Open Models Hub

16th October 2025

Cristiano Baldassi
Scientific & Data-Intensive Computing

Advanced Data Management
2024-2025

Introduction

Objective: Enable researchers to efficiently manage, discover, replicate, and reuse machine learning assets, while also tracking computational costs and environmental impact.

Guiding Principles:

- **FAIR Data**: Findable, Accessible, Interoperable, Reusable assets
- **Model Reproducibility:** Ensure experiments and results can be reliably reproduced
- **Resource Awareness**: Mandatory tracking of computational costs and environmental impact

Data Modeling

Conceptual Model

The main entities of the data model can be cast in four categories:

- **ML Assets**
- **Experiment Support**
- **Resource Management**
- **Research Context**

Conceptual Model: Entities

Machine Learning Assets:

- ***MLAsset (Base)***: Common properties shared by all ML assets (ID, metadata, versioning)
  - ***Model***: Trained machine learning models
  - ***Dataset***: Collections of data used for training, validation, or testing
  - ***Experiment***: Complete training runs that produce models from datasets

Research Context:

- ***Researcher***: Individual contributors with expertise and affiliations
- ***Organization***: Institutions that provide resources and governance

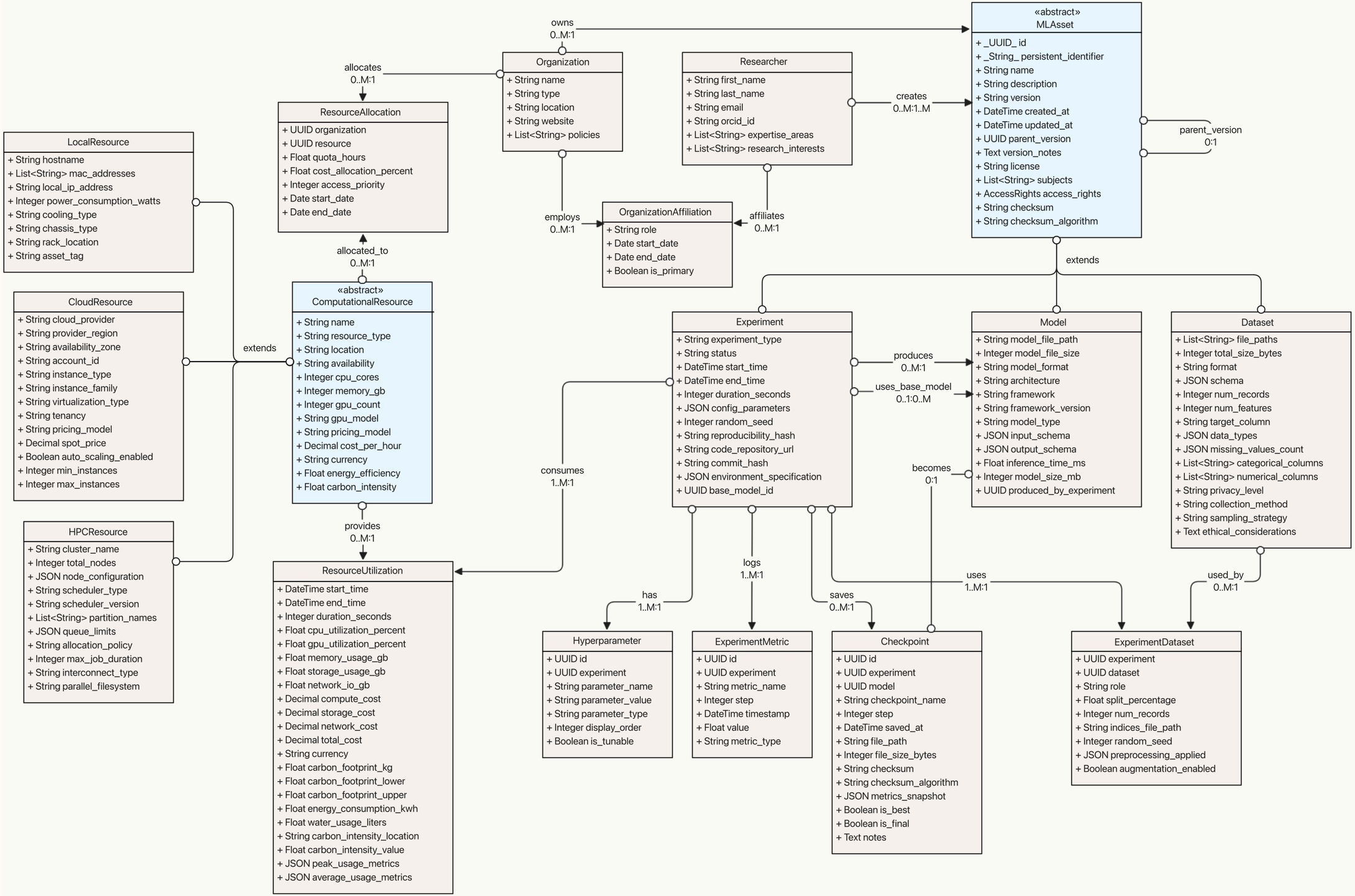
Conceptual Model: Entities

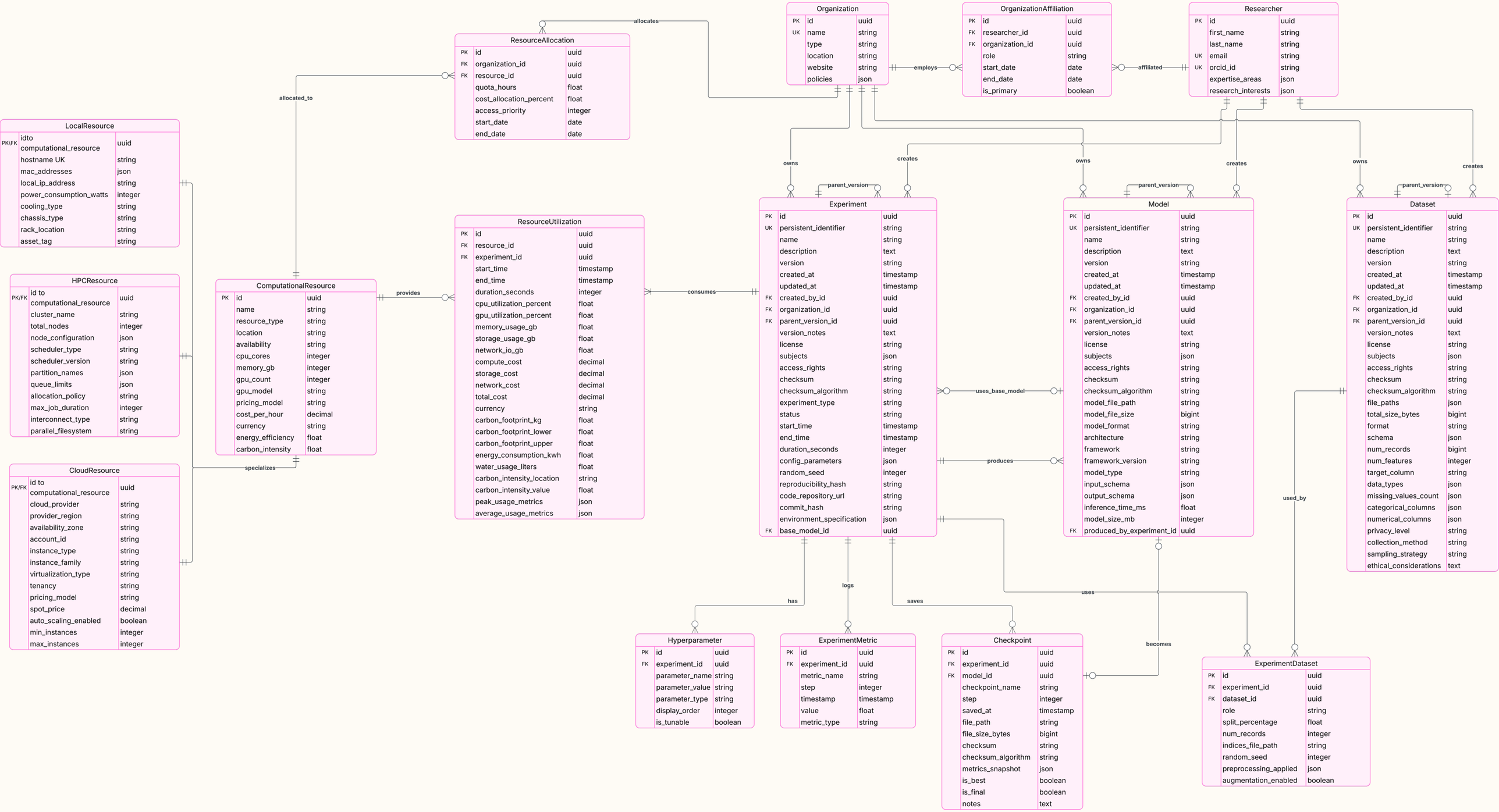
Experiment Support

- ***Hyperparameter***: Individual experiment configuration parameters
- ***ExperimentMetric***: Time-series metrics logged during training
- ***Checkpoint***: Saved model states during training for resume and rollback

Resource Management:

- ***ComputationalResource***: Base class for computing infrastructure
  - ***LocalResource***: On-premises hardware
  - ***CloudResource***: Cloud provider instances
  - ***HPCResource***: High-performance computing cluster resources
- ***ResourceUtilization***: usage metrics with integrated cost and environmental tracking


owns
0..M:1
«abstract»
MLAsset
+ _UUID_ id
+ _String_ persistent_identifier
+ String name
+ String description
+ String version
+ DateTime created_at
+ DateTime updated_at
+ UUID parent_version
+ Text version_notes
+ String license
+ List<String> subjects
+ AccessRights access_rights
+ String checksum
+ String checksum_algorithm
parent_version
0:1
allocates
0..M:1
Organization
+ String name
+ String type
+ String location
+ String website
+ List<String> policies
Researcher
+ String first_name
+ String last_name
+ String email
+ String orcid_id
+ List<String> expertise_areas
+ List<String> research_interests
creates
0..M:1..M
ResourceAllocation
+ UUID organization
+ UUID resource
+ Float quota_hours
+ Float cost_allocation_percent
+ Integer access_priority
+ Date start_date
+ Date end_date
LocalResource
+ String hostname
+ List<String> mac_addresses
+ String local_ip_address
+ Integer power_consumption_watts
+ String cooling_type
+ String chassis_type
+ String rack_location
+ String asset_tag
employs
0..M:1
OrganizationAffiliation
+ String role
+ Date start_date
+ Date end_date
+ Boolean is_primary
affiliates
0..M:1
allocated_to
0..M:1
extends
CloudResource
+ String cloud_provider
+ String provider_region
+ String availability_zone
+ String account_id
+ String instance_type
+ String instance_family
+ String virtualization_type
+ String tenancy
+ String pricing_model
+ Decimal spot_price
+ Boolean auto_scaling_enabled
+ Integer min_instances
+ Integer max_instances
«abstract»
ComputationalResource
+ String name
+ String resource_type
+ String location
+ String availability
+ Integer cpu_cores
+ Integer memory_gb
+ Integer gpu_count
+ String gpu_model
+ String pricing_model
+ Decimal cost_per_hour
+ String currency
+ Float energy_efficiency
+ Float carbon_intensity
provides
0..M:1
HPCResource
+ String cluster_name
+ Integer total_nodes
+ JSON node_configuration
+ String scheduler_type
+ String scheduler_version
+ List<String> partition_names
+ JSON queue_limits
+ String allocation_policy
+ Integer max_job_duration
+ String interconnect_type
+ String parallel_filesystem
ResourceUtilization
+ DateTime start_time
+ DateTime end_time
+ Integer duration_seconds
+ Float cpu_utilization_percent
+ Float gpu_utilization_percent
+ Float memory_usage_gb
+ Float storage_usage_gb
+ Float network_io_gb
+ Decimal compute_cost
+ Decimal storage_cost
+ Decimal network_cost
+ Decimal total_cost
+ String currency
+ Float carbon_footprint_kg
+ Float carbon_footprint_lower
+ Float carbon_footprint_upper
+ Float energy_consumption_kwh
+ Float water_usage_liters
+ String carbon_intensity_location
+ Float carbon_intensity_value
+ JSON peak_usage_metrics
+ JSON average_usage_metrics
consumes
1..M:1
extends
Experiment
+ String experiment_type
+ String status
+ DateTime start_time
+ DateTime end_time
+ Integer duration_seconds
+ JSON config_parameters
+ Integer random_seed
+ String reproducibility_hash
+ String code_repository_url
+ String commit_hash
+ JSON environment_specification
+ UUID base_model_id
produces
0..M:1
uses_base_model
0..1:0..M
Model
+ String model_file_path
+ Integer model_file_size
+ String model_format
+ String architecture
+ String framework
+ String framework_version
+ String model_type
+ JSON input_schema
+ JSON output_schema
+ Float inference_time_ms
+ Integer model_size_mb
+ UUID produced_by_experiment
becomes
0:1
Dataset
+ List<String> file_paths
+ Integer total_size_bytes
+ String format
+ JSON schema
+ Integer num_records
+ Integer num_features
+ String target_column
+ JSON data_types
+ JSON missing_values_count
+ List<String> categorical_columns
+ List<String> numerical_columns
+ String privacy_level
+ String collection_method
+ String sampling_strategy
+ Text ethical_considerations
has
1..M:1
logs
1..M:1
saves
0..M:1
uses
1..M:1
used_by
0..M:1
Hyperparameter
+ UUID id
+ UUID experiment
+ String parameter_name
+ String parameter_value
+ String parameter_type
+ Integer display_order
+ Boolean is_tunable
ExperimentMetric
+ UUID id
+ UUID experiment
+ String metric_name
+ Integer step
+ DateTime timestamp
+ Float value
+ String metric_type
Checkpoint
+ UUID id
+ UUID experiment
+ UUID model
+ String checkpoint_name
+ Integer step
+ DateTime saved_at
+ String file_path
+ Integer file_size_bytes
+ String checksum
+ String checksum_algorithm
+ JSON metrics_snapshot
+ Boolean is_best
+ Boolean is_final
+ Text notes
ExperimentDataset
+ UUID experiment
+ UUID dataset
+ String role
+ Float split_percentage
+ Integer num_records
+ String indices_file_path
+ Integer random_seed
+ JSON preprocessing_applied
+ Boolean augmentation_enabled


Organization
PK id uuid
UK name string
type string
location string
website string
policies json
OrganizationAffiliation
PK id uuid
FK researcher_id uuid
FK organization_id uuid
role string
start_date date
end_date date
is_primary boolean
Researcher
PK id uuid
first_name string
last_name string
UK email string
UK orcid_id string
expertise_areas json
research_interests json
employs
affiliated
allocates
ResourceAllocation
PK id uuid
FK organization_id uuid
FK resource_id uuid
quota_hours float
cost_allocation_percent float
access_priority integer
start_date date
end_date date
allocated_to
LocalResource
PK\FK idto computational_resource uuid
hostname UK string
mac_addresses json
local_ip_address string
power_consumption_watts integer
cooling_type string
chassis_type string
rack_location string
asset_tag string
HPCResource
PK/FK id to computational_resource uuid
cluster_name string
total_nodes integer
node_configuration json
scheduler_type string
scheduler_version string
partition_names json
queue_limits json
allocation_policy string
max_job_duration integer
interconnect_type string
parallel_filesystem string
CloudResource
PK/FK id to computational_resource uuid
cloud_provider string
provider_region string
availability_zone string
account_id string
instance_type string
instance_family string
virtualization_type string
tenancy string
pricing_model string
spot_price decimal
auto_scaling_enabled boolean
min_instances integer
max_instances integer
specializes
ComputationalResource
PK id uuid
name string
resource_type string
location string
availability string
cpu_cores integer
memory_gb integer
gpu_count integer
gpu_model string
pricing_model string
cost_per_hour decimal
currency string
energy_efficiency float
carbon_intensity float
provides
ResourceUtilization
PK id uuid
FK resource_id uuid
FK experiment_id uuid
start_time timestamp
end_time timestamp
duration_seconds integer
cpu_utilization_percent float
gpu_utilization_percent float
memory_usage_gb float
storage_usage_gb float
network_io_gb float
compute_cost decimal
storage_cost decimal
network_cost decimal
total_cost decimal
currency string
carbon_footprint_kg float
carbon_footprint_lower float
carbon_footprint_upper float
energy_consumption_kwh float
water_usage_liters float
carbon_intensity_location string
carbon_intensity_value float
peak_usage_metrics json
average_usage_metrics json
consumes
owns
creates
parent_version
Experiment
PK id uuid
UK persistent_identifier string
name string
description text
version string
created_at timestamp
updated_at timestamp
FK created_by_id uuid
FK organization_id uuid
FK parent_version_id uuid
version_notes text
license string
subjects json
access_rights string
checksum string
checksum_algorithm string
experiment_type string
status string
start_time timestamp
end_time timestamp
duration_seconds integer
config_parameters json
random_seed integer
reproducibility_hash string
code_repository_url string
commit_hash string
environment_specification json
FK base_model_id uuid
uses_base_model
produces
Model
PK id uuid
UK persistent_identifier string
name string
description text
version string
created_at timestamp
updated_at timestamp
FK created_by_id uuid
FK organization_id uuid
FK parent_version_id uuid
version_notes text
license string
subjects json
access_rights string
checksum string
checksum_algorithm string
model_file_path string
model_file_size bigint
model_format string
architecture string
framework string
framework_version string
model_type string
input_schema json
output_schema json
inference_time_ms float
model_size_mb integer
FK produced_by_experiment_id uuid
Dataset
PK id uuid
UK persistent_identifier string
name string
description text
version string
created_at timestamp
updated_at timestamp
FK created_by_id uuid
FK organization_id uuid
FK parent_version_id uuid
version_notes text
license string
subjects json
access_rights string
checksum string
checksum_algorithm string
file_paths json
total_size_bytes bigint
format string
schema json
num_records bigint
num_features integer
target_column string
data_types json
missing_values_count json
categorical_columns json
numerical_columns json
privacy_level string
collection_method string
sampling_strategy string
ethical_considerations text
used_by
has
logs
saves
uses
Hyperparameter
PK id uuid
FK experiment_id uuid
parameter_name string
parameter_value string
parameter_type string
display_order integer
is_tunable boolean
ExperimentMetric
PK id uuid
FK experiment_id uuid
metric_name string
step integer
timestamp timestamp
value float
metric_type string
Checkpoint
PK id uuid
FK experiment_id uuid
FK model_id uuid
checkpoint_name string
step integer
saved_at timestamp
file_path string
file_size_bytes bigint
checksum string
checksum_algorithm string
metrics_snapshot json
is_best boolean
is_final boolean
notes text
becomes
ExperimentDataset
PK id uuid
FK experiment_id uuid
FK dataset_id uuid
role string
split_percentage float
num_records integer
indices_file_path string
random_seed integer
preprocessing_applied json
augmentation_enabled boolean

Metadata

Metadata Framework

OpenModelsHub uses a layered integration approach:

- **Foundation Layer:** Dublin Core, Schema.org, DataCite, DCAT, and PROV-O provide universal metadata for all assets
- **ML Layer:** Croissant, ML-Schema, FAIR4ML, and MEX add domain-specific vocabulary
- **Infrastructure Layer:** HPC Ontology describes computational resources and usage

Metadata Framework: Dublin Core (DCMI Terms)

Description:
A universal metadata standard defining 15 core elements for describing digital resources, widely used for cross-domain interoperability.

Key Vocabulary:
`title, creator, description, date, identifier, publisher, rights, subject`

Use in OpenModelsHub:
Provides foundational metadata for all ML assets (MLAsset base class). Essential for basic resource description and cross-repository interoperability.

Metadata Framework: Schema.org

Description:
A web-oriented metadata vocabulary for structured data, enabling search engines to index and interpret datasets. The Dataset class standardizes descriptions of data collections, distributions, and coverage

Key Classes & Properties:
`Dataset, Person, Organization, name, description, identifier, creator, distribution, contentSize, encodingFormat`

Use in OpenModelsHub:
Facilitates web discoverability. Provides base vocabulary extended by Croissant for ML datasets.

Metadata Framework: DataCite Metadata Schema

Description:
A metadata standard for research data citation, providing required and recommended properties data formally citable and trackable.

Required Properties:
`Identifier, Creator, Title, Publisher, PublicationYear, ResourceType`

Additional Properties:
`Subject, Contributor, Date, RelatedIdentifier, Description, Rights`

Use in OpenModelsHub:
DOI assignment and research citation. Supports dataset and model publication with persistent identifiers.

Metadata Framework: W3C DCAT

Description:
A W3C standard for describing datasets, distributions, and data services, enabling interoperability between web-based data catalogs.

Key Classes & Properties:
`Dataset, Distribution, Catalog, distribution, byteSize, mediaType, keyword, theme, accessRights`

Use in OpenModelsHub:
Dataset cataloging and distribution metadata. Links datasets to downloadable distributions.

Metadata Framework: W3C PROV-O

Description:
A W3C ontology for representing provenance, defining Entities, Activities, and Agents to capture how digital objects are created, modified, and used.

Key Classes & Relationship:
`Entity, Activity, Agent, wasGeneratedBy, used, wasAttributedTo, wasDerivedFrom, wasAssociatedWith, startedAtTime, endedAtTime`

Use in OpenModelsHub:
Records complete provenance chains for ML experiments, linking Datasets/Models to Experiments and generated Models.

Metadata Framework: Croissant

Description:
An MLCommons metadata format extending Schema.org Dataset, capturing dataset structure at file, record and features including transformations and relationships.

Key Classes & Properties:
`RecordSet, Field, Split, url, encodingFormat, recordCount, dataType`

Use in OpenModelsHub:
Represents internal dataset features, labels, splits, types, and preprocessing—enabling automated, reproducible data loading and ML experimentation.

Metadata Framework: ML-Schema

Description:
A W3C Community ontology for ML experiments, defining Experiments, Runs, Algorithms, Models, Data, Evaluations, and HyperParameters, with structured relationships linking datasets, algorithms, runs, models, and evaluations.

Key Classes & Properties:
`Run, Model, Algorithm, Dataset, HyperParameter, hasInput, hasOutput, executes, realizes, implements, hasHyperParameter, definedOn`

Use in OpenModelsHub:
Captures complete ML experiment workflows, connecting Datasets, Algorithms, Runs, Models, and Evaluations. Treats hyperparameters as first-class entities.

Metadata Framework: FAIR4ML

Description:
A metadata schema for FAIR ML models, extending Schema.org and CodeMeta.

Key Classes and Properties:
`MLModel MLModelEvaluation, trainedOn, validatedOn, testedOn, fineTunedFrom`

Use in OpenModelsHub:
FAIR compliance assessment and comprehensive model documentation. Supports ethical AI and environmental impact tracking.

Metadata Framework: MEX Vocabulary

Description:
A lightweight vocabulary for ML experiment metadata, extending PROV-O with three layers: mexcore, mexalgo, mexperf

Key Classes & Properties:
`Experiment, ExperimentConfiguration, Execution, Dataset, Model, Phase, Algorithm, Tool, HyperParameter, LearningMethod, PerformanceMeasure, ExecutionPerformance`

Use in OpenModelsHub: Experiment tracking with performance measures. Provides hyperparameter and measurement vocabulary.

Metadata Framework: HPC Ontology

Description:

A semantic framework for describing high-performance computing resources, hardware, and compute environments, including systems, CPUs/GPUs, memory, storage, and execution metrics.

Core Classes & Properties:

`Cluster, Hardware, Processor, Accelerator, Memory, Computer, Server,cpuCoreCount, cpuFrequency, memorySize, gpuMemorySizePerNode, processorPeakPerformance`

Use in OpenModelsHub:

Tracks resources across Local, Cloud, and HPC systems. Records hardware specifications, utilization metrics and power consumption for energy and carbon footprint analysis.

Metadata Integration: ML Asset

OpenModelsHub Attribute	Dublin Core	Schema.org	DataCite	PROV-O
id	–	@id	Identifier	prov:Entity
persistent_identifier	dc:identifier	schema:identifier	datacite:identifier	–
name	dc:title	schema:name	datacite:title	–
description	dc:description	schema:description	datacite:description	–
version	–	schema:version	datacite:version	–
created_at	dc:date	schema:dateCreated	datacite:date	–
updated_at	dct:modified	schema:dateModified	–	–
created_by	dc:creator	schema:creator	datacite:creator	prov:wasAttributedTo
organization	dc:publisher	schema:publisher	datacite:publisher	–
license	dc:rights	schema:license	datacite:rights	–
subjects	dc:subject	schema:keywords	datacite:subject	–
access_rights	dct:accessRights	–	datacite:rights	–
parent_version	dct:isVersionOf	schema:isBasedOn	datacite:relatedIdentifier	prov:wasDerivedFrom

Metadata Integration: Researcher

OpenModelsHub Attribute	Dublin Core	Schema.org	DataCite	PROV-O
Researcher entity	`dc:creator`	`schema:Person`	`datacite:creator`	`prov:Agent`
id	–	`schema:@id`	`datacite:Identifier`	–
first_name	–	`schema:givenName`	`datacite:givenName`	–
last_name	–	`schema:familyName`	`datacite:familyName`	–
email	–	`schema:email`	–	–
orcid_id	`dc:identifier`	`schema:identifier`	`datacite:nameIdentifier`	–
expertise_areas	–	`schema:knowsAbout`	–	–
research_interests	–	`schema:knowsAbout`	–	–
Relationship to Organizations	–	`schema:memberOf`	`datacite:affiliation`	–
Relationship to ML Assets	`dc:creator`	`schema:creator`	`datacite:creator`	`prov:wasAttributedTo`

Metadata Integration: Organization

OpenModelsHub Attribute	Dublin Core	Schema.org	DataCite	PROV-O
Organization entity	`dc:publisher`	`schema:Organization`	`datacite:publisher / datacite:contributor`	–
id	–	`schema:@id`	`datacite:Identifier`	–
name	`dc:publisher`	`schema:name`	`datacite:publisher`	–
type	–	`schema:@type (Organization types)`	`datacite:contributorType`	–
location	–	`schema:address`	–	–
website	–	`schema:url`	–	–
policies	–	–	–	–
Relationship to Researchers	–	`schema:member`	–	–
Relationship to Resources	–	`schema:ownerOf`	–	–

Metadata Integration: Model

OpenModelsHub Attribute	ML-Schema	FAIR4ML	MEX	PROV-O
Model entity	`mls:Model`	`fair4ml:MLModel`	`mex:Model`	`prov:Entity`
architecture	`mls:algorithm`	`fair4ml:modelCategory`	`mex:Algorithm`	–
framework	`mls:software`	`fair4ml:softwareRequirements`	`mex:Tool`	–
model_type	–	`fair4ml:mlTask`	–	–
input_schema	–	Related to `fair4ml:usageInstructions`	–	–
output_schema	–	Related to `fair4ml:usageInstructions`	–	–
training_datasets	`mls:dataset`	`fair4ml:trainedOn`	`om:mlFeatures.featureSources`	`prov:used`

Metadata Integration: Dataset

OpenModelsHub Attribute	Schema.org	DCAT	Croissant	ML-Schema
Dataset entity	`schema:Dataset`	`dcat:Dataset`	`Extends schema:Dataset`	`mls:Dataset`
file_paths	`schema:distribution`	`dcat:distribution`	`cr:url`	–
total_size_bytes	`schema:contentSize`	`dcat:byteSize`	–	–
format	`schema:encodingFormat`	`dcat:mediaType`	`cr:encodingFormat`	–
schema	`schema:variableMeasured`	–	`cr:RecordSet`	–
num_records	–	–	`cr:recordCount`	`mls:DatasetCharacteristic`
num_features	–	–	`cr:Field`	`mls:DatasetCharacteristic`
target_column	–	–	`cr:Field`	–
data_types	`schema:variableMeasured`	–	`cr:dataType`	–
categorical_columns	–	–	`cr:Field`	–
numerical_columns	–	–	`cr:Field`	–

Metadata Integration: Experiment

OpenModelsHub Attribute	ML-Schema	FAIR4ML	MEX	PROV-O
Experiment entity	`mls:Run`	`fair4ml:TrainingRun`	`mex:Execution`	`prov:Activity`
experiment_type	–	–	`mex:ApplicationContext`	–
start_time	–	–	–	`prov:startedAtTime`
end_time	–	–	–	`prov:endedAtTime`
code_repository_url	–	Inherits `schema:codeRepository`	–	–
environment_specification	–	`fair4ml:softwareRequirements`	–	–
base_model relationship	`mls:Model`	`fair4ml:fineTunedFrom`	`mex:Model`	`prov:used`
datasets relationship	`mls:hasInput`	–	–	`prov:used`
produced_models relationship	`mls:hasOutput`	–	`prov:wasGeneratedBy`	–
hyperparameters relationship	`mls:hasHyperParameter`	–	Via `mex:HyperParameter`	–
metrics relationship	`mls:hasQuality`	–	`mexperf:PerformanceMeasure`	–
checkpoints relationship	–	–	–	`prov:wasGeneratedBy`
resource_utilizations relationship	–	–	`mex:Execution`	`prov:Activity`

Metadata Integration: Hyperparameter, Checkpoint, ComputationalResource

OpenModelsHub Attribute	ML-Schema	MEX
Hyperparameter entity	`mls:HyperParameter`	`mex:HyperParameter`
Relationship to Experiment	`mls:hasHyperParameter`	`mex:HyperParameterCollection`

OpenModelsHub Attribute	PROV-O	ML-Schema
Checkpoint entity	`prov:Entity`	`mls:Model`
saved_at	`prov:generatedAtTime`	–
metrics_snapshot	–	`mls:Evaluation`
Relationship to Experiment	`prov:wasGeneratedBy`	–
Relationship to Model	–	`mls:Model`

OpenModelsHub Attribute	ML-Schema	MEX	PROV-O
Metric entity	`mls:Evaluation + mls:Measure`	`mexperf:PerformanceMeasure`	–
metric_name	`mls:Measure`	–	–
value	`mls:hasValue`	–	–
timestamp	–	–	`prov:generatedAtTime`
metric_type	`mls:Measure`	`mexperf`	–

Metadata Integration: Experiment Dataset

OpenModelsHub Attribute	Croissant	ML-Schema	MEX
ExperimentDataset entity	`cr:Split`	`mls:DatasetCharacteristic`	`mex:Phase`
role	–	Via `mls:hasInput`	`mex:Phase`
split_percentage	Via `cr:Split`	–	–
num_records	`cr:recordCount`	`mls:DatasetCharacteristic`	–
random_seed	Via `cr:Split`	–	–
preprocessing_applied	–	`mls:DataTransformation`	–
augmentation_enabled	–	`mls:DataTransformation`	–
Relationship to Experiment	–	`mls:hasInput`	`mex:Execution`
Relationship to Dataset	Parent `cr:Dataset`	`mls:dataset`	–

Metadata Integration: Computational Resources

ComputationalResource

OpenModelsHub Attribute	HPC Ontology
Resource entity	`hpc:Hardware / hpc:Computer / hpc:Server`
resource_type	`hpc:Workstation/hpc:Server, hpc:Computer, hpc:Cluster`
cpu_cores	`hpc:cpuCoreCount`
memory_gb	`hpc:memorySize`
gpu_count	`Via hpc:Accelerator count`
gpu_model	`Via hpc:Accelerator properties`
energy_efficiency	`hpc:powerEfficiency`

HPCResource

OpenModelsHub Attribute	HPC Ontology
HPCResource entity	`hpc:Cluster / hpc:SuperComputer`
cluster_name	`Via hpc:Cluster identifier`
total_nodes	`hpc:computeNodeCount`
node_configuration	`hpc:systemArchitecture`
max_job_duration	`Related to hpc:maxExecutionTime`
interconnect_type	`hpc:gpuInterconnect`

LocalResource

OpenModelsHub Attribute	HPC Ontology
LocalResource entity	`hpc:Workstation / hpc:Server`
hostname	`Via hpc:Computer identifier`
power_consumption_watts	`hpc:power`

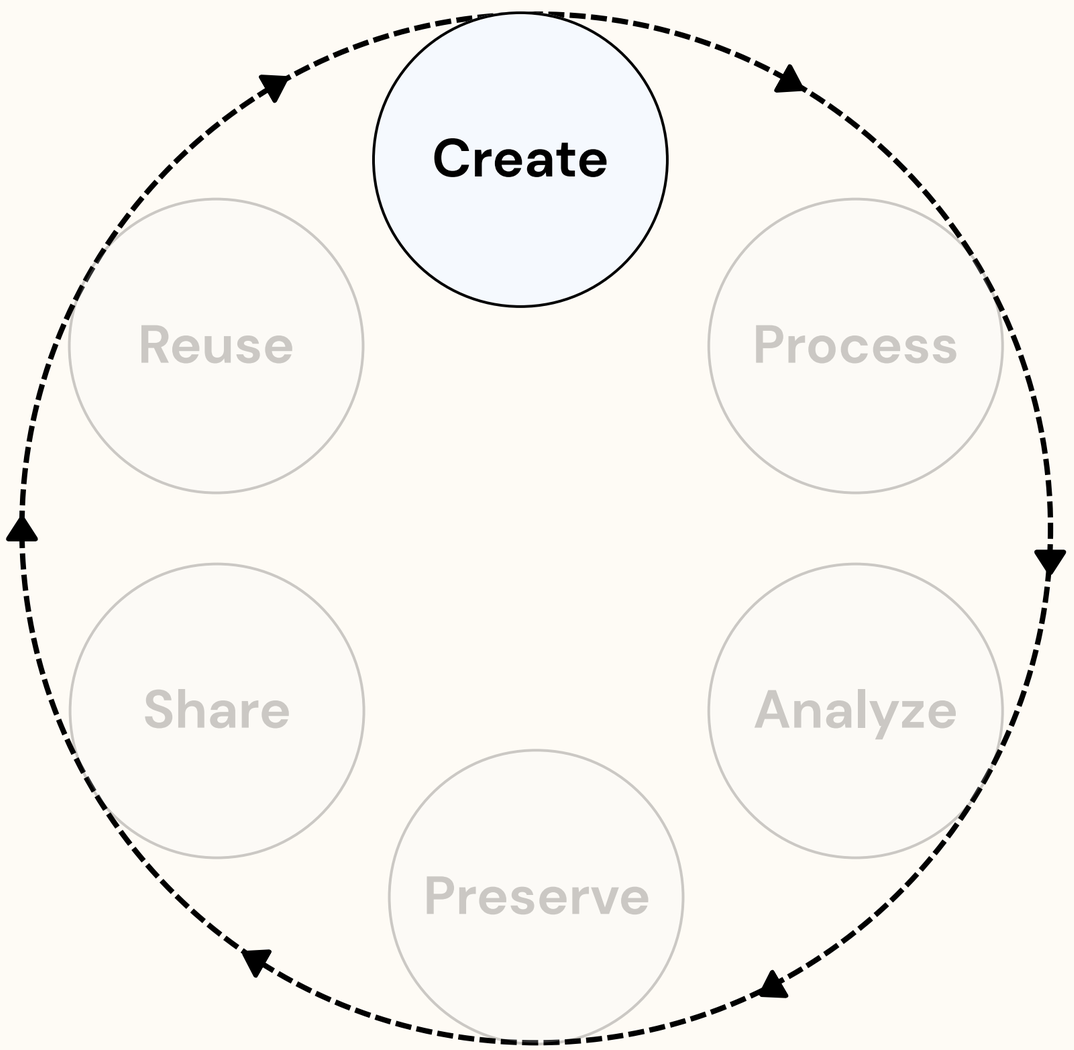
CloudResource

OpenModelsHub Attribute	HPC Ontology
CloudResource entity	`hpc:Computer`
provider_region	`hpc:country`

Metadata Integration: Resource Utilization

OpenModelsHub Attribute	HPC Ontology
ResourceUtilization entity	Related to hpc:Hardware usage
resource	→ hpc:Computer / hpc:Cluster
duration_seconds	hpc:averageExecutionTime
cpu_utilization_percent	hpc:streamingProcessorUtilizationRate
gpu_utilization_percent	hpc:streamingProcessorUtilizationRate
memory_usage_gb	hpc:memoryOccupancy
storage_usage_gb	Related to hpc:harddriveSize
network_io_gb	hpc:dataTransferSize
energy_consumption_kwh	Related to hpc:power
peak_usage_metrics	hpc:maxExecutionTime, various utilization properties
average_usage_metrics	hpc:averageExecutionTime, hpc:memoryThroughputRate

Data Life Plan

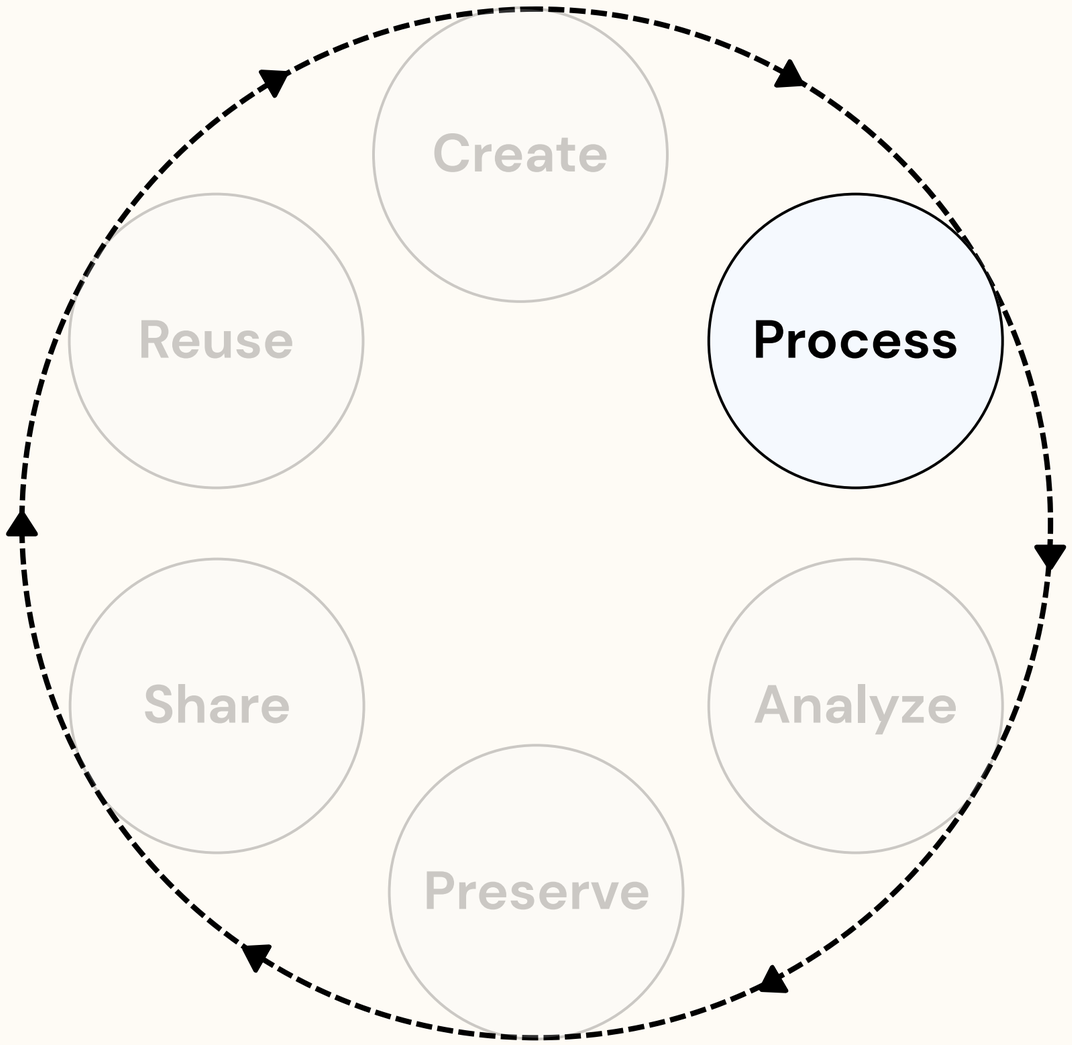


Objective:

Register ML assets as digital research objects with complete, validated metadata and provenance.

Activities:

- Capture configurations, metrics, and resource logs from training runs
- Assign persistent identifiers (DOIs, UUIDs, ORCIDs)
- Validate metadata completeness and schema compliance
- Record authorship and institutional ownership (via ORCID)
- Ensure legal and ethical compliance (licensing, privacy classification)
- Generate integrity checksums
- Define access rights
- Normalize metadata (metrics, parameters, environment specs)

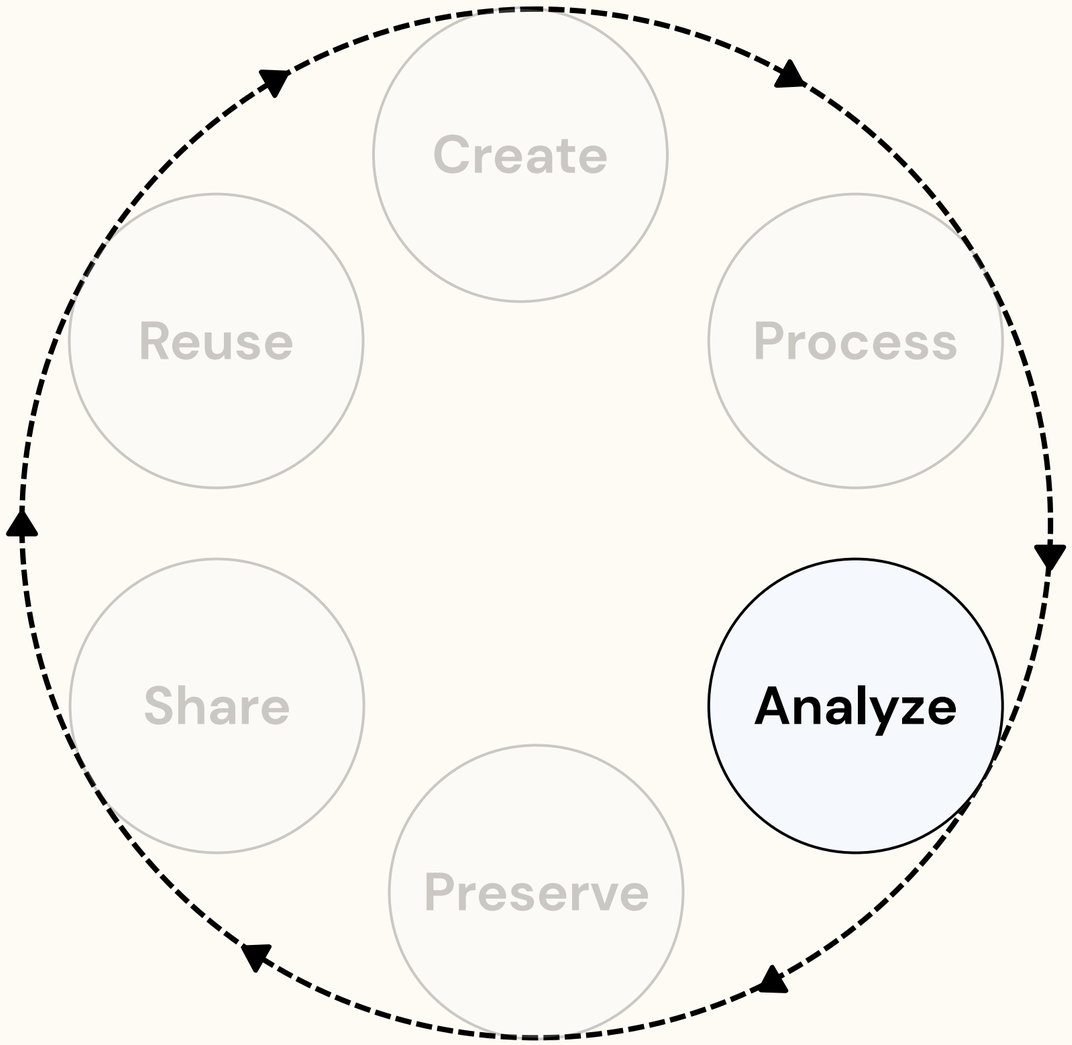


Objective:

Enrich metadata with relationships, provenance, and cross-references to improve interoperability and discoverability.

Activities:

- Define provenance links (experiment ↔ dataset ↔ model ↔ base model)
- Standardize computational usage metrics across resources
- Record preprocessing and augmentation details in dataset associations
- Add subject classifications and keywords for discoverability
- Cross-reference external assets (repositories, commits, DOIs)

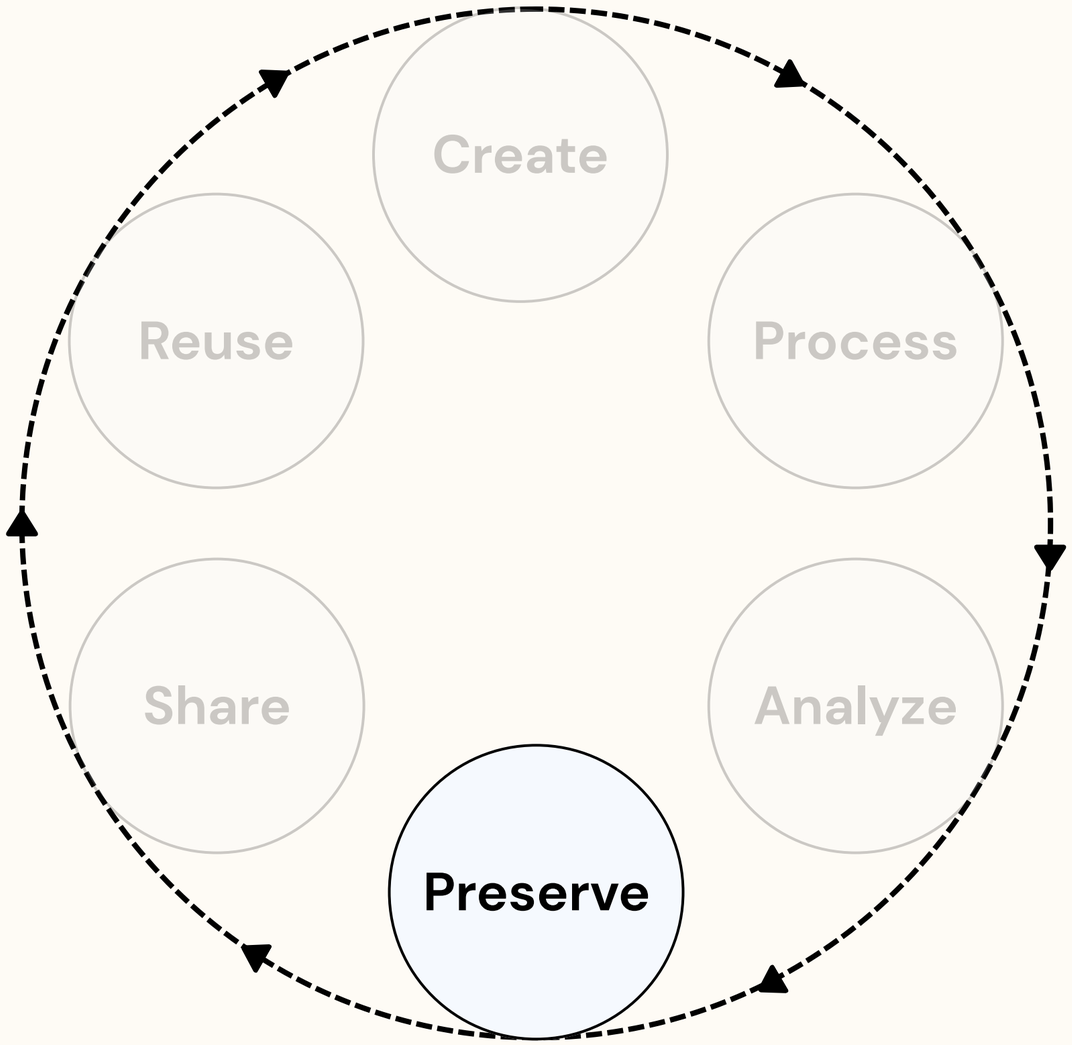


Objective:

Enable meta-level analysis through structured access to experiment metrics, resource usage, and provenance data.

Activities:

- Support comparative queries (e.g., performance vs. cost, framework comparisons)
- Provide reproducibility metrics (environment completeness, config consistency)
- Enable resource impact analysis:
  - Energy and carbon footprint trends
  - Cost–performance efficiency
- Deliver analytical views for institutional monitoring:
  - Performance and utilization trends
  - Sustainability and dataset reuse dashboards
- Support hyperparameter exploration and outcome correlation

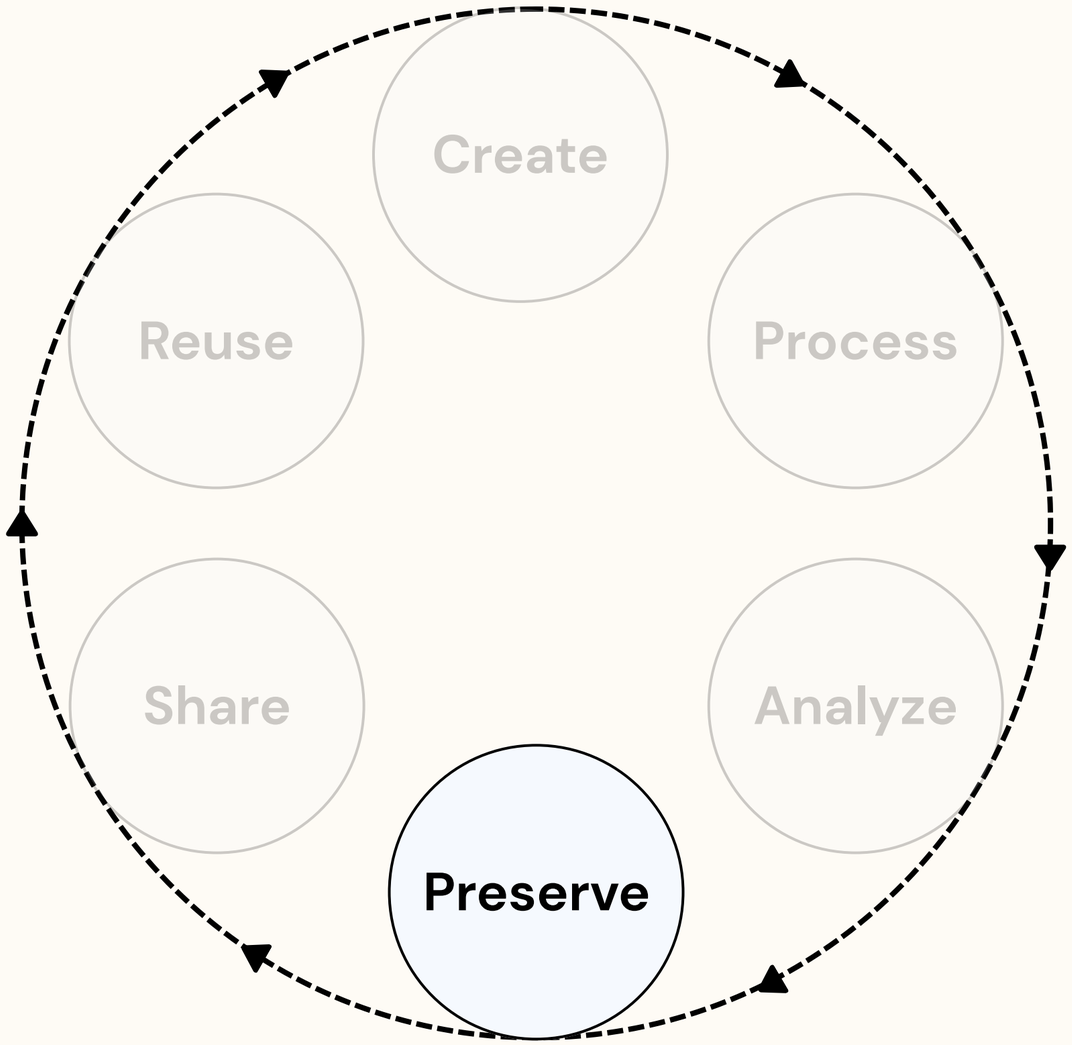


Objective:

Ensure long-term preservation, integrity, and accessibility of ML assets and metadata through durable formats and tiered storage.

Activities:

- Archive datasets, models, and metadata in institutional/national repositories (e.g., Zenodo)
- Store assets in durable, non-proprietary formats:
  - Metadata: JSON-LD, XML
  - Models: ONNX, PyTorch (.pt), TensorFlow SavedModel
  - Datasets: Parquet, HDF5, CSV
  - Configurations: YAML, JSON
- Validate integrity with periodic checksums
- Preserve complete provenance and version lineage

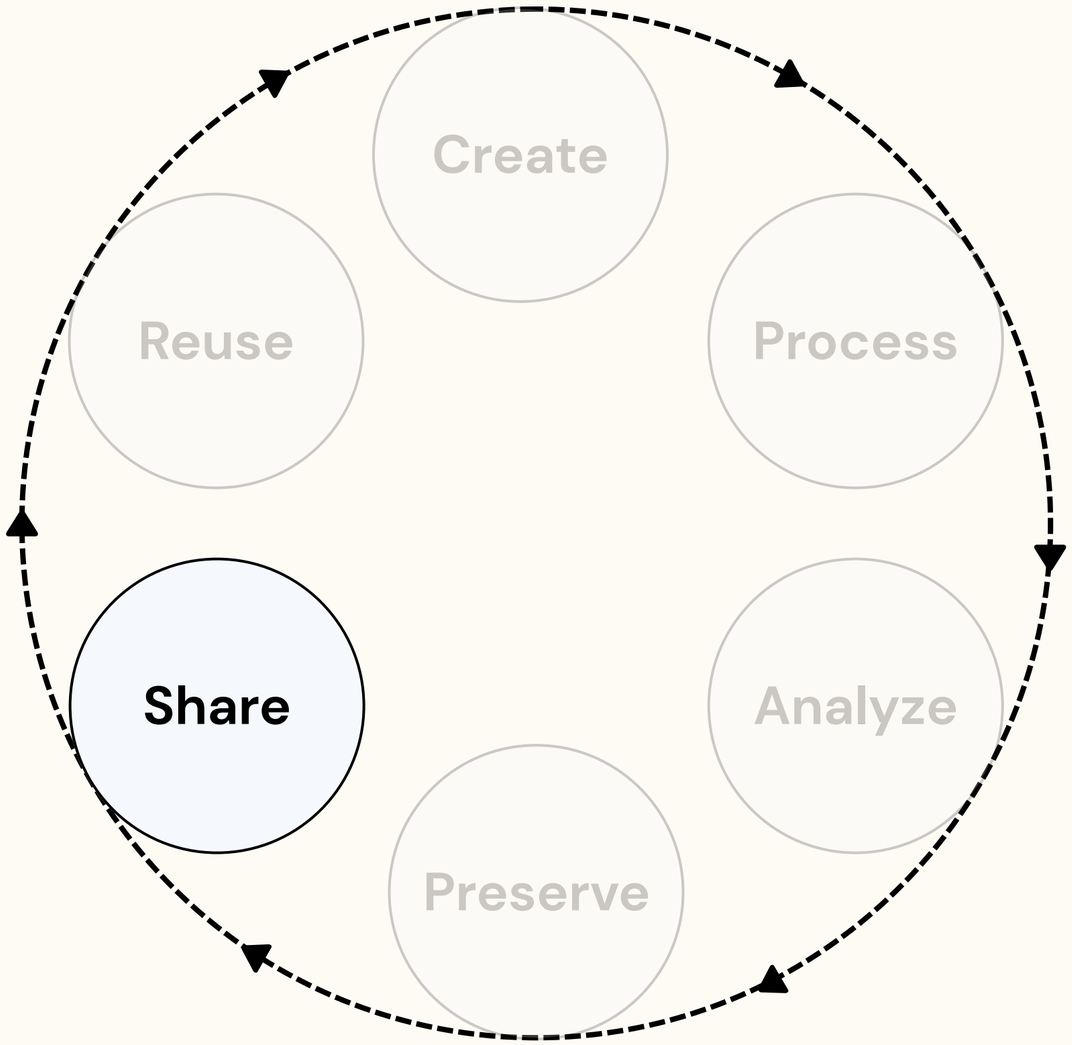


Infrastructure strategy:

- Implement tiered storage:
  - Active: SSD for frequent access
  - Cold: HDD/object storage for infrequent access
  - Archival: Geo-redundant institutional/cloud storage
- Maintain redundant storage across geographic locations

Metadata Retention:

- Metadata-only records: Persist after data/model deletion
- Citation continuity: DOIs resolve to metadata
- Lifecycle states:
  - ACTIVE: Full metadata + accessible files
  - DEPRECATED: Full metadata + files + deprecation notice
  - DELETED: Metadata-only record, no file access

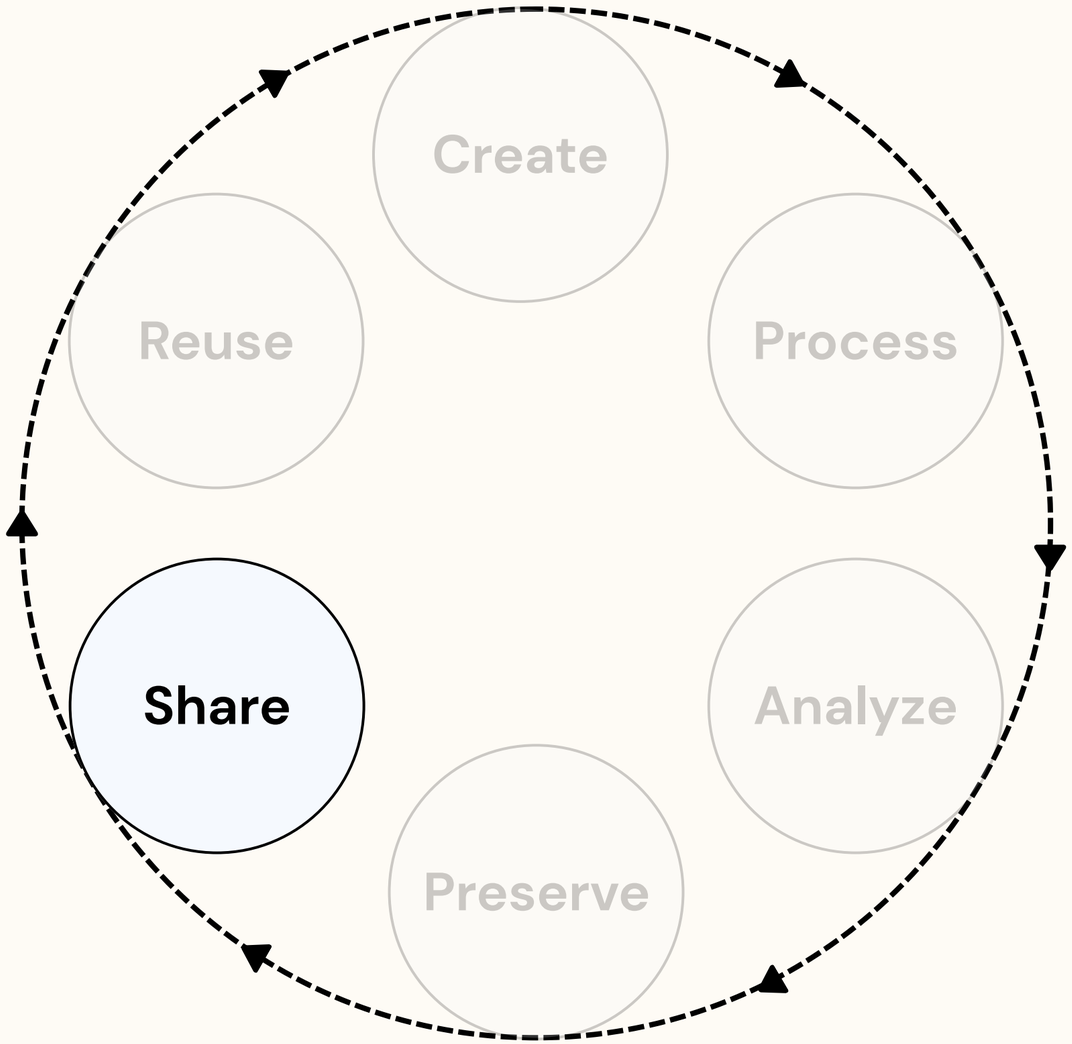


Objective:

Disseminate preserved assets and metadata for discovery, reuse, and citation in accordance with FAIR and ethical principles.

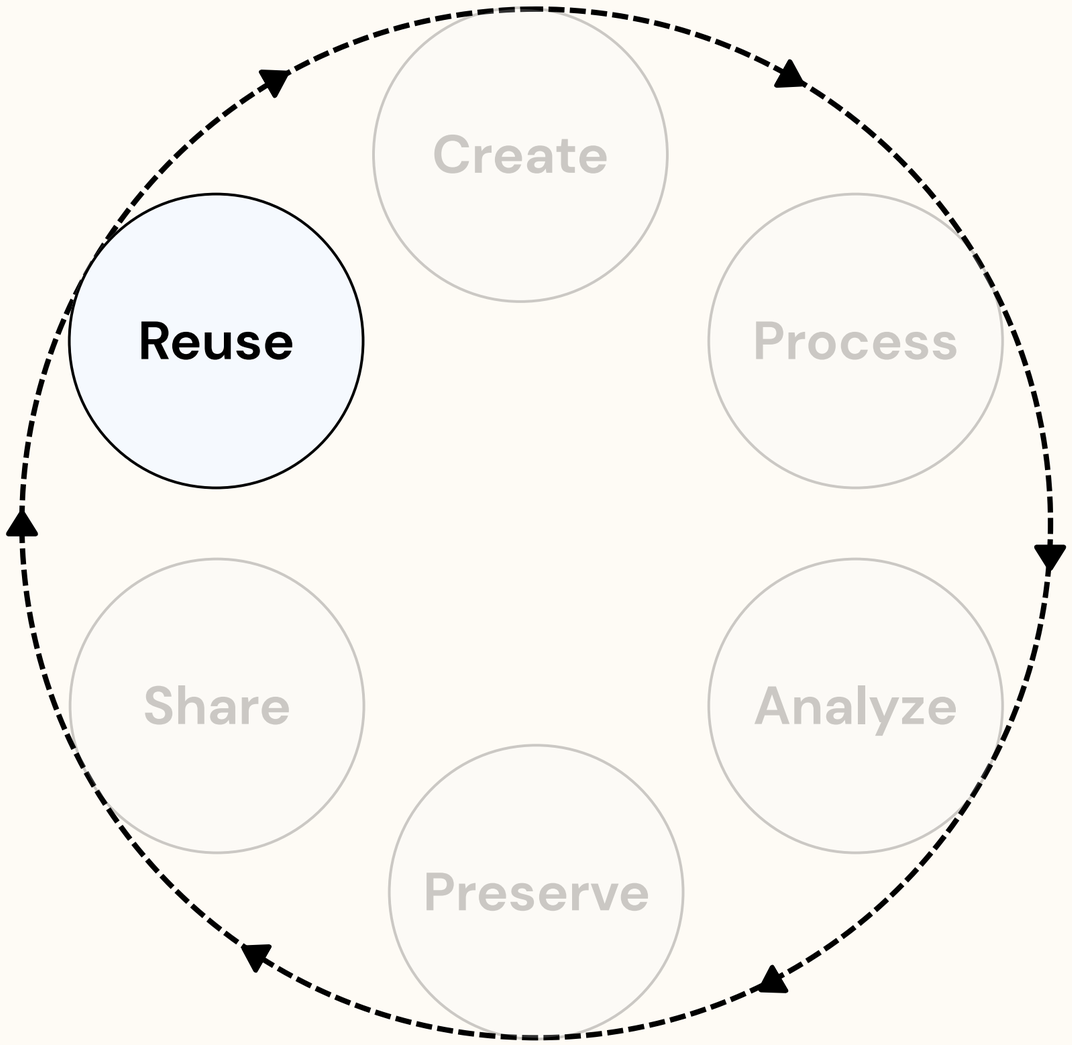
Activities:

- Publish assets via OpenModelsHub catalogue and REST API
- Filter by metadata fields and full-text search
- Assign licenses (MIT, Apache 2.0, CC, ODC)
- Configure access rights per privacy, ethics, and institutional policies
- Expose metadata through standard interfaces (REST API)
- Link datasets, experiments, and models to publications and code repositories
- Provide citation metadata (BibTeX, RIS, CSL-JSON)



External Catalog Integration:

- DOIs: Register via DataCite, resolving to OpenModelsHub landing pages
- Repository listing: Registered in re3data.org
- Web discoverability: Indexed by Google Dataset Search and academic engines
- Disciplinary catalogs: Cross-reference with community platforms (Papers with Code, Hugging Face, OpenML, UCI ML Repository)



Objective:

Enable reuse of ML assets for replication, benchmarking, and meta-research, with usage feedback integrated into the asset lifecycle.

Activities:

- Support experiment replication using preserved configurations and environments
- Enable comprehensive metadata search: by framework, model type, dataset format, organization, or timeframe
- Provide provenance-based browsing
- Link derived works to originals via version lineage
- Ensure code and configurations are accessible for reproducibility
- Track reuse metrics to inform future asset creation (Downloads, citations, and derived works)

FAIR Evaluation

Findable

✓ F1: (Meta)data are assigned a globally unique and persistent

Evidence:

- MLAsset base class includes both id(UUID, system-generated) and persistent_identifier (external PID like DOI)
- DataCite Metadata Schema integration documented for DOI support
- Data plan Phase 1 mentions DOI assignment for published assets

Findable

✓ F2: Data are described with rich metadata

Evidence:

- MLAsset provides comprehensive base metadata (identity, ownership, versioning, integrity, licensing, access rights)
- Entity-specific extensions: Model (architecture, framework, performance), Dataset (statistics, privacy, collection), Experiment (configuration, reproducibility, environment)
- Integration with international metadata standards documented

Findable

✓ F3: Metadata clearly and explicitly include the identifier of the data they describe

Evidence:

- All entities include explicit id (UUID) and persistent_identifier fields
- Integration mapping shows identifier fields mapped to dc:identifier, schema:identifier, datacite:identifier
- Metadata structure explicitly links identifiers to all asset descriptions

Findable

✓ F4: (Meta)data are registered or indexed in a searchable resource

Evidence:

- Internal indexing well documented: database indexes on created_at, access_rights, organization, framework, model_type, format, privacy_level
- Data plan Phase 5 documents REST API with query/filter capabilities by metadata fields and full-text search
- Data plan Phase 6 documents search by framework, model type, dataset format, organization, time period
- **External catalog integration** documented in data-plan Phase 5:
  - DataCite DOI registration with automatic metadata synchronization
  - Schema.org structured data for Google Dataset Search indexing
  - Disciplinary catalog integration (Papers with Code, Hugging Face)

Accessible

✓ A1: (Meta)data are retrievable by their identifier using a standardized communications protocol

Evidence:

- REST API documented in data plan Phase 5 (Share)
- Django REST Framework
- Standard HTTP/HTTPS protocols implied

Accessible

✓ A1.2: The protocol allows for an authentication and authorization procedure, where necessary

Evidence:

- MLAsset includes access_rights enum
- Dataset includes privacy_level enum
- Data plan Phase 5 documents access rights configuration based on privacy, ethics, and institutional policies
- REST Framework authentication configured (planned)

Accessible

✓ A2: Metadata are accessible even when the data are no longer available

Evidence:

- Comprehensive metadata persistence policy documented in data-plan Phase 4
- Metadata-only records: Complete metadata persists indefinitely even when data files are removed
- Asset lifecycle states defined: ACTIVE, DEPRECATED, DELETED
- Citation continuity: DOIs continue to resolve to metadata

Interoperable

✓ I1: (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation

Evidence:

- Integration with 11 formal vocabularies with defined namespaces
- All namespace URIs documented (dc:, dct:, schema:, dcat:, prov:, cr:, mls:, fair4ml:, mex:, hpc:, omh:)
- Data plan Phase 4 mentions metadata export in XML formats
- Integration mapping provides complete attribute-to-vocabulary mappings

Interoperable

✓ I2: (Meta)data use vocabularies that follow FAIR principles

Evidence:

- Standards from recognized international bodies (W3C, Dublin Core Metadata Initiative, DataCite, MLCommons, RDA)
- Standards are actively maintained with persistent URIs and formal specifications
- Documented rationale for vocabulary selection emphasizing broad interoperability and domain specificity

Interoperable

✓ I3: (Meta)data include qualified references to other (meta)data

Evidence:

- Comprehensive relationship documentation with cardinality specifications
- PROV-O integration for provenance chains (wasGeneratedBy, used, wasAttributedTo, wasDerivedFrom)
- Versioning via parent_version
- Experiment relationships: base_model (fine-tuning), datasets (via ExperimentDataset), produced_models
- ExperimentDataset captures qualified usage (role, split definition, preprocessing)

Reusable

✓ R1: Meta(data) are richly described with a plurality of accurate and relevant attributes

Evidence:
Detailed attribute specifications in logical-model with types, constraints, defaults, validation rules
Multi-layered metadata: foundation layer (Dublin Core, Schema.org, DataCite, DCAT), ML layer (Croissant, ML-Schema, FAIR4ML, MEX), infrastructure layer (HPC Ontology)
Enumeration types defined for controlled vocabularies
Data plan Phase 1 documents metadata validation and normalization at creation time

Reusable

✓ R1.1: (Meta)data are released with a clear and accessible data usage license

Evidence:

- MLAsset includes required license field
- Data plan Phase 5 documents license assignment strategy (MIT, Apache 2.0, Creative Commons, Open Data Commons)
- Mapped to dc:rights, schema:license, datacite:rights

Reusable

✓ R1.2: (Meta)data are associated with detailed provenance

Evidence:

- W3C PROV-O integration throughout entities
- Versioning: parent_version field with recursive lineage retrieval method
- Experiment tracking: code_repository_url, code_commit_hash, environment_specification, random_seed, reproducibility_hash
- ExperimentDataset: split definitions with random seeds for reproducibility
- ResourceUtilization: complete computational resource tracking
- Attribution: created_by (Researcher with ORCID), organization, timestamps

Reusable

✓ R1.3: (Meta)data meet domain-relevant community standards

Evidence:

- ML-specific standards: Croissant (MLCommons), ML-Schema (W3C Community Group)
- Infrastructure standards: HPC Ontology (HPC-FAIR)
- Foundation standards: Dublin Core, Schema.org, DataCite, DCAT, PROV-O
- Data lifecycle framework: UK Data Service Data Lifecycle
- Complete attribute-to-standard mapping documentation

XSD Schema

```
<xs:complexType name="MLAssetBaseType" abstract="true">
   <xs:sequence>
     <!-- Identity -->
     <xs:element name="id" type="xs:string">
     </xs:element>
     <xs:element name="persistentIdentifier" type="xs:string">
     </xs:element>
     <!-- Basic Information -->
     <xs:element name="name" type="xs:string">
     </xs:element>
     <xs:element name="description" type="xs:string">
     </xs:element>
     <xs:element name="version" type="xs:string">
     </xs:element>
     ...
```

```
<xs:complexType name="ModelType">

   <xs:complexContent>
     <xs:extension base="omh:MLAssetBaseType">
       <xs:sequence>
         <!-- File Information (OpenModelsHub extension) -->
         <xs:element name="modelFilePath" type="xs:string"/>
         <xs:element name="modelFileSize" type="xs:long"/>
         <xs:element name="modelFormat" type="omh:ModelFormatEnum"/>
         <!-- Architecture -->
         <xs:element name="architecture" type="xs:string">
         </xs:element>
         <xs:element name="framework" type="xs:string">
         </xs:element>
         <xs:element name="frameworkVersion" type="xs:string"/>
         <xs:element name="modelType" type="omh:ModelTypeEnum">
         <!-- Schemas -->
         <xs:element name="inputSchema" type="omh:SchemaDefinitionType" minOccurs="0"/>
         <xs:element name="outputSchema" type="omh:SchemaDefinitionType" minOccurs="0"/>
         <!-- Performance -->
         <xs:element name="inferenceTimeMs" type="xs:float" minOccurs="0"/>
         <xs:element name="modelSizeMb" type="xs:float" minOccurs="0"/>
         ...
```

```
<xs:complexType name="DatasetType">
    <xs:complexContent>
      <xs:extension base="omh:MLAssetBaseType">
        <xs:sequence>
          <!-- File Information -->
          <xs:element name="filePaths" type="omh:FilePathListType">
          </xs:element>
          <xs:element name="totalSizeBytes" type="xs:long">
          </xs:element>
          <xs:element name="format" type="omh:DatasetFormatEnum">
          </xs:element>
          <!-- Structure -->
          <xs:element name="schema" type="omh:SchemaDefinitionType" minOccurs="0">
          </xs:element>
          <xs:element name="numFeatures" type="xs:int" minOccurs="0">
          </xs:element>
          <xs:element name="targetColumn" type="xs:string" minOccurs="0"/>
          <!-- Data Analysis -->
          <xs:element name="dataTypes" type="omh:DataTypesMapType" minOccurs="0">
          </xs:element>
          <xs:element name="categoricalColumns" type="omh:ColumnListType" minOccurs="0">
          </xs:element>
          <xs:element name="numericalColumns" type="omh:ColumnListType" minOccurs="0">
          ...
```

```
<xs:complexType name="ExperimentType">

   <xs:complexContent>
     <xs:extension base="omh:MLAssetBaseType">
       <xs:sequence>
         <!-- Type and Status -->
         <xs:element name="experimentType" type="omh:ExperimentTypeEnum">
         </xs:element>
         <xs:element name="status" type="omh:ExperimentStatusEnum"/>
         <!-- Timing -->
         <xs:element name="startTime" type="xs:dateTime" minOccurs="0">
         </xs:element>
         <xs:element name="endTime" type="xs:dateTime" minOccurs="0">
         </xs:element>
         <xs:element name="durationSeconds" type="xs:int" minOccurs="0"/>
         <!-- Configuration (OpenModelsHub extension) -->
         <xs:element name="configParameters" type="omh:ConfigParametersType" minOccurs="0"/>
         <xs:element name="randomSeed" type="xs:int" minOccurs="0"/>
         <xs:element name="reproducibilityHash" type="xs:string" minOccurs="0"/>
         <!-- Code & Environment -->
         <xs:element name="codeRepositoryUrl" type="xs:anyURI" minOccurs="0">
         </xs:element>
         <xs:element name="codeCommitHash" type="xs:string" minOccurs="0"/>
         <xs:element name="environmentSpecification" type="omh:EnvironmentSpecType" minOccurs="0"
         ...
```

Django Mapping

```python
class MLAsset(models.Model):
    """
    Abstract base class for all ML assets (Models, Datasets, Experiments).
    Provides common identity, metadata, versioning, and ownership functionality.
    """
    # Identity
    id = models.UUIDField(
        primary_key=True,
        default=uuid.uuid4,
        editable=False,
        help_text="System-generated unique identifier"
    )
    persistent_identifier = models.CharField(
        max_length=255,
        unique=True,
        help_text="External persistent ID (DOI, ARK, etc.)"
    )

    # Basic info
    name = models.CharField(
        max_length=255,
        db_index=True,
        help_text="Human-readable asset name"
    )
    description = models.TextField(
        help_text="Detailed description of the asset"
    )    version = models.CharField(
        max_length=50,
        help_text="Version identifier (e.g., v2.1.0)"
    )
    ...
```

```python
class Model(MLAsset):
    """
    Trained machine learning model with performance metrics and deployment info.
    """
    # File info
    model_file_path = models.CharField(
        max_length=500,
        help_text="Location of model file"
    )
    model_file_size = models.BigIntegerField(
        validators=[MinValueValidator(1)],
        help_text="Size in bytes"
    )
    model_format = models.CharField(
        max_length=30,
        choices=ModelFormat.choices,
        help_text="Model file format"
    )

    # Architecture
    architecture = models.TextField(
        help_text="Model architecture description"
    )
    framework = models.CharField(
        max_length=30,
        choices=Framework.choices,
        help_text="ML framework used"
    )
    framework_version = models.CharField(
        max_length=20,
        help_text="Framework version"
    )
    ...
```

```python
class Dataset(MLAsset):
    """
    Collection of data used for ML training, validation, or testing.
    """
    # File and structure
    file_paths = models.JSONField(
        default=list,
        help_text="List of dataset file locations"
    )
    total_size_bytes = models.BigIntegerField(
        validators=[MinValueValidator(1)],
        help_text="Combined size of all files"
    )
    format = models.CharField(
        max_length=20,
        choices=DatasetFormat.choices,
        help_text="Primary data format"
    )
    schema = models.JSONField(
        null=True,
        blank=True,
        help_text="Data structure definition"
    )

    # Statistics
    num_records = models.BigIntegerField(
        null=True,
        blank=True,
        validators=[MinValueValidator(0)],
        help_text="Total number of data points"
    )
    ...
```

```python
class Experiment(MLAsset):
    """
    Complete ML training or evaluation run with full reproducibility information.
    """
    # Classification
    experiment_type = models.CharField(
        max_length=30,
        choices=ExperimentType.choices,
        help_text="Purpose of experiment"
    )
    status = models.CharField(
        max_length=20,
        choices=ExperimentStatus.choices,
        default=ExperimentStatus.PENDING,
        db_index=True,
        help_text="Execution state"
    )

    # Timing
    start_time = models.DateTimeField(
        null=True,
        blank=True,
        db_index=True,
        help_text="When experiment began"
    )
    end_time = models.DateTimeField(
        null=True,
        blank=True,
        help_text="When experiment finished"
    )
    ...
```

Proof Of Concept

OpenModelsHub

Enhanced Machine Learning Repository with Computational Resource Tracking and Environmental Impact Assessment

Demonstrating FAIR principles, seven-standard metadata integration (Dublin Core, DataCite, DCAT, Hugging Face, Croissant, MLflow, ISO 14067), and comprehensive ML asset lifecycle management.

Explore Assets

1
ML Models
View All →

4
Datasets
View All →

1
Experiments
View All →

1
Researchers
View All →

Recent Models

Climate Detection ResNet...
Pytorch • 13 hours, 3 minutes ago
%

Recent Datasets

Test Dataset
CSV • 13 hours, 3 minutes ago
%

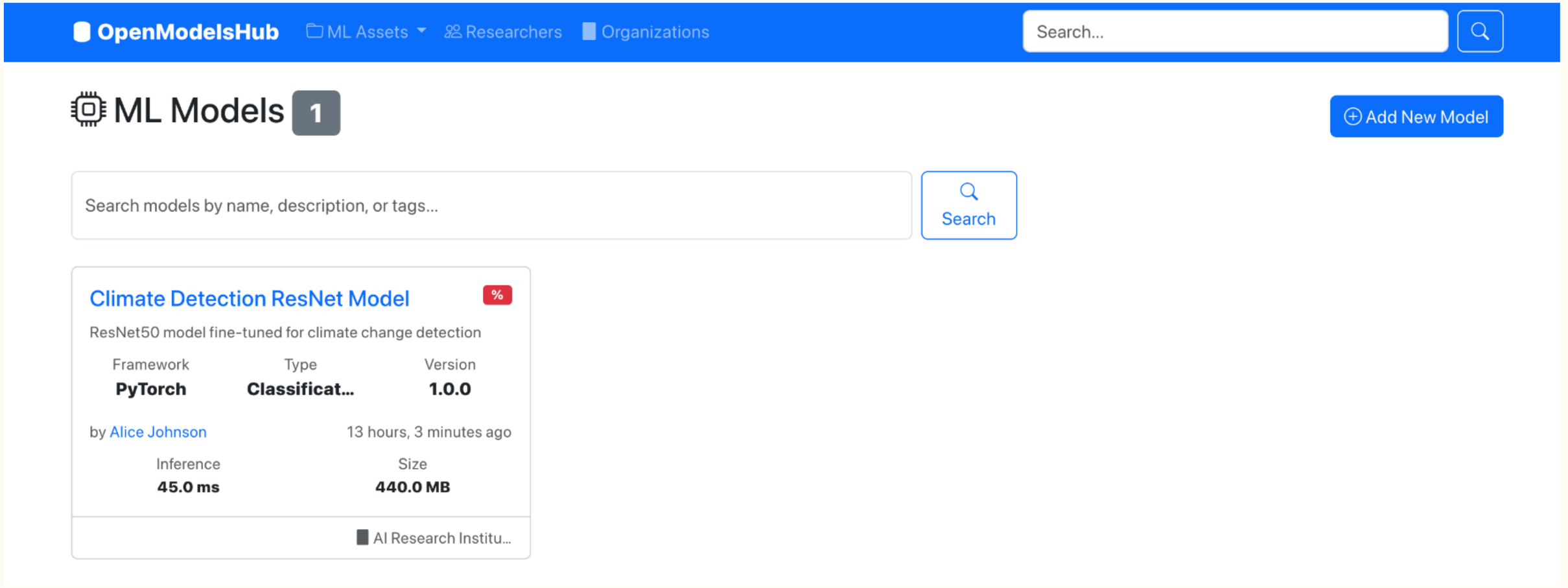
Test Dataset
CSV • 13 hours, 3 minutes ago
%

Test Dataset
CSV • 13 hours, 3 minutes ago
%

Climate Change Detection...
IMAGE_FOLDER • 10,000 records • 13 hours, 3 minutes ago
%

Recent Experiments

Climate Detection Traini...
Training • Completed
%


OpenModelsHub
ML Assets
Researchers
Organizations
Search...
ML Models 1
Add New Model
Search models by name, description, or tags...
Search
Climate Detection ResNet Model
%
ResNet50 model fine-tuned for climate change detection
Framework
PyTorch
Type
Classificat...
Version
1.0.0
by Alice Johnson
13 hours, 3 minutes ago
Inference
45.0 ms
Size
440.0 MB
AI Research Institu...

Climate Detection ResNet Model v1.0.0

% Quality Score

Model Overview

ResNet50 model fine-tuned for climate change detection

Technical Details

Framework: PyTorch v1.12.0

Model Type: Classification

Format: PyTorch

File Size: 419.6 MB

Performance

Inference Time: 45.0 ms

Memory Size: 440.0 MB

Architecture

```
ResNet50 with custom classification head (5 classes)
```

Metadata

Created by:
Alice Johnson

Organization:
AI Research Institute

Created:
Oct 14, 2025 (13 hours, 4 minutes ago)

Last Updated:
Oct 14, 2025

Access Rights:

License:
MIT

Persistent ID:
doi:10.5555/11223344

Input Schema:

▸ View JSON Schema

Output Schema:

▸ View JSON Schema

Resource Utilization 1 runs

Average CPU Usage **85.5%**

Average Memory Usage **48.5 GB**

Average GPU Usage **92.3%**

4h 14400s

Total Training Time

Environmental Impact High Impact

12.5 kg CO_2

Total Carbon Footprint

15.5 kWh

Energy Consumed

12.5000 kg CO_2/kWh

Carbon Intensity

⚠ Above 5kg CO_2 Threshold

Training Runs Resource Usage

Resource	CPU %	Memory GB	GPU %	Duration	CO_2 Impact	Started
ML Workstation 01 Local • Lab 301, Build...	85.5%	48.5 GB	92.3%	4h		13 hours, 4 minutes ago

← Back to Models

Export Metadata

Edit

Datasets 4

Add New Dataset

Search datasets by name, description, or tags... Search

Test Dataset

%

Major update

Format
CSV

Public 1.9 MB

by Alice Johnson 13 hours, 5 minutes ago

AI Research Institu...

Test Dataset

%

Bug fixes

Format
CSV

Public 1.0 MB

by Alice Johnson 13 hours, 5 minutes ago

AI Research Institu...

Test Dataset

%

Original version

Format
CSV

Public 976.6 KB

by Alice Johnson 13 hours, 5 minutes ago

AI Research Institu...

Climate Change Detection Data...

%

Satellite imagery dataset for climate change detection

Format	Records	Features
IMAGE_FOLDER	**10,000**	**512**

Public 2.0 GB

by Alice Johnson 13 hours, 5 minutes ago

AI Research Institu...

Climate Change Detection Dataset v1.0.0

Quality Score

Dataset Overview

Satellite imagery dataset for climate change detection

Dataset Statistics

Format: IMAGE_FOLDER

Total Size: 2.0 GB

Records: 10,000

Features: 512

Target Column: `label`

Privacy & Ethics

Privacy Level: Public

Ethical Considerations:
Dataset contains no personal information

Data Collection

Collection Method:
Satellite imagery from public sources

Sampling Strategy:
Random sampling across geographic regions

Column Types

Categorical: label

Numerical: temperature, elevation

Schema

▸ View Schema Details

Metadata

Created by:
Alice Johnson

Organization:
AI Research Institute

Created:
Oct 14, 2025 (13 hours, 5 minutes ago)

Last Updated:
Oct 14, 2025

Access Rights:

License: CC-BY-4.0

Persistent ID:
`doi:10.5555/12345678`

Citation

Copy Citation

File Locations

OpenModelsHub ML Assets Researchers Organizations Search...

Experiments 1

Add New Experiment

Search experiments by name, description, or tags...

Search

Climate Detection Training Run

%

Training CNN model on climate detection dataset

Type	Status	Version
Training	**Completed**	**1.0.0**

Duration	Started
4h	**13 hours, 5 minutes ago**

by Alice Johnson

13 hours, 5 minutes ago

AI Research Institu...

Climate Detection Training Run v1.0.0 Completed

Quality Score

Experiment Overview

Training CNN model on climate detection dataset

Execution Details

Type: Training

Status: Completed

Started: Oct 14, 2025 17:42

Completed: Oct 14, 2025 17:42

Duration: 4 hours (14400 seconds)

Reproducibility

Random Seed: 42

Reproducibility Hash: `abc123def456`

Code Repository: View Repository

Commit Hash: `a1b2c3d4e5f6789…`

Configuration Parameters

Model_Architecture: resnet50 **Optimizer:** adam **Loss_Function:** cross_entropy

Hyperparameters

Environment Specification

▸ View Environment Details

Metadata

Created by: Alice Johnson

Organization: AI Research Institute

Created: Oct 14, 2025 (13 hours, 5 minutes ago)

Last Updated: Oct 14, 2025

Access Rights:

License: MIT

Persistent ID: `doi:10.5555/87654321`

Citation

Copy Citation

OpenModelsHub ML Assets Researchers Organizations Search...

Researchers 1

Add New Researcher

Search researchers by name or affiliation... Search

Alice Johnson

H-Index Citations

Research Interests:

transformers multimodal learning

Expertise:

deep learning computer vision

alice.johnson@airi.edu

Joined ago

Alice Johnson

Research Profile

Professional Information

Affiliation:

Organization:

()

Location:

,

Research Metrics

H-Index	Citations
1 Models	4 Datasets

Research Interests

transformers multimodal learning

Expertise Areas

deep learning computer vision

Contact Information

Email:

alice.johnson@airi.edu

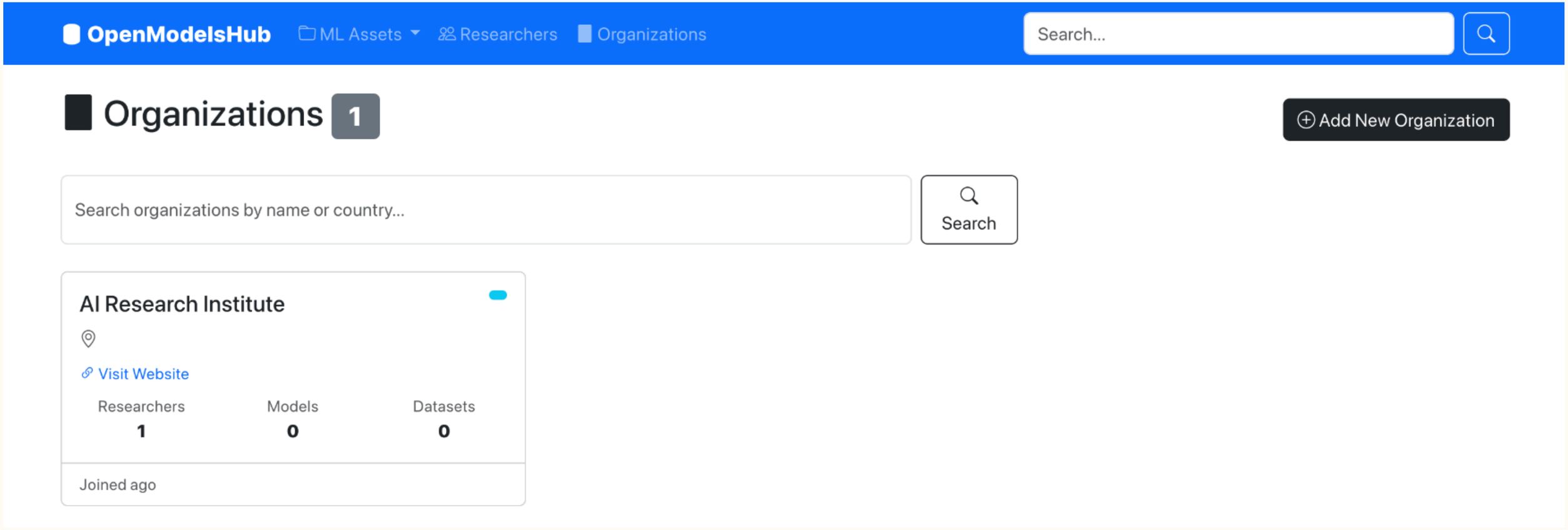
Contribution Statistics

ML Models	1
Datasets	4
Experiments	1
Publications	0

Recent Activity

Joined:

(ago)


OpenModelsHub
ML Assets
Researchers
Organizations
Search...
Organizations 1
Add New Organization
Search organizations by name or country...
Search
AI Research Institute
Visit Website
Researchers
1
Models
0
Datasets
0
Joined ago

Department of Mathematics, Informatics and Geoscience.

Course in

Scientific & Data-Intensive Computing

Thank for your attention

Presented by

Cristiano Baldassi