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

Binger Chen, Haralampos Gavriilidis, Luca Gaedicke, Tacettin Emre Bök,
Matthias Boehm, Ziawasch Abedjan, Begüm Demir, Volker Markl
BIFOLD and TU Berlin
Berlin, Germany
{chen,gavriilidis,luca.gaedicke,boek,matthias.boehm,abedjan,demir,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, Volker Markl. Demonstrating EarthLake: A Model Lake System for Earth Observation Foundation Model Management. PVLDB, 14(1): XXX-XXX, 2026.
doi:XX.XX/XXX.XX

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 are rapidly proliferating 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 [4, 8, 11]. As EO organizations accumulate large collections of heterogeneous FMs, these repositories increasingly resemble what recent data management research introduced as *model lakes* [9]: systems that manage models together with their metadata, benchmarks, and related artifacts. However, current model repositories mainly support storage and browsing of models [5], leaving the end-to-end process of *discovering*, *evaluating*, and *operating* models 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 still identify feasible models, evaluate them under consistent settings, and operationalize the selected model. At model lake scale, this becomes a data management problem; prior work on *model-zoo search* likewise shows that manual browsing does not scale [7].

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 equipped with a single GPU. The analyst must navigate a search space of hundreds of EO FMs and their variants to identify those compatible with the available data modality, often synthetic aperture radar (SAR) imagery under cloud cover [11]. Candidate models must be filtered by hardware constraints, adapted using limited labeled data, and evaluated under consistent preprocessing and tiling configurations to enable 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 performed manually using a combination of model hubs, scripts, and experimentation frameworks, making the workflow repetitive, difficult to reproduce, and hard to operationalize [1, 12].

A Model Lake Management System. To address this challenge, we introduce EarthLake, a model lake system for EO FMs. EarthLake manages models, metadata, experiments, and evaluation results within a unified framework. At its core is a schema-guided *model registry* that represents EO FMs as structured records describing their capabilities, input requirements, and operational constraints. This schema enables systematic querying and comparison of heterogeneous models within the lake, making these properties searchable rather than remaining implicit in scripts or documentation [6]. On top of this registry, *REMSA* [2], our recent work on task-driven model selection, interactively maps natural language task descriptions and constraints to candidate models in the lake. EarthLake then enables users to evaluate shortlisted models through reproducible benchmarking workflows and to 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

This work is supported by the European Research Council Project Agent-BigEarth (No. 101292498) and the European Union Horizon Project PROTEUS (No. 101296397).

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

for a task 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 and finally deploy and interact with the selected model on new imagery. The demonstration illustrates how EarthLake integrates FM model discovery, benchmarking, and operationalization within 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 major storm events, analysts must rapidly generate flood maps from satellite imagery. Multiple FMs may support this task, but their assumptions and operational requirements vary. Some models are designed for optical imagery [8], while others support SAR data that can penetrate cloud cover [4]. 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 candidate models, adapt them to local datasets using limited labeled samples, and compare their performance under consistent configurations. They must also consider operational constraints such as GPU memory, inference latency, and throughput. When many models are available 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 [10]. Similarly, model repositories and registries organize models and their metadata to support discovery and operational deployment [5]. 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, the process of selecting and validating models remains fragmented. Model metadata is often scattered across papers, model cards, and code repositories, making it difficult to identify feasible candidates for a given task [9]. Operational constraints such as supported modalities, memory requirements, or runtime dependencies are rarely represented in a structured and queryable form. As a result, analysts repeatedly reconstruct feasibility checks and benchmarking pipelines when evaluating models. Benchmark results are also difficult to reuse because they are not stored together with the configuration and hardware context needed for meaningful comparisons. These challenges highlight the need for model lake systems that support end-to-end workflows connecting *model discovery*, *benchmark evaluation*, and *operational model use*. To address this, EarthLake integrates schema-guided model management, task-driven model selection, and experiment management within a unified EO model lake, enabling users to interactively discover, evaluate, and deploy models in a single workflow.

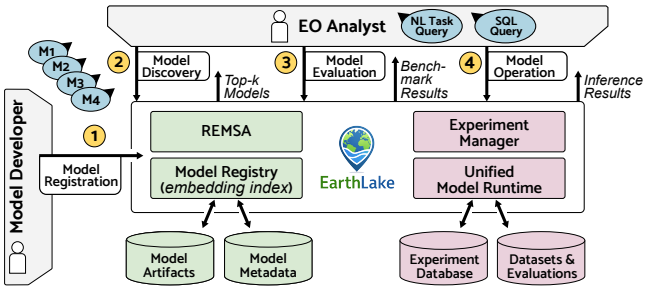

Figure 1: EarthLake Architecture.

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 within a *model lake*, building on the recent data management vision of model lakes [9]. It treats models and their associated artifacts as first-class managed assets within a unified repository. The lake stores heterogeneous artifacts, including model weights, metadata, benchmarking configurations, and experimental results. Models and their associated information are organized through a set of system components that support discovery, evaluation, and operational use. More concretely, the model lake contains four main classes of assets: (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 that enables 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 a registration process initiated by model developers or administrators. Developers upload model artifacts and documentation, such as 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 within the model lake.

(2) **Model Discovery.** EO analysts can search the model lake by submitting either natural language task queries or SQL queries. These queries specify a task description and optional constraints, such as input modality or hardware limits. Then, *REMSA*, our discovery module, retrieves candidates from the Model Registry and returns a ranked top-k list of recommended models together with relevant metadata and rationale for their capabilities and compatibility with the specified task.

[1]The complete JSON schema is available at: https://github.com/polydbms/earthlake/blob/main/fm.schema.json

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

④ **Model Deployment.** After selecting a model based on the recommendations and the evaluation results, analysts can deploy the validated model configuration for operational use. Through the unified model runtime, the selected model can be applied to new input data, producing inference results for the target task.

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 describing each model, including supported tasks, data modalities, input requirements, architecture information, and operational constraints, providing a unified basis for model discovery over metadata that is otherwise spread across repositories, model cards, and documentation. To support semantic task-driven search, the system maintains an embedding index over selected textual metadata fields such as model descriptions and documentation alongside the relational registry. These structures together provide repository-scale access to the model catalog, enabling candidate models to be retrieved efficiently based on semantic similarity to a task description and then filtered and ranked using the structured metadata stored in the registry. Because model metadata is stored in a relational schema, users can also explore and select models through SQL queries over the registry. In our current deployment, EarthLake hosts more than 160 EO FMs and their variants.

REMSA: Task-Driven Model Discovery. The *REMSA* module provides task-driven model discovery over the model registry. While the registry exposes structured metadata that can be queried directly, users often express their needs in terms of high-level task descriptions and operational constraints rather than specific schema fields. REMSA acts as a conversational semantic query interface over the model registry, which combines 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 both semantic model similarity and the explicit capabilities and constraints represented in the registry. It interprets task intent, retrieves candidate models through the embedding index, and ranks them according to semantic relevance and feasibility. Our evaluation showed that this semantic retrieval approach 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. Since shortlisted models may be implemented in different frameworks and exposed through different execution interfaces, EarthLake provides a Unified Model Runtime that places 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 required for different input modalities, allowing heterogeneous pipelines to be executed transparently through the unified interface. The Experiment Manager records metrics, configurations, and execution context in the Experiment Database. These experiment records become persistent assets of the model lake, enabling reproducible comparisons and allowing benchmark results from previous runs to be reused when evaluating models for future tasks.

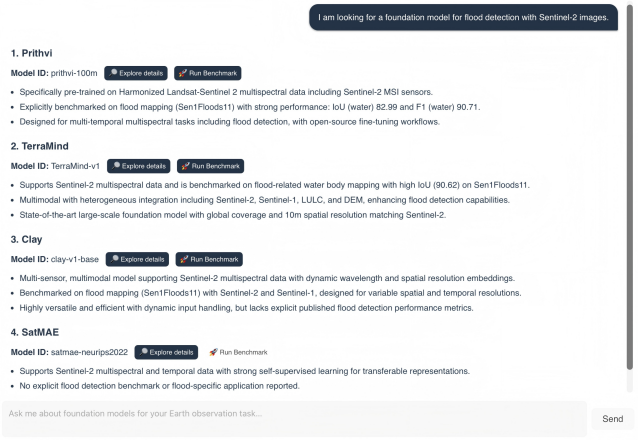

Figure 2: UI for Conversational Model Discovery (REMSA).

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 this *discover-evaluate-operate* process repeatable across large collections of EO FMs.

4 DEMONSTRATION OVERVIEW

The demonstration presents an interactive workflow for managing EO FMs within EarthLake. We illustrate the *discover-evaluate-operationalize* workflow through the scenario of an EO analyst updating a flood-mapping pipeline after a storm event. Given newly available satellite imagery and operational constraints, e.g., task description, sensor modality, and available resources, the analyst must identify suitable models, evaluate their performance on labeled data, and select a model for operational use. During the demonstration, participants interact with EarthLake through a sequence of operations that illustrate how models are discovered, inspected, evaluated, and ultimately used in practice.

Model Ingestion and Registry Population. The demonstration begins with the ingestion of 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 information, and training details to populate the model registry. Extracted values are associated with confidence scores that are visualized in the interface using color cues. Model developers can inspect the automatically generated metadata and correct fields when necessary. This interaction illustrates how the model lake evolves as new models are added and how structured metadata can be generated from heterogeneous documentation.

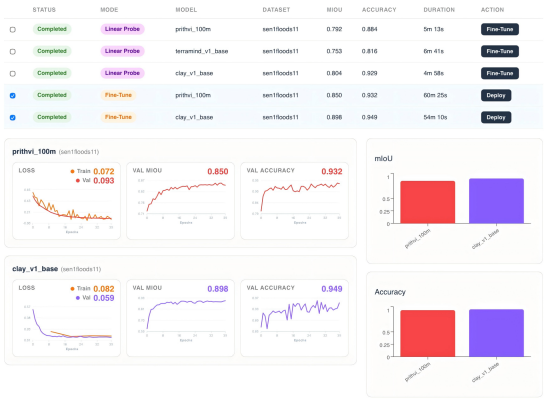

Figure 3: UI for Model Benchmarking and Fine-tuning.

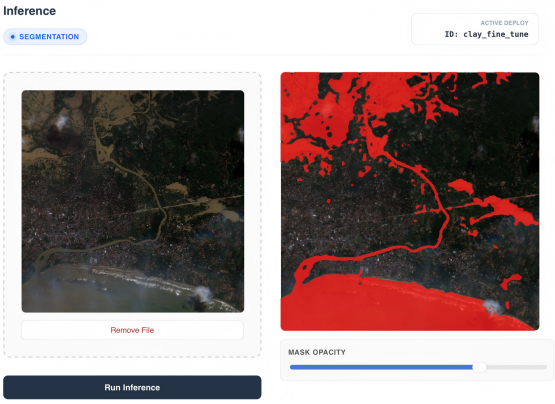

Figure 4: UI for Model Inference on EO tasks.

Task-Driven Model Discovery. Participants first explore the model registry to understand the scale and heterogeneity of the model lake. The registry interface presents more than 160 EO FMs in a tabular view showing key attributes such as supported tasks, sensor modalities, architectures, and operational constraints. Users can browse the catalog, inspect individual model records, filter models using metadata fields, and issue SQL queries over the relational schema. While this interface enables systematic exploration of the model lake, 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 the conversational model discovery interface of EarthLake. Users specify a task description and optional operational constraints using natural language, for example, requesting a model for *flood detection with Sentinel-2 images under limited GPU resources*. EarthLake invokes REMSA to retrieve and rank candidate models from the registry using both semantic similarity and feasibility constraints. The interface presents a ranked shortlist of models together with compatibility signals such as task match, modality support, reported performance, and hardware feasibility (Figure 2). Users can inspect each model's metadata and select models for further evaluation.

Benchmarking and Comparing Models. After obtaining a shortlist of candidate models, users can evaluate them on their datasets to determine which model performs best for their task. Then, the Experiment Manager benchmarks selected models 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 updating model parameters, enabling quick comparison of pretrained representations. In *fine-tuning*, models are adapted using a small labeled dataset to assess task-specific performance. EarthLake executes all benchmarking runs through the unified model runtime, which ensures consistent execution across heterogeneous model implementations.

After execution, EarthLake presents the results in an interactive comparison view summarizing performance metrics across candidate models. 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. All experiment configurations, datasets, and hardware contexts are recorded in the experiment database, enabling users to revisit and compare results across runs. This capability demonstrates how experiment records become reusable evidence within the model lake.

Running Inference on New Imagery. Finally, users select the best performing model and deploy it. EarthLake loads the validated model configuration and performs inference through the unified runtime interface. Users can upload new satellite imagery and directly execute the target task (e.g., segmentation or classification) with the selected model, with the predictions shown in the interface (Figure 4). This step shows how a selected model can be taken from evaluation to operational use within the same system.

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

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. REMSA: Foundation Model Selection for Remote Sensing via a Constraint-Aware Agent. arXiv:2511.17442 [cs.CV]

[3] Behrouz Derakhshan, Alireza Rezaei Mahdiraji, et al. 2020. Optimizing Machine Learning Workloads in Collaborative Environments. In *SIGMOD*.

[4] 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.

[5] Hugging Face. 2026. Hugging Face Hub Documentation. https://huggingface.co/docs/hub/en/index. Accessed: 2026-02-21.

[6] Ziyu Li, Henk Kant, et al. 2023. Metadata Representations for Queryable Repositories of Machine Learning Models. *IEEE Access* 11 (2023), 125616–125630.

[7] Ziyu Li, Hilco van der Wilk, et al. 2024. Model Selection with Model Zoo via Graph Learning. In *ICDE*. 1296–1309.

[8] Matías Mendieta, Boran Han, et al. 2023. Towards geospatial foundation models via continual pretraining. In *CVPR*. 16806–16816.

[9] Koyena Pal, David Bau, and Renée J. Miller. 2025. Model Lakes. In *EDBT*. 985–995.

[10] Manasi Vartak and Samuel Madden. 2018. MODELDB: Opportunities and Challenges in Managing Machine Learning Models. *IEEE Data Engineering Bulletin* 41, 4 (2018), 16–25.

[11] Aoran Xiao, Weihao Xuan, et al. 2025. Foundation models for remote sensing and earth observation: A survey. *IEEE Geoscience and Remote Sensing Magazine* (2025).

[12] Matei Zaharia, Andrew Chen, et al. 2018. Accelerating the Machine Learning Lifecycle with MLflow. *IEEE Data Engineering Bulletin* 41, 4 (2018), 39–45.