Demonstrating EarthLake: A Model Lake System for Earth Observation Foundation Model Management

Binger Chen*
HU Berlin
Berlin, Germany
binger.chen@hu-berlin.de

Haralampos Gavriilidis*
ICSI & UC Berkeley
Berkeley, United States
gavriilidis@berkeley.edu

Luca Gaedicke
BIFOLD & TU Berlin
Berlin, Germany
luca.gaedicke@campus.tu-berlin.de

Tacettin Emre Bök*
5U AI
Munich, Germany
tacettinemre1@gmail.com

Matthias Boehm
Ziawasch Abedjan
BIFOLD & TU Berlin
Berlin, Germany
matthias.boehm@tu-berlin.de
abedjan@tu-berlin.de

Begüm Demir
Volker Markl
BIFOLD & TU Berlin
Berlin, Germany
demir@tu-berlin.de
volker.markl@tu-berlin.de

ABSTRACT

Foundation models (FMs) and their task-specific variants are increasingly forming large and heterogeneous collections that resemble *model lakes*. Managing such lakes requires more than hosting models: users must be able to discover relevant models, compare them using reproducible evidence, and operationalize selected models for downstream use. We present EarthLake, a model lake system for Earth Observation (EO) FMs. The system integrates a schema-guided model registry, task-driven model discovery, and an experiment management framework for benchmarking and comparing candidate FMs. In the demo, participants explore the model registry, discover FMs using natural language task descriptions, benchmark selected FMs on EO datasets, compare experiment results, and execute chosen FMs on new imagery. Our interactive demonstration illustrates how EarthLake supports end-to-end workflows for discovering, evaluating, and operating EO FMs.

PVLDB Reference Format:
Binger Chen, Haralampos Gavriilidis, Luca Gaedicke, Tacettin Emre Bök, Matthias Boehm, Ziawasch Abedjan, Begüm Demir, and Volker Markl. Demonstrating EarthLake: A Model Lake System for Earth Observation Foundation Model Management. PVLDB, 19(12): 4610 - 4613, 2026.
doi:10.14778/3827998.3828078

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/polydbms/earthlake/.

1 INTRODUCTION

When Model Repositories Become Model Lakes. Foundation models (FMs) and their task-specific variants proliferate rapidly across domains such as computer vision, natural language processing, and Earth observation (EO), where diverse sensors, tasks, and operational constraints make systematic model management particularly important [6, 10, 13]. As EO organizations accumulate large collections of heterogeneous FMs, these repositories increasingly resemble what recent data management research introduced as *model lakes* [11]: systems that manage models with their metadata, benchmarks, and related artifacts. However, current model repositories mainly support storage and browsing [7], leaving end-to-end model *discovery*, *evaluation*, and *operation* largely manual, which does not scale as model collections grow [3]. Given an EO task (e.g., flood mapping) and operational constraints, e.g., input data modality, available hardware, users must identify feasible models, evaluate them under consistent settings, and operationalize them. At model lake scale, this becomes a data management problem; prior work on *model-zoo search* shows that manual browsing does not scale [9].

From Task to Operational Model. Consider an EO analyst who must generate flood extent maps shortly after a storm using limited computational resources, e.g., a server with a single GPU. The analyst must navigate a search space of hundreds of EO FMs and variants to identify those compatible with the available data modality, often synthetic aperture radar (SAR) imagery under cloud cover [13]. Candidate models must be filtered by hardware constraints, adapted using limited labeled data, and evaluated under consistent preprocessing and tiling configurations for meaningful comparison. Finally, the selected model must be deployed so analysts can upload new scenes and generate flood maps. In practice, these steps are often manual using a combination of model hubs, scripts, and experimentation frameworks, making the workflow repetitive, difficult to reproduce, and hard to operationalize [1, 14].

A Model Lake Management System. To address the challenge, we introduce EarthLake, a model lake system for EO FMs. EarthLake manages models, metadata, experiments, and evaluation results in a unified framework. Its core is a schema-guided *model registry* that represents EO FMs as structured records with their capabilities, input requirements, and operational constraints. This schema enables systematic querying and comparing of heterogeneous models in the lake, making these properties searchable rather than implicit in scripts or documentation [8]. Using this registry, *REMSA* [2], our recent work on task-driven model selection, interactively maps natural language task descriptions and constraints

*Work partially done while the author was at BIFOLD and TU Berlin.

Proceedings of the VLDB Endowment, Vol. 19, No. 12 ISSN 2150-8097.
doi:10.14778/3827998.3828078

to candidate models in the lake. EarthLake then enables users to evaluate shortlisted models via reproducible benchmarking workflows and operationalize the selected model for downstream use. Together, these capabilities turn the model lake into a unified system for *discover-evaluate-operate* workflows over EO FMs.

Demonstration Overview. The demonstration mirrors the workflow of an EO data analyst who needs to identify and deploy models for tasks such as flood mapping. Starting from a task description, users explore candidate models in the model lake conversationally, obtain task-driven recommendations, and interactively evaluate shortlisted models. Users can observe model performance, then deploy and interact with the model on new imagery. The demonstration shows how EarthLake integrates FM discovery, benchmarking, and operationalization in a single model lake system.

2 BACKGROUND AND MOTIVATION

Running Example: Flood Mapping. We use flood mapping to illustrate the model selection challenges in an EO model lake. After a major storm, analysts must rapidly generate flood maps from satellite imagery. Multiple FMs may support this task, but their assumptions and operational requirements vary. Some models support optical imagery [10], while others support cloud-penetrating SAR data [6]. Models also differ in input resolution, preprocessing pipelines, runtime environments, and hardware requirements. To update or deploy a flood-mapping pipeline, analysts must identify feasible models, adapt them to local datasets with limited labeled samples, and compare performance under consistent configurations. They must consider operational constraints such as GPU memory, inference latency, and throughput. With many FMs in a repository, performing this process manually becomes increasingly difficult.

Managing Models Beyond Weights. At model lake scale, managing models becomes a data management problem. The objects that must be managed include not only model weights, but also model metadata, input assumptions, preprocessing pipelines, runtime environments, benchmarking configurations, and evaluation results. Managing all this information is essential for reproducibility and reuse. Previous work in model management and experiment tracking emphasizes the importance of recording datasets, configurations, metrics, and artifacts generated during experimentation [12]. Similarly, model repositories and registries organize models and their metadata to support discovery and operational deployment [7]. However, these systems typically focus on storing models or tracking experiments, rather than enabling repository-scale comparison and selection across heterogeneous models.

Toward End-to-End Model Lake Workflows. Despite these advances, model selection and validation remain fragmented. Model metadata is often scattered among papers, model cards, and code repositories, making it difficult to identify feasible models for a task [11]. Operational constraints such as supported modalities, memory requirements, or runtime dependencies are rarely structured and queryable. Thus, analysts repeatedly reconstruct feasibility checks and benchmarking pipelines during evaluation. Benchmark results are also difficult to reuse because they lack the configuration and hardware context for meaningful comparisons. These challenges motivate model lake systems supporting end-to-end workflows that connect *model discovery*, *benchmarking*, and *operational model use*. To address this, EarthLake integrates schema-guided model management, task-driven model selection, and experiment management in a unified EO model lake, enabling interactive model discovery, evaluation, and deployment in a single workflow.

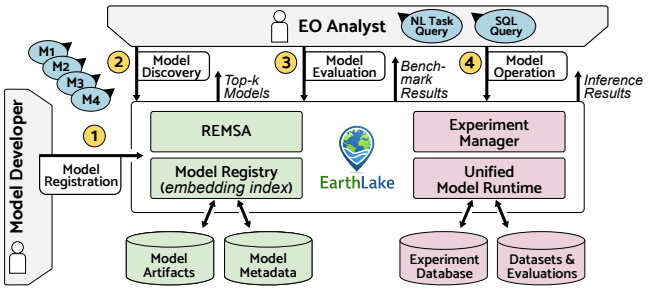

Figure 1: EarthLake Architecture.

3 EARTHLAKE SYSTEM ARCHITECTURE

In this section, we present EarthLake's architecture for managing EO FMs and related artifacts in a unified model lake.

3.1 A Model Lake for EO FMs

EarthLake manages EO FMs in a *model lake*, building on the recent model lake vision [11]. It treats models and associated artifacts as first-class managed assets in a unified repository. The lake stores heterogeneous artifacts: model weights, metadata, benchmarking configurations, and experimental results. Models and associated information are organized through system components supporting discovery, evaluation, and operational use. Concretely, the model lake contains four asset classes: (i) *model artifacts*, including weights, code, and runtime dependencies, (ii) *model metadata*, stored in our structured registry, (iii) *experiment database*, capturing benchmarking configurations and results, and (iv) *datasets and evaluation outputs*. These artifacts together form a persistent repository enabling systematic model discovery, comparison, and reuse.

3.2 Architecture & End-to-End Workflow

Figure 1 illustrates EarthLake's four-stage end-to-end workflow: ① **Model Registration.** Models enter the lake through registration by model developers or administrators. Developers upload model artifacts and documentation, e.g., research papers or model cards. EarthLake automatically extracts metadata from these heterogeneous sources and maps them to the registry schema, creating structured records that describe model capabilities, input data requirements, and operational constraints[1]. These records populate the *Model Registry* and become searchable assets in the model lake. ② **Model Discovery.** EO analysts can search the model lake using either natural language or SQL queries. They specify a task description and optional constraints, such as input modality or hardware limits. *REMSA*, our discovery module, retrieves candidates from the registry and returns ranked top-k models with relevant metadata and rationale for their task compatibility and capabilities.

[1]The complete JSON schema is available in the `fm.schema.json` file in our repository.

③ **Model Evaluation.** From the recommended shortlist, analysts can select models to evaluate on their datasets. The *Experiment Manager* orchestrates benchmarking via the *Unified Model Runtime*, which runs heterogeneous models under consistent configurations. Evaluation metrics and execution context are recorded in the *Experiment Database*. EarthLake compares the performance of candidates on user tasks.

④ **Model Deployment.** After selecting a model based on recommendations and evaluation results, analysts can deploy the validated configuration for operational use. Via the *Unified Model Runtime*, the selected model can run on new data, producing task inference results.

3.3 Core System Components

Model Registry and Embedding Index. The *Model Registry* serves as the metadata backbone of the model lake. It stores structured information of each model, including supported tasks, data modalities, input requirements, architecture, and operational constraints, providing a unified basis for model discovery over metadata otherwise spread across repositories, model cards, and documentation. For semantic task-driven search, EarthLake maintains an embedding index over textual metadata such as model descriptions and documentation with the relational registry. These structures together provide repository-scale model catalog access, enabling models to be retrieved efficiently based on semantic similarity to a task description and then filtered and ranked by structured metadata stored in the registry. Because model metadata is in a relational schema, users can explore and select models through SQL queries. Currently, EarthLake hosts over 160 EO FMs and their variants.

REMSA: Task-Driven Model Discovery. The *REMSA* module supports task-driven discovery over the *Model registry*. While the registry exposes directly queryable structured metadata, users often express needs as high-level task descriptions and operational constraints rather than schema fields. REMSA acts as a conversational semantic interface over the registry, combining semantic retrieval with structured metadata filtering. This enables users to identify feasible models without manually exploring large repositories. Given a natural language task description and optional constraints, REMSA connects user intent with semantic model similarity and explicit capabilities and constraints in the registry. It interprets task, retrieves candidates via the embedding index, and ranks them by semantic relevance and feasibility. Our evaluation shows that REMSA effectively discovers relevant models for diverse EO tasks [2].

Experiment Manager and Unified Runtime. The *Experiment Manager* and *Unified Runtime* support benchmarking and operationalizing selected models, extending our previous work [4, 5]. Since shortlisted models may use different frameworks and be exposed through different execution interfaces, EarthLake provides a model runtime that unifies execution behind a common interface and also serves as the basis for deploying selected models to end users. Each model registers a runtime plug-in that encapsulates model-specific preprocessing and data alignment logic for different input modalities, allowing transparent execution of heterogeneous pipelines via the unified interface. The Experiment Manager records metrics, configurations, and execution context in the Experiment Database. These experiment records become persistent model lake assets, enabling reproducible comparisons and allowing previous benchmark results to be reused for future tasks.

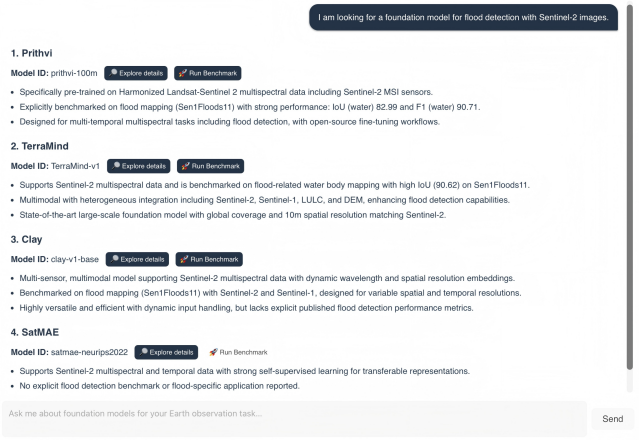

Figure 2: UI for Conversational Model Discovery (REMSA).

Together, these components enable an end-to-end workflow from model discovery to evaluation and operation. By managing models, metadata, and experiment results in a unified system, EarthLake makes the *discover-evaluate-operate* process repeatable across large EO FM collections.

4 DEMONSTRATION OVERVIEW

The demonstration presents an interactive workflow for managing EO FMs in EarthLake. We illustrate the *discover-evaluate-operationalize* workflow using an EO analyst scenario updating a flood-mapping pipeline after a storm. Given new satellite imagery and operational constraints, e.g., task, sensor modality, and available resources, the analyst must find suitable models, evaluate them on labeled data, and select one for operational use. During the demonstration, participants interact with EarthLake through a set of operations showing how models are discovered, inspected, evaluated, and used in practice.

Model Ingestion and Registry Population. The demonstration begins by ingesting a new model into the model lake. A model developer uploads model documentation such as a research paper or model card. EarthLake automatically extracts metadata fields including supported tasks, data modalities, architecture, and training details for the model registry. Extracted values have confidence scores visualized in the UI with color cues. Developers can inspect the automatically generated metadata and correct fields when needed. This interaction shows how the model lake evolves as models are added and how heterogeneous documentation can yield structured metadata.

Task-Driven Model Discovery. Participants explore the model registry to understand the scale and heterogeneity of the model lake. The registry interface presents over 160 EO FMs in a table showing key attributes such as supported tasks, sensor modalities, architectures, and operational constraints. Users can browse the catalog, inspect individual model records, filter models by metadata, and query the relational schema with SQL. While this interface enables

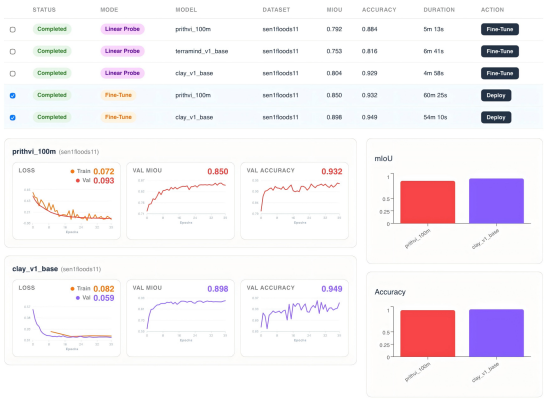

Figure 3: UI for Model Benchmarking and Fine-tuning.

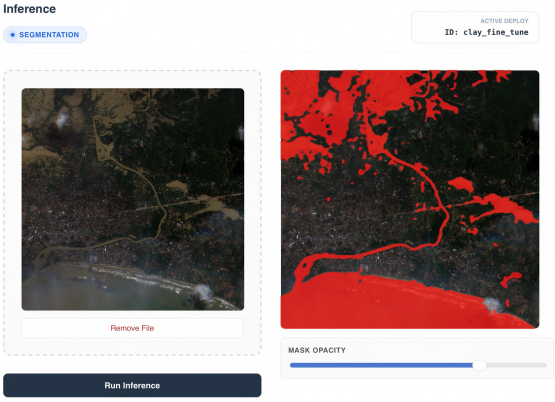

Figure 4: UI for Model Inference on EO tasks.

systematic model lake exploration, participants quickly observe that manually identifying a suitable model becomes difficult as the number of models grows and task requirements become more specific. To address this challenge, users can interact with EarthLake's conversational model discovery interface. Users specify a task and optional operational constraints in natural language, e.g., requesting a model for *flood detection with Sentinel-2 images under limited GPU resources*. EarthLake invokes REMSA to retrieve and rank candidates from the registry using semantic similarity and feasibility constraints. The interface presents a ranked model shortlist with compatibility signals such as task match, modality support, reported performance, and hardware feasibility (Figure 2). Users can inspect model metadata and select models to evaluate.

Benchmarking and Comparing Models. After obtaining a candidate shortlist, users can evaluate them on their datasets to determine which model performs best for their task. The *Experiment Manager* benchmarks candidates on user-provided labeled EO datasets. The benchmarking interface allows users to select models, upload datasets, and configure evaluation settings. EarthLake supports two evaluation modes. In *linear probing*, models are evaluated without parameter updates, enabling quick pretrained representations comparison. In *fine-tuning*, models are adapted with a small labeled dataset to assess task-specific performance. EarthLake executes all benchmarks via the unified model runtime, ensuring consistent execution across heterogeneous model implementations. Afterward, EarthLake presents results in an interactive comparison view summarizing performance metrics across candidates. Metrics such as task accuracy, segmentation accuracy (Mean Intersection over Union – mIoU), and inference latency are displayed side-by-side (Figure 3). This interface allows users to explore trade-offs between model performance and operational cost when selecting a model for deployment. Experiment configurations, datasets, and hardware contexts are stored in the experiment database, enabling result revisiting and comparing across runs. This demonstrates how experiment records become reusable evidence in the model lake.

Running Inference on New Imagery. Finally, users select and deploy the best-performing model. EarthLake loads the validated model and performs inference via the unified runtime interface. Users can upload satellite imagery and execute the target task (e.g., segmentation/classification) with the selected model, with predictions shown in the interface (Figure 4). This shows how a selected model moves from evaluation to operational in the same system.

Demonstration Summary. The demonstration highlights three key capabilities of EarthLake: (i) schema-guided model registry for structured exploration and querying of large model collections, (ii) integrated experiment management for reproducible benchmarking and candidate models comparison, and (iii) end-to-end workflow connecting model discovery, evaluation, and operation.

ACKNOWLEDGMENTS

This work is supported by the European Research Council Project Agent-BigEarth (No. 101292498) and the European Union Horizon Project PROTEUS (No. 101296397). Haralampos Gavriilidis was partially supported by a scholarship of the German Academic Exchange Service (DAAD).

REFERENCES

[1] Andrew Chen, Andy Chow, et al. 2020. Developments in MLflow: A System to Accelerate the Machine Learning Lifecycle. In *DEEM@SIGMOD*. 5:1–5:4.
[2] Binger Chen, Tacettin Emre Bök, et al. 2026. Foundation Model Selection for Remote Sensing via a Constraint-Aware Agent. In *ECCV*.
[3] Behrouz Derakhshan, Alireza Rezaei Mahdiraji, et al. 2020. Optimizing Machine Learning Workloads in Collaborative Environments. In *SIGMOD*.
[4] Gereon Dusella, Haralampos Gavriilidis, et al. 2025. Automating Earth Observation Analytics Pipelines with Agent Raven. *Proc. Big Data from Space (BiDS)*.
[5] Gereon Dusella, Haralampos Gavriilidis, et al. 2026. Benchmarking Spatial Operations over Heterogeneous Data: The Case of Zonal Statistics. In *DBTest*.
[6] Xin Guo, Jiangwei Lao, et al. 2024. Skysense: A multi-modal remote sensing foundation model towards universal interpretation for earth observation imagery. In *CVPR*. 27672–27683.
[7] Hugging Face. 2026. Hugging Face Hub Documentation. https://huggingface.co/docs/hub/en/index. Accessed: 2026-02-21.
[8] Ziyu Li, Henk Kant, et al. 2023. Metadata Representations for Queryable Repositories of Machine Learning Models. *IEEE Access* 11 (2023), 125616–125630.
[9] Ziyu Li, Hilco van der Wilk, et al. 2024. Model Selection with Model Zoo via Graph Learning. In *ICDE*. 1296–1309.
[10] Matías Mendieta, Boran Han, et al. 2023. Towards geospatial foundation models via continual pretraining. In *CVPR*. 16806–16816.
[11] Koyena Pal, David Bau, and Renée J. Miller. 2025. Model Lakes. In *EDBT*. 985–995.
[12] Manasi Vartak and Samuel Madden. 2018. MODELDB: Opportunities and Challenges in Managing Machine Learning Models. *IEEE Data Engineering Bulletin* 41, 4 (2018), 16–25.
[13] Aoran Xiao, Weihao Xuan, et al. 2025. Foundation models for remote sensing and earth observation: A survey. *IEEE Geoscience and Remote Sensing Magazine*.
[14] Matei Zaharia, Andrew Chen, et al. 2018. Accelerating the Machine Learning Lifecycle with MLflow. *IEEE Data Engineering Bulletin* 41, 4 (2018), 39–45.