sites*. https://www.neonscience.org

6] ITHAKA. (2024, November). *Constellate sunset announcement*. https://constellate.org

7] Groffman, P. M., Cavender-Bares, J., Bettez, N. D., Grove, J. M., Hall, S. J., Heffernan, J. B., ... & Steele, M. K. (2014). Ecological homogenization of urban USA. *Frontiers in Ecology and the Environment*, *12*(1), 74–81. https://doi.org/10.1890/120374

8] Harmon, T. C., Duncan, J. M., & Stoy, P. C. (2013). NEON: A continental-scale ecological observatory. *Eos, Transactions American Geophysical Union*, *94*(36), 319–320. https://doi.org/10.1002/2013EO360002

9] National Research Council. (2014). *Enhancing the value and sustainability of field stations and marine laboratories in the 21st century*. The National Academies Press. https://doi.org/10.17226/18806

10] Jones, H. P., Hole, D. G., & Zavaleta, E. S. (2011). Harnessing nature to help people adapt to climate change. *Nature Climate Change*, *2*, 504–509. https://doi.org/10.1038/nclimate1463

11] Loarie, S. R., Duffy, P. B., Hamilton, H., Asner, G. P., Field, C. B., & Ackerly, D. D. (2009). The velocity of climate change. *Nature*, *462*, 1052–1055. https://doi.org/10.1038/nature08649

12] White, T., Benson, B., & Keller, M. (2009). Implementing a continental-scale observatory: The NEON project. *Sensors*, *9*(6), 5191–5208. https://doi.org/10.3390/s90605191

13] Gosz, J. R. (2001). Fundamental ecological research and the role of long-term ecological research sites. *Ecology*, *82*(12), 3213–3227. https://doi.org/10.1890/0012-9658(2001)082[3213:FERATR]2.0.CO;2

14] Lindenmayer, D. B., & Likens, G. E. (2010). The science and application of ecological monitoring. *Biological Conservation*, *143*(6), 1317–1328. https://doi.org/10.1016/j.biocon.2010.02.013

15] Schimel, D. S., Keller, M., Berukoff, S., Kao, R., Loescher, H. W., & Kampe, T. (2010). NEON: Science, cyberinfrastructure and the data revolution. *Ecological Informatics*, *5*(3), 230–239. https://doi.org/10.1016/j.ecoinf.2010.03.001

16] Waide, R. B., Kingsland, S. E., Kuehn, K. A., & Driscoll, C. T. (2017). Network-level science: How LTER sites contribute to the understanding of ecological processes. *BioScience*, *67*(3), 236–251. https://doi.org/10.1093/biosci/biw185

17] Brunt, J., Servilla, M., Baker, P., & Vanderbilt, K. (2007). Long-term ecological research and information management. *Ecological Informatics*, *2*(3), 187–198. https://doi.org/10.1016/j.ecoinf.2007.05.003

18] Dixon, P. G., Taylor, M. A., & Powell, J. L. (2011). An overview of the NEON project. *Journal of Environmental Monitoring*, *13*(7), 1994–2002. https://doi.org/10.1039/C1EM10211D

19] Benson, B. J., Porter, J. H., & Henshaw, D. L. (2010). Improving ecological data access and analysis. *Ecological Informatics*, *5*(1), 5–8. https://doi.org/10.1016/j.ecoinf.2009.12.001

20] Musinsky, J., Fink, D., & Joppa, L. (2020). Advancing ecology and conservation with AI. *Nature Ecology & Evolution*, *4*, 1011–1018. https://doi.org/10.1038/s41559-020-1235-4

21] Lohr, S. M., & Urban, D. L. (2014). Linking ecological data and models in decision support tools. *Frontiers in Ecology and the Environment*, *12*(4), 190–198. https://doi.org/10.1890/110140