Hewlett Packard Enterprise

COMMON METADATA FRAMEWORK FOR TRUSTED AI

HPE AI Research Lab, Data Foundation for AI project team

Annmary Justine, Aalap Tripathy, Sergey Serebryakov, Cong Xu, Suparna Bhattacharya, Martin Foltin, Paolo Faraboschi

6/23/2022

AGENDA

- Data Foundation for AI
- Common Metadata Framework (CMF)
  - Architecture
- Integration with ecosystem of ml frameworks and ml tracking platforms

SELF-LEARNING DATA FOUNDATION FOR TRUSTWORTHY AI

A separate software Layer between MLOps Platforms and Storage (agnostic to both)

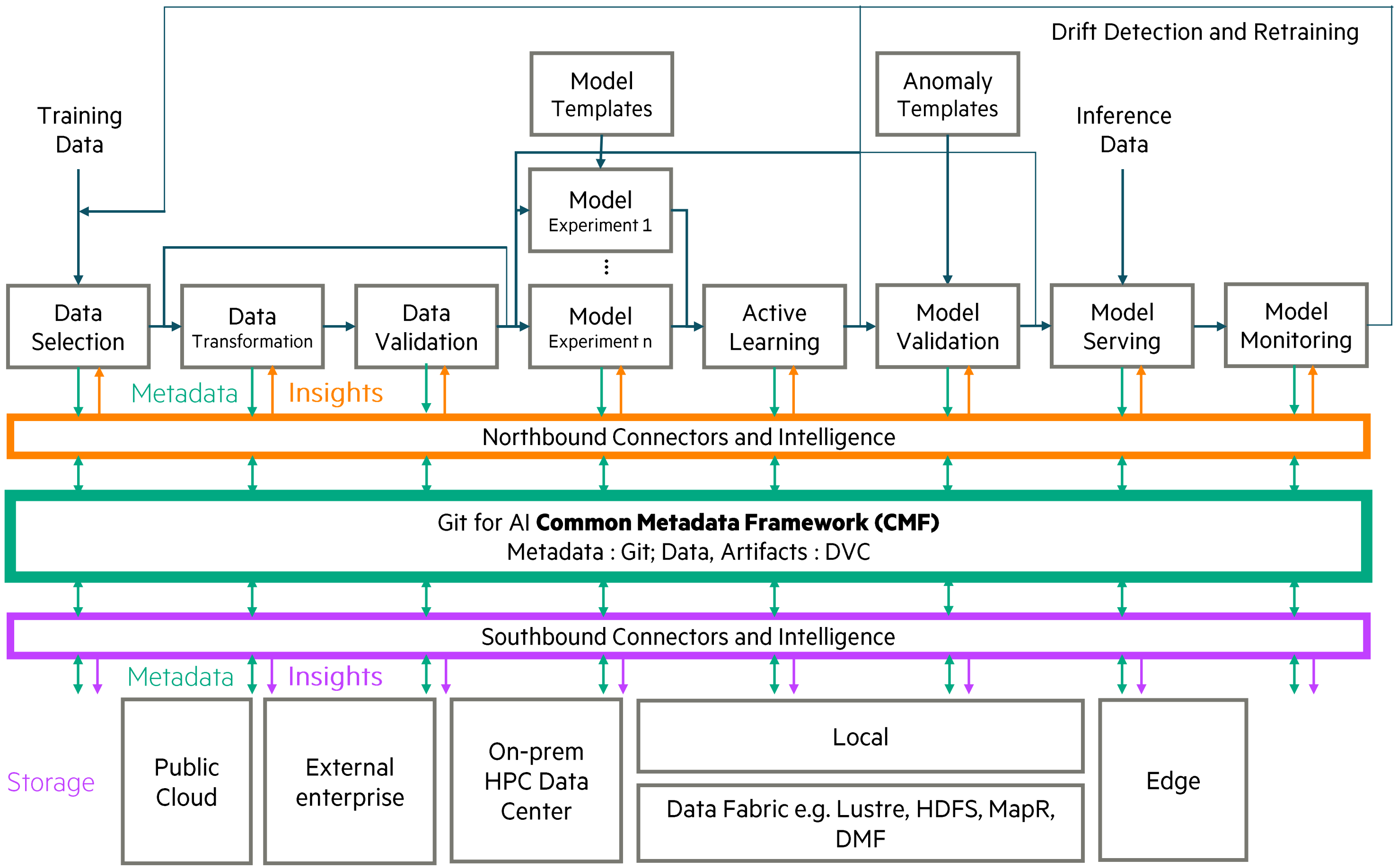
Motivation

- Building Trustworthy AI models can stress infrastructure and requires significant human effort
- Focus shifting from optimizing AI models to co-optimizing the data selection (good data rather than big data), labeling, transformations and models
- Optimization scope expanding from few pipeline stages (AI model training and test) to many stages, and across different pipelines
- Optimizations carried out for more metrics than before: besides model accuracy, the Trust features (robustness, explainability, and fairness) are critical

Meeting Trust objectives while co-optimizing AI models and data in complex pipelines requires infrastructure for tracking and managing data lineage and pipeline metadata

Example AI Pipeline

(running on Pytorch Lightening, MLFlow, Kubeflow, etc., MLOps Platform)

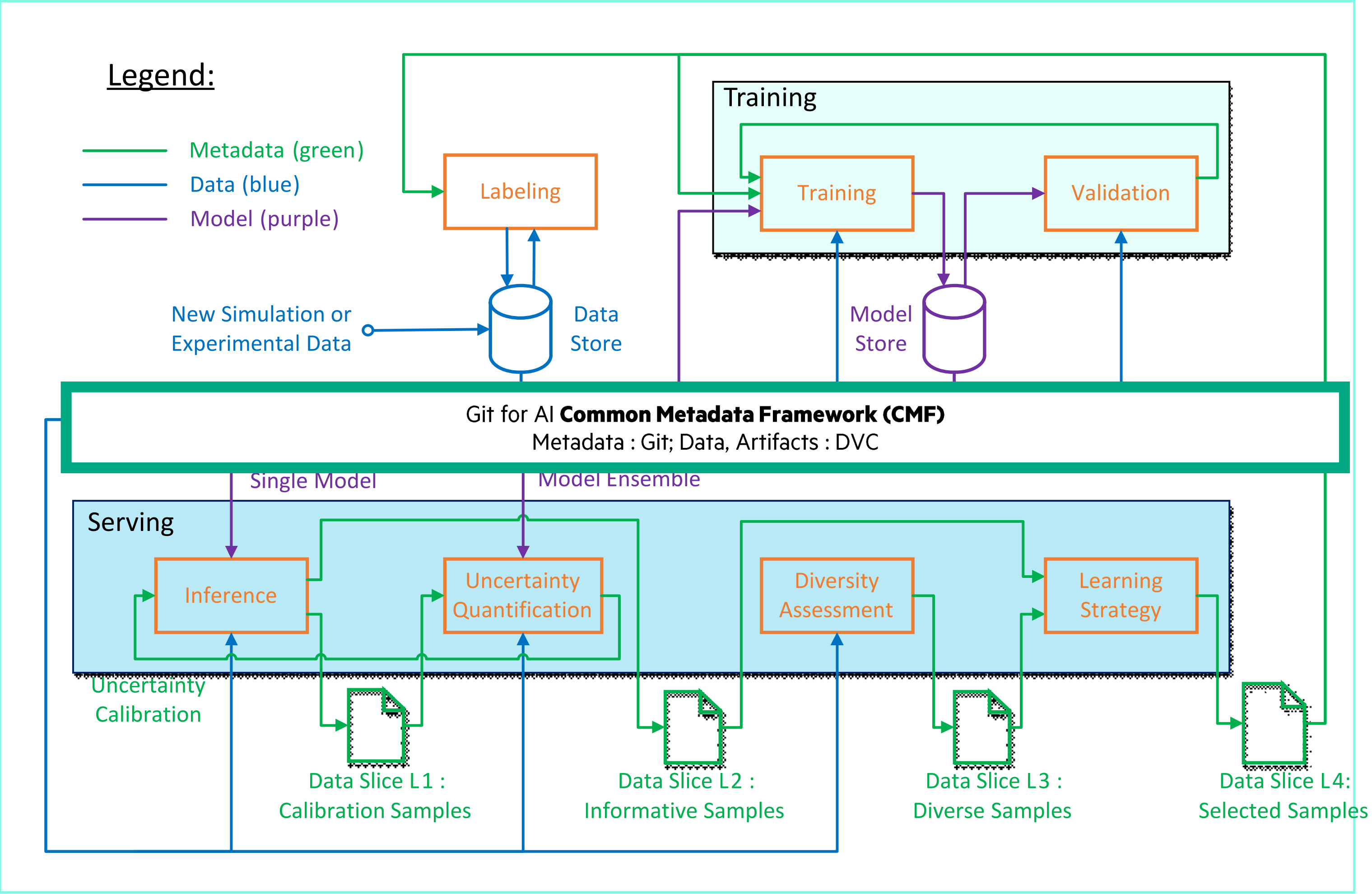


TRACKING AND MANAGING METADATA IN THE ERA OF DATA-CENTRIC AI
COMMON METADATA FRAMEWORK - Tracking metadata for complex linked AI pipelines

Challenges

- Multiple stages with interlinked dependencies and each stage could be executed in distributed asynchronous mode.
- Data centricity requires artifact lineages and tracking influence of different artifacts and data slices on model performance.
- Pipelines should be Reproducible , Auditable and Traceable.

Representative AI Pipeline

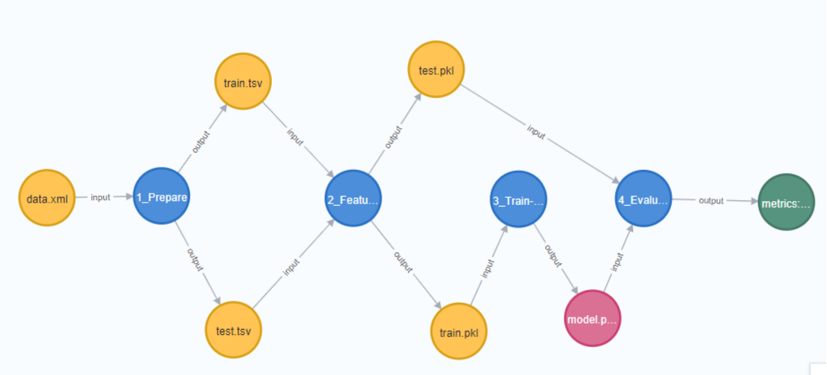


COMMON METADATA FRAMEWORK (CMF)

ML METADATA TRACKING TOOLS

	MLFLOW TRACKING	MLMD	CMF
COLLABORATIVE DEVELOPMENT			
Git like Support	No	No	Yes
DATACENTRIC SUPPORT			
Integrated Versioning	No	No	Yes
Identified by	Path / ID		Uniquely Identified by Hashes
Subset Identification	No	No	Yes
METRICS TRACKING			
Fine grained experiment tracking metrics	Yes	No	Yes
PIPELINE METADATA			
Pipeline Lineage	No	Yes	Yes

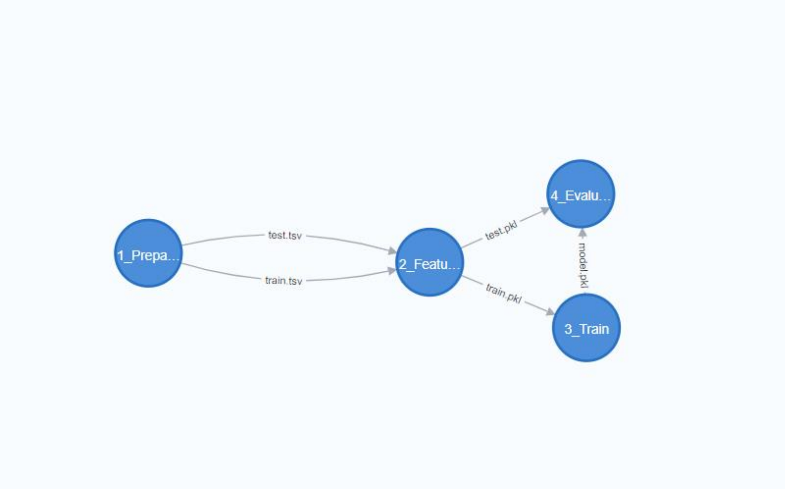
CMF TRACKS BOTH EXECUTION AND ARTIFACT LINEAGE



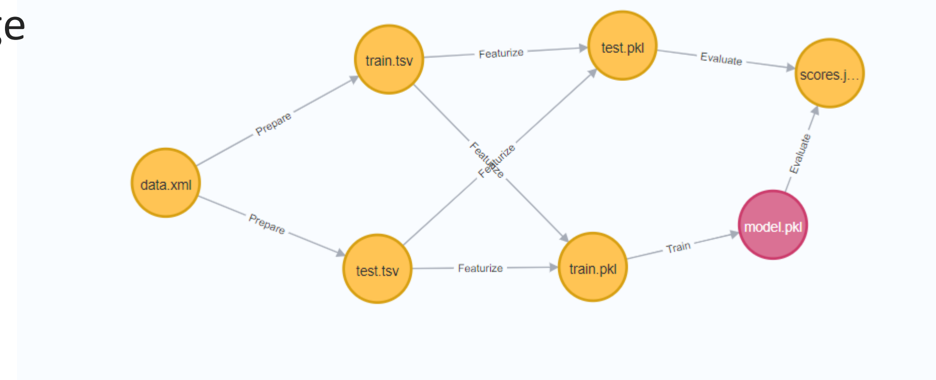
Lineage DAG

- Lineage stored as DAGs , Provenance

CMF Tracks both execution and artifact Lineage



Execution Lineage



Artifact Lineage

COMMON METADATA FRAMEWORK

Faster development of Models, increasing efficiency for Data Scientist, Accelerating new Science

Python library

```
metawriter = cmf.Cmf(filename="mlmd", pipeline_name="Test-env")
_ = metawriter.create_context(pipeline_stage="Prepare", custom_properties={"user-metadata1": "metadata_value"})
_ = metawriter.create_execution(execution_type="Prepare", custom_properties=params)
_ = metawriter.log_dataset(input_file, "input", custom_properties={"user-metadata1": "metadata_value"})
```

Example Metadata Collected

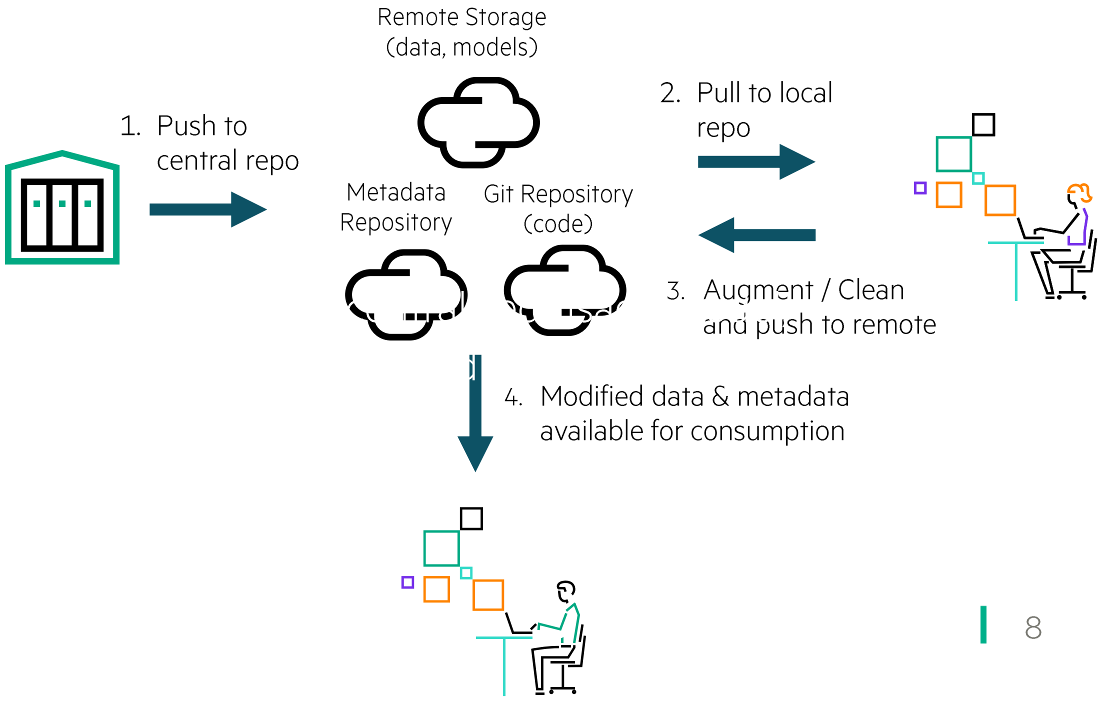
<id>	80
Commit	[commit 8abd7324ce3ed64e70c80d e2d86f987a158441ae]
Name	[MNIST]
Path	[data/MNIST:ab3353d41bd 7a24a20a31f29b64e3b3c. dir]
Type	[MNIST]
git_repo	[git@github.hpe.com:ann mary-roy/-deep-active-learning.git]
pipeline_id	[1]
pipeline_name	[active-learning]
uri	ab3353d41bd7a24a20a31f 29b64e3b3c.dir

Node Properties

Metrics

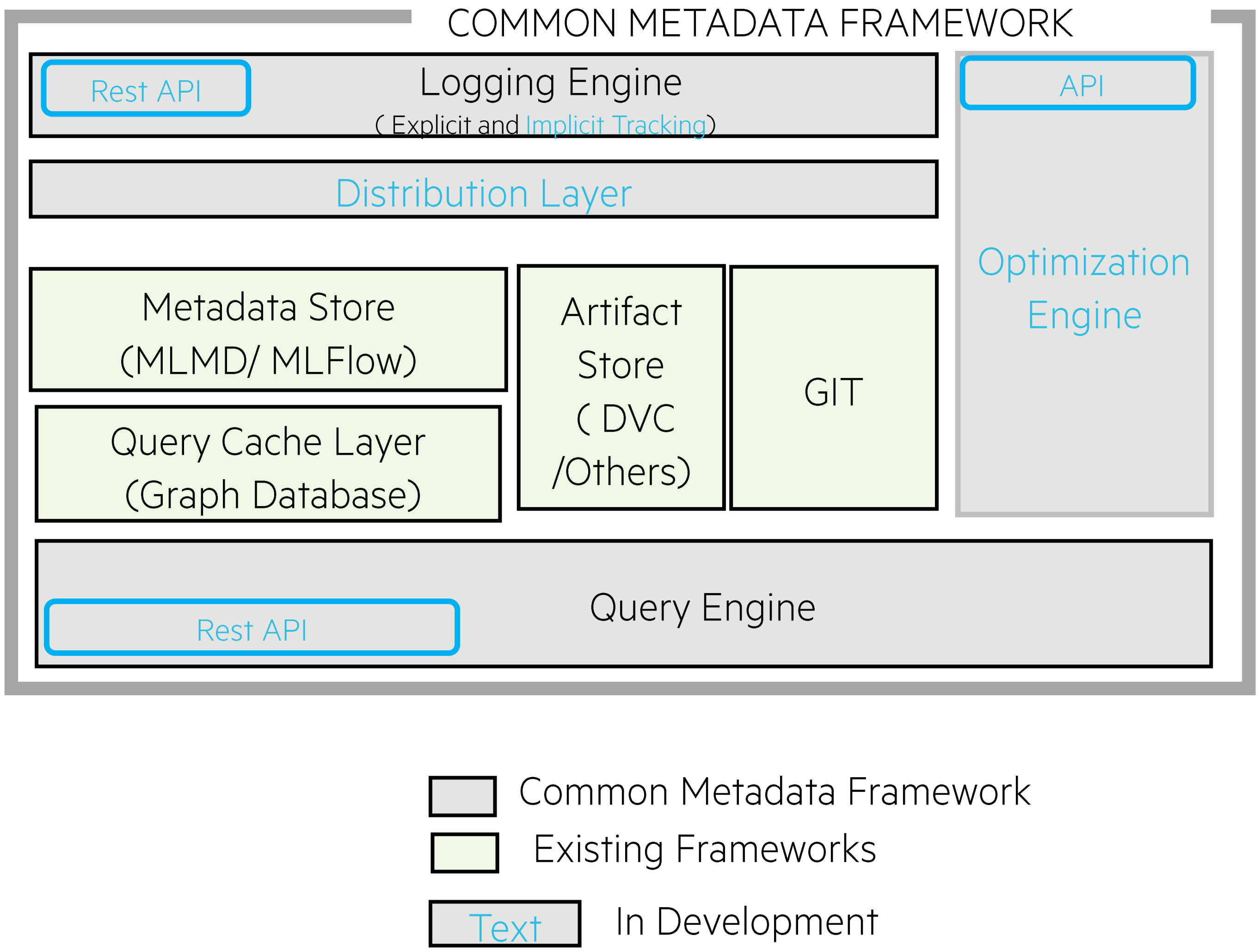
<id>	141
Name	Test:a6d47ed4-d757-11ec-b8fa-b47af137252e:7
accuracy	0.9682
pipeline_id	1
pipeline_name	active-learning
uri	a6d47ed4-d757-11ec-b8fa-b47af137252e

DECENTRALIZED USAGE MODEL,
EASILY CLONED IN ANY ENVIRONMENT



COMMON METADATA FRAMEWORK (CMF) ARCHITECTURE

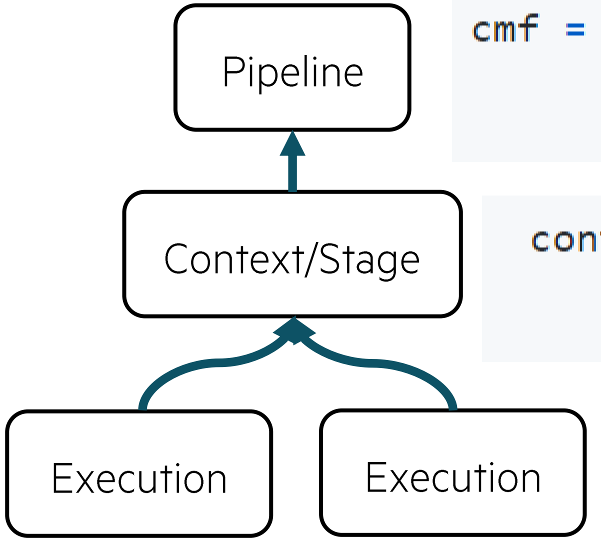
UNIFIED MANAGEMENT OF CODE, DATA AND METADATA



CMF API

Execution

- Pipeline, Stage, Executions, Asynchronous and distributed




```
cmf = cmf.Cmf(filename="mlmd",
                                    pipeline_name="Test-env")
```

```
context = cmf.create_context(pipeline_stage="Prepare",
                    custom_properties={"user-metadata1":"metadata_value"})
```

```
execution = cmf.create_execution(execution_type="Prepare-1",
            custom_properties = {"user-metadata1":"metadata_value"})
```

CMF API

- Stored in content addressable repo with help of DVC + GIT
- Versioned Automatically
- Identified by content hash

Dataset

```
cmf.log_dataset(input, "input", custom_properties={"user-metadata1":"metadata_value"})
cmf.log_dataset(output_train, "output", custom_properties={"user-metadata1":"metadata_value"})
```

Dataslice

```
dataslice = cmf.create_dataslice("slice-a")
for i in range(1,20,1):
    j = random.randrange(100)
    dataslice.add_data("data/raw_data/"+str(j)+".xml")
dataslice.commit()
```

Model

```
cmf.log_model(path="model.pkl", event="output", model_framework="SKlearn",
cmf.log_model(path="model.pkl", event="input", model_framework="SKlearn",
```

CMF API

Metrics

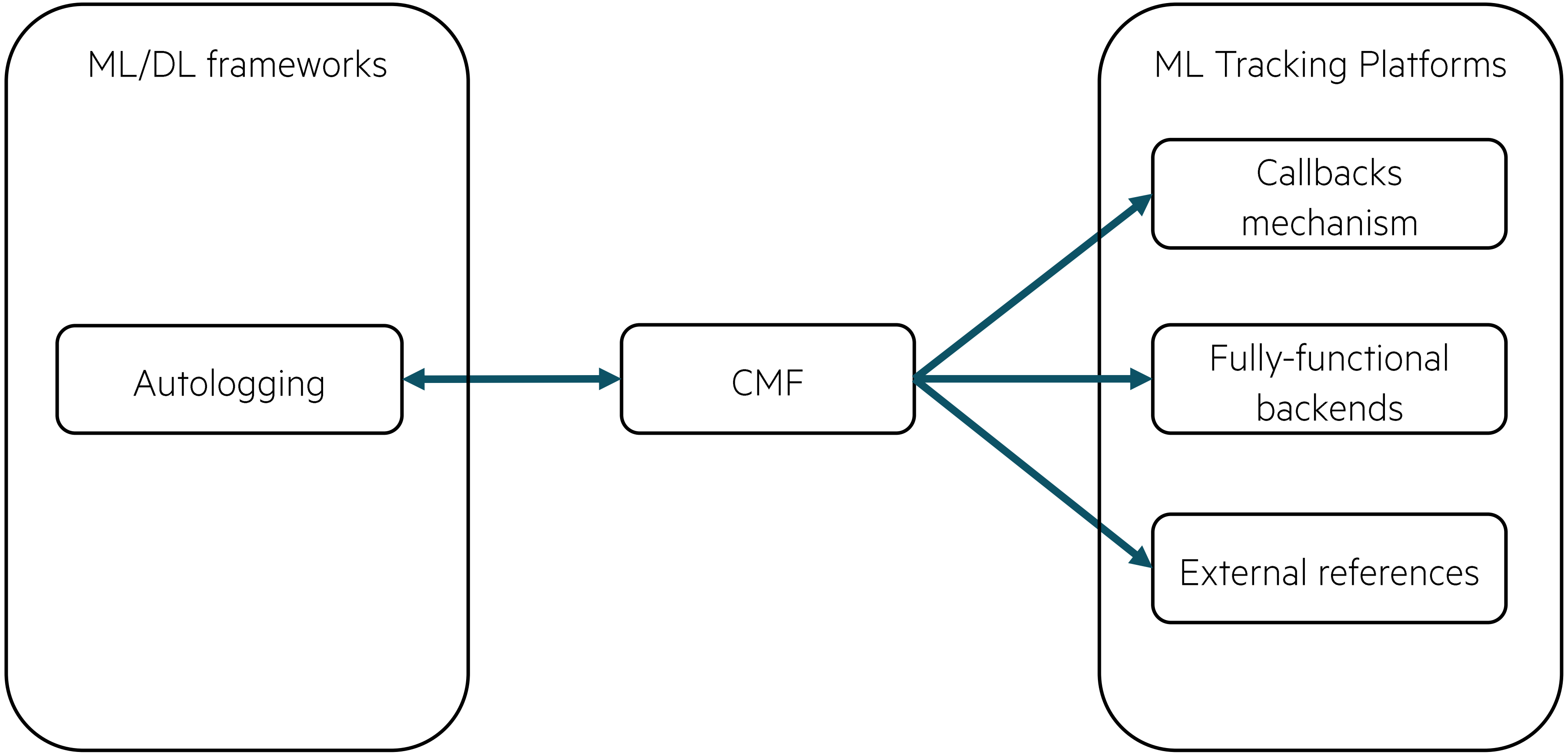
- Scalable
- Support for Consolidated output metric and fine drill down to per step metrics

```
while True: #Inside training loop
     cmf.log_metric("training_metrics", {"loss":loss})
cmf.commit_metrics("training_metrics")
```

```
cmf.log_execution_metrics("metrics", {"avg_prec":avg_prec, "roc_auc":roc_auc})
```

INTEGRATION WITH ECOSYSTEM OF ML FRAMEWORKS AND ML TRACKING PLATFORMS

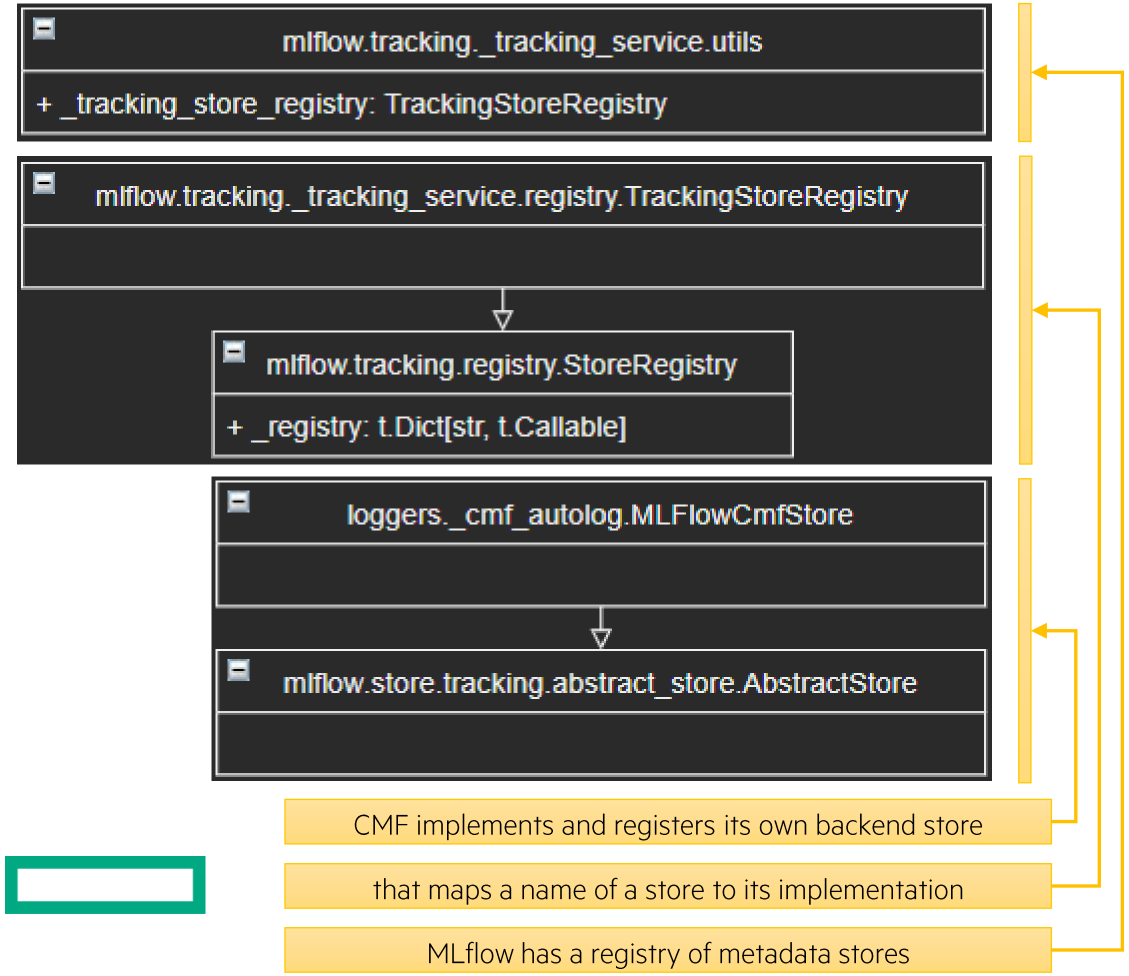
PROOF-OF-CONCEPT IMPLEMENTATIONS



AUTOLOGGING WITH ML/DL FRAMEWORKS

- MLflow (and other frameworks) already integrate with many Machine Learning and Deep Learning frameworks. Let's try to use it!

MLflow components related to backend stores

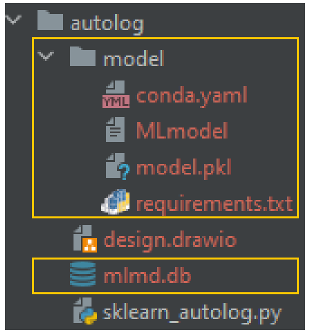


Code example

```python
import pathlib
from mdflow.loggers._cmf_autolog import autolog
import numpy as np
from sklearn.linear_model import LinearRegression

def main() -> None:
    autolog(
        cmf_cfg={
            'tracking_uri': (pathlib.Path.cwd() / 'mlmd.db').as_posix(),
            'pipeline_name': 'sklearn-lr',
            'graph': False
        },
        ctx_cfg={
            'pipeline_stage': 'train',
            'custom_properties': {}
        },
        exec_cfg={
            'execution_type': 'train',
            'custom_properties': {}
        }
    )

    x = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])
    y = np.dot(x, np.array([1, 2])) + 3

    model = LinearRegression()
    model.fit(x, y)
```

Standard scikit-learn code

Enable autologging
("from cmf import autolog")



INTEGRATION WITH TRACKING PLATFORMS: CALLBACKS

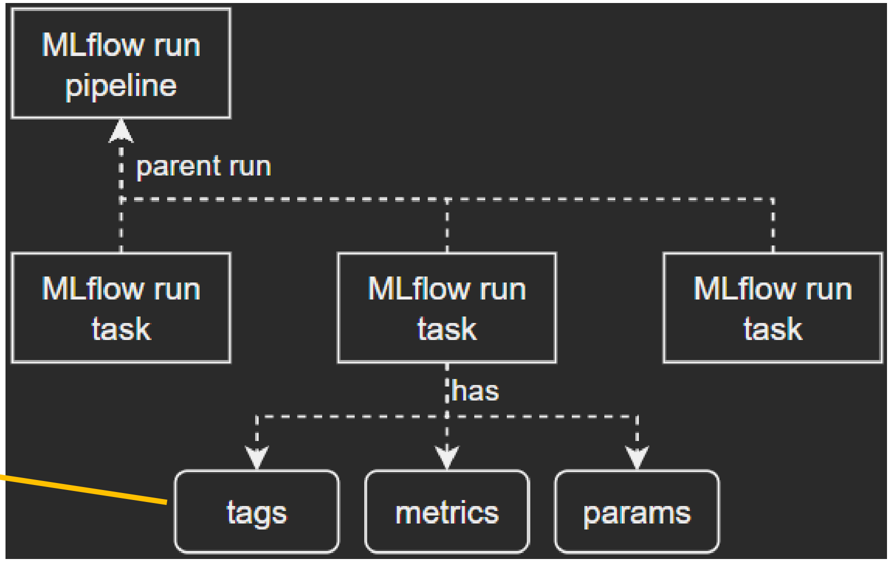
- Reporting: integration via callbacks.
  - Idea: similar to how Ray Tune integrates with MLflow (one example).
  - Status: PoC implementation
  - How: call any registered callback whenever CMF API is called.
  - Why: could be useful for visualization and quick analysis with standard MLflow tools (search …).
  - Important: CMF uses its own metadata (MLMD) and artifact(DVC) stores as primary store options.

```
class Callback(object):
    def on_log_dataset(self):
        ...


class Neo4jCallback(Callback):
    ...


class MLFlowCallback(Callback):
    ...


class Cmf(object):
    def __init__(self):
        self._callbacks: t.List[Callback] = []

    def add_callback(self, callback: Callback)
        self._callbacks.append(callback)

    def log_dataset(self):
        for callback in self._callbacks:
            callback.on_log_dataset()
```

INTEGRATION WITH TRACKING PLATFORMS: FULLY-FUNCTIONAL BACKENDS

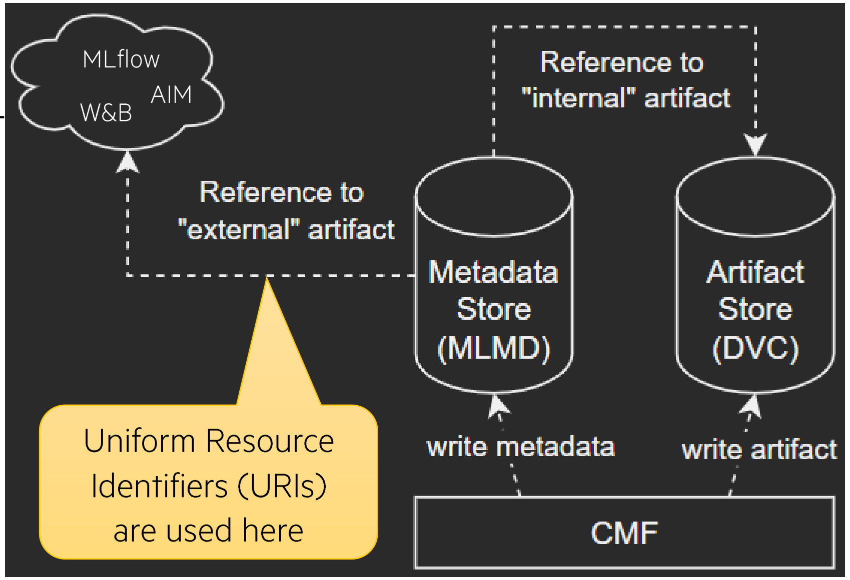
- Fully-functional backends to replace MLMD.
  - Idea: similar to how MLflow can use different backends (FileStore, RestStore and SqlAlchemyStore).
  - Status: MLflow-based PoC implementation with limited features available.
  - How: completely replace MLMD backend with something else (e.g., MLflow).
  - Why: Organizations and teams may already be using some tracking library.
  - Important: Could be tricky to support all features available in MLMD, maybe does not make too much sense (need to be opinionated about certain things).

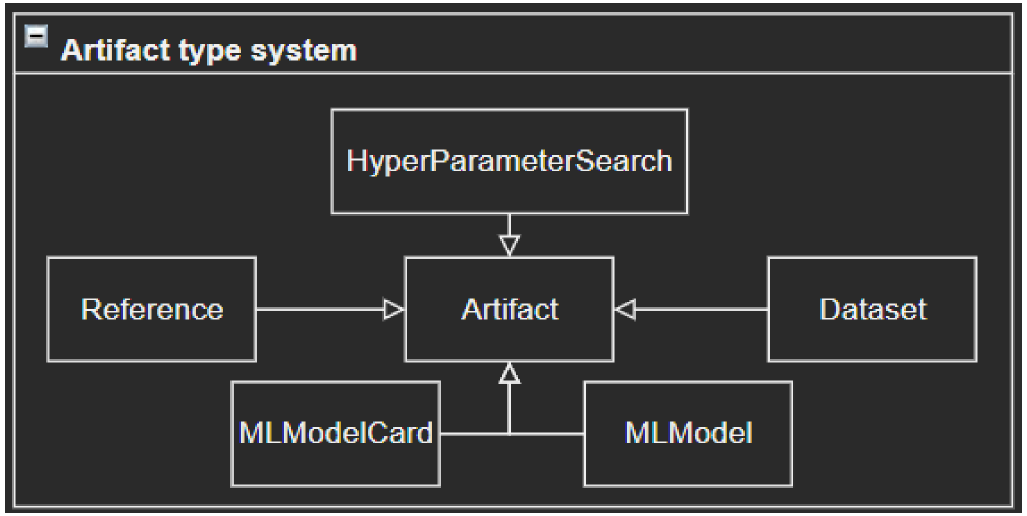
```
class ExecTag(object):
    """Execution tags used to track pipeline metadata."""
    TYPE = "cmf.execution.type"
    """One of: `pipeline` or `task` (see ExecType)."""
    NAME = "cmf.execution.name"
    """Task name, by default, equals to a task function name."""
    STATE = "cmf.execution.state"
    """Current execution state (see ExecState)."""
    LABELS = "cmf.execution.labels"
    """Set of labels for the execution record."""
    PARAMS = "cmf.execution.params"
    """Dictionary of execution parameters."""
    INPUT = "cmf.execution.input"
    """Dictionary of input artifacts (str -> artifact)."""
    OUTPUT = "cmf.execution.output"
    """Dictionary of output artifacts (str -> artifact)."""
```

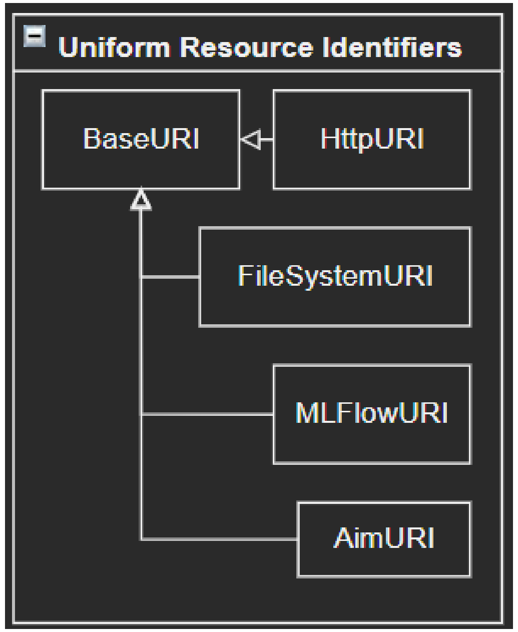


INTEGRATION WITH TRACKING PLATFORMS: EXTERNAL REFERENCES

- Referencing externally-hosted artifacts.
  - Idea: let MLflow, AIM or W&B host artifacts, parameters and metrics for all or subset of pipeline tasks.
  - Status: MLflow-based PoC implementation available, AIM-based PoC is in progress.
  - How: introduce new API to CMF, e.g., `log_artifact`. Idea is to use an artifact type system.
  - Why: Organizations and teams may already be using some tracking libraries.
  - Important: CMF just references external artifacts; it does not own them.







https://storage.com/tf-keras-datasets/mnist.npz

file:///C:/Users/ds/.cmf/data/mnist/mnist.npz

mlflow:///runs/e4066de5ff3840a98f00e9314eac3ac5/mnist.npz
mlflow:///experiments/2?tags.ray.run_name=%22RUN_ID%22
mlflow:///models/cdu_01/production

aim:///runs/RUN_HASH

THANK YOU

annmary.roy@hpe.com
aalap.tripathy@hpe.com
sergey.serebryakov@hpe.com
cong.xu@hpe.com
suparna.bhattacharya@hpe.com
martin.foltin@hpe.com
paolo.faraboschi@hpe.com

CMF DEMO

Notes:

- Toy example with more scenarios: Different hyper-parameters. Different versions of the same dataset.
- Lead to more complex use cases
- Data transformation after serving: opportunistic

CMF DEMO

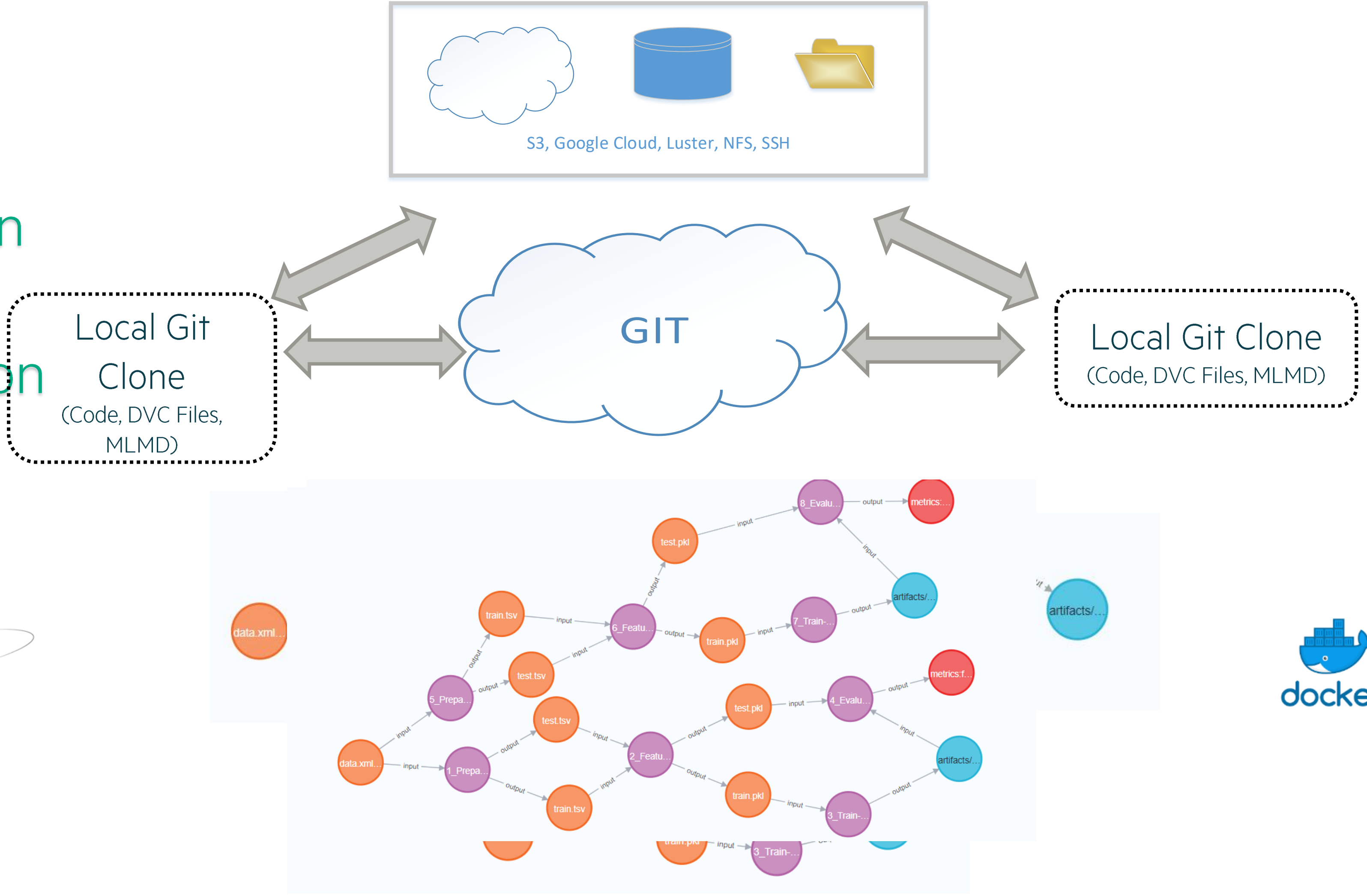
COLLABORATE ON A PIPELINE AND SHARE METADATA

Text classification with Random Forest Classifier

Data Preparation

Feature Extraction

Train

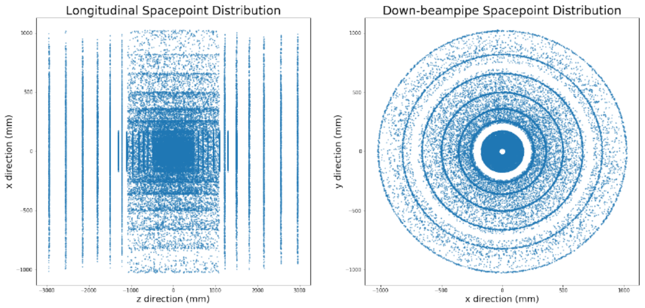


Evaluate

USE CASES

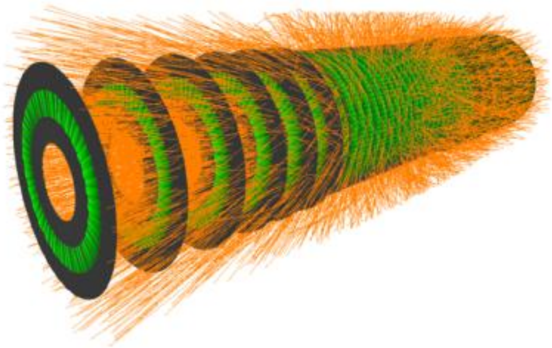
OBJECTIVES

- Faster development of Trustworthy AI models in complex pipelines



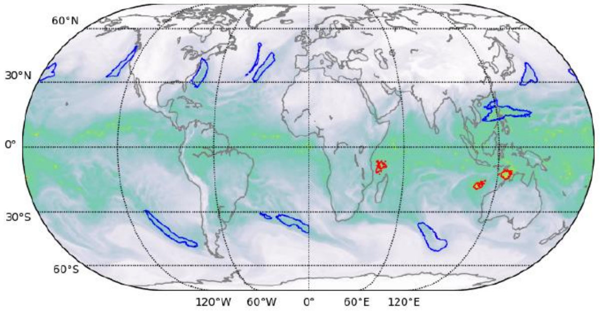
Exa.trkX
High Energy Physics
Particle Trajectory Reconstruction

(Problem example from
Eur. Phys. J. C **81** (2021), 876
and arXiv 2103.06695 by Ju et. al.)



- Reduction of AI model training time

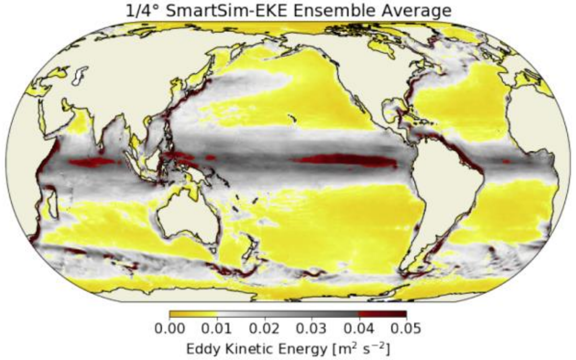
DeepCam
Extreme Weather
Feature Identification



(Problem example from SC'18 and arXiv 1810.01993 by Kurth et. al.)

- Trustworthy AI models for acceleration of complex simulations

Ocean Climate Modeling
MOM6 with EKE Surrogate Models



(Problem example from arXiv 2104.09355 by Partee et. al.)

- Reduction of data labeling effort for supervised learning (looking for collaboration opportunities)
- Reduction of experiment or simulation data volume required to build Trustworthy AI models

PROBLEM STATEMENT

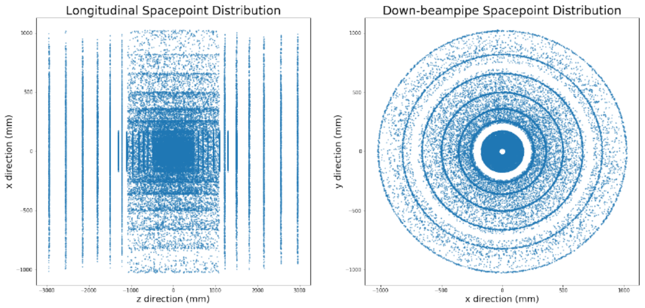
- Building Trustworthy AI models often requires large volumes of data, stressing AI compute and storage infrastructures and requiring significant human efforts to label the data and optimize AI models
  - Example: DeepCam
- Optimization scope expanding from few pipeline stages (AI model training and test) to many stages, and across different pipelines
  - Example: Exa.Trkx
- Optimizations carried out for more metrics than before: besides model accuracy, the Trust features (robustness, explainability) are critical
  - Example: Ocean Climate Modeling
- Focus shifting from optimizing AI models to co-optimizing the data selection (good data rather than big data), data labeling, data transformations, and AI models (the Data-centric AI paradigm)

$\Rightarrow$ **Meeting Trust objectives while co-optimizing AI models and data in complex pipelines requires:**

- infrastructure for tracking and managing data lineage and pipeline metadata
- intelligence to learn from this metadata to optimize data and pipelines

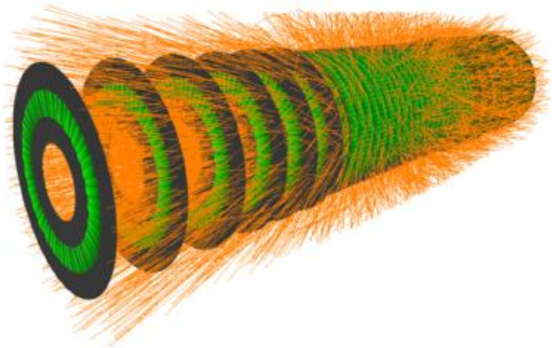
EXAMPLE USE CASE 1

- Faster development of Trustworthy AI models in complex pipelines



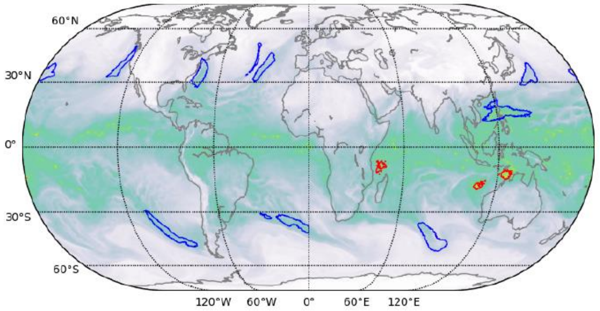
Exa.trkX
High Energy Physics
Particle Trajectory Reconstruction

(Problem example from
Eur. Phys. J. C **81** (2021), 876
and arXiv 2103.06695 by Ju et. al.)



- Reduction of AI model training time

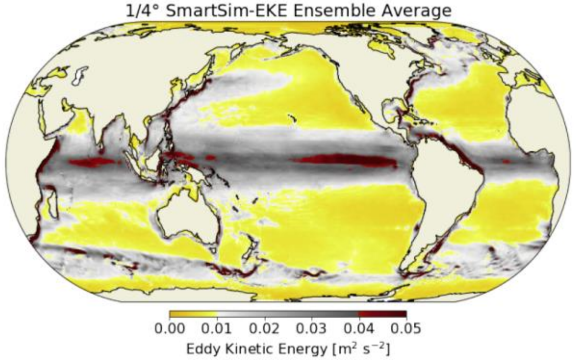
DeepCam
Extreme Weather
Feature Identification



(Problem example from SC'18 and arXiv 1810.01993 by Kurth et. al.)

- Trustworthy AI models for acceleration of complex simulations

Ocean Climate Modeling
MOM6 with EKE Surrogate Models

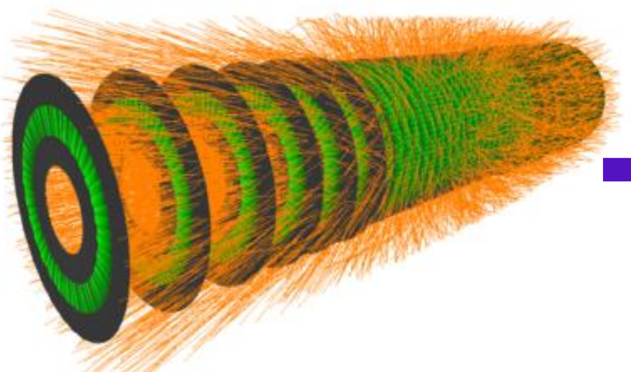


(Problem example from arXiv 2104.09355 by Partee et. al.)

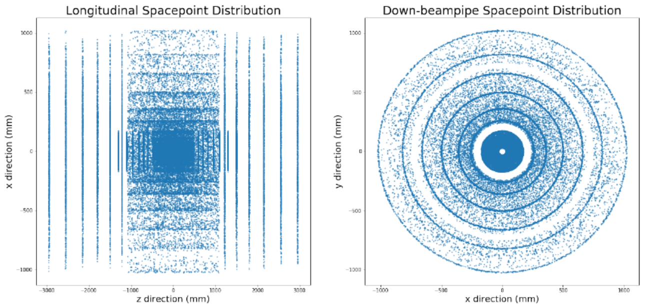
- Reduction of data labeling effort for supervised learning (looking for collaboration opportunities)
- Reduction of experiment or simulation data volume required to build Trustworthy AI models

Exa.trkX High Energy Physics Particle Reconstruction

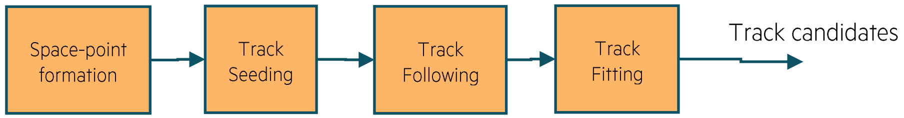
Challenges



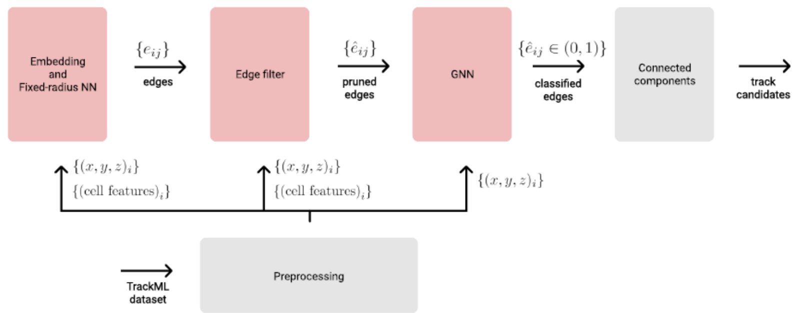
Collider Experiments
High Luminosity Large Hardon Collider (HL-LHC), ATLAS, CMS, …..



Source: Ju, Murnane et al. Performance of a Geometric Deep Learning Pipeline for HL-LHC Particle Tracking, Eur. Phys. J., 2021



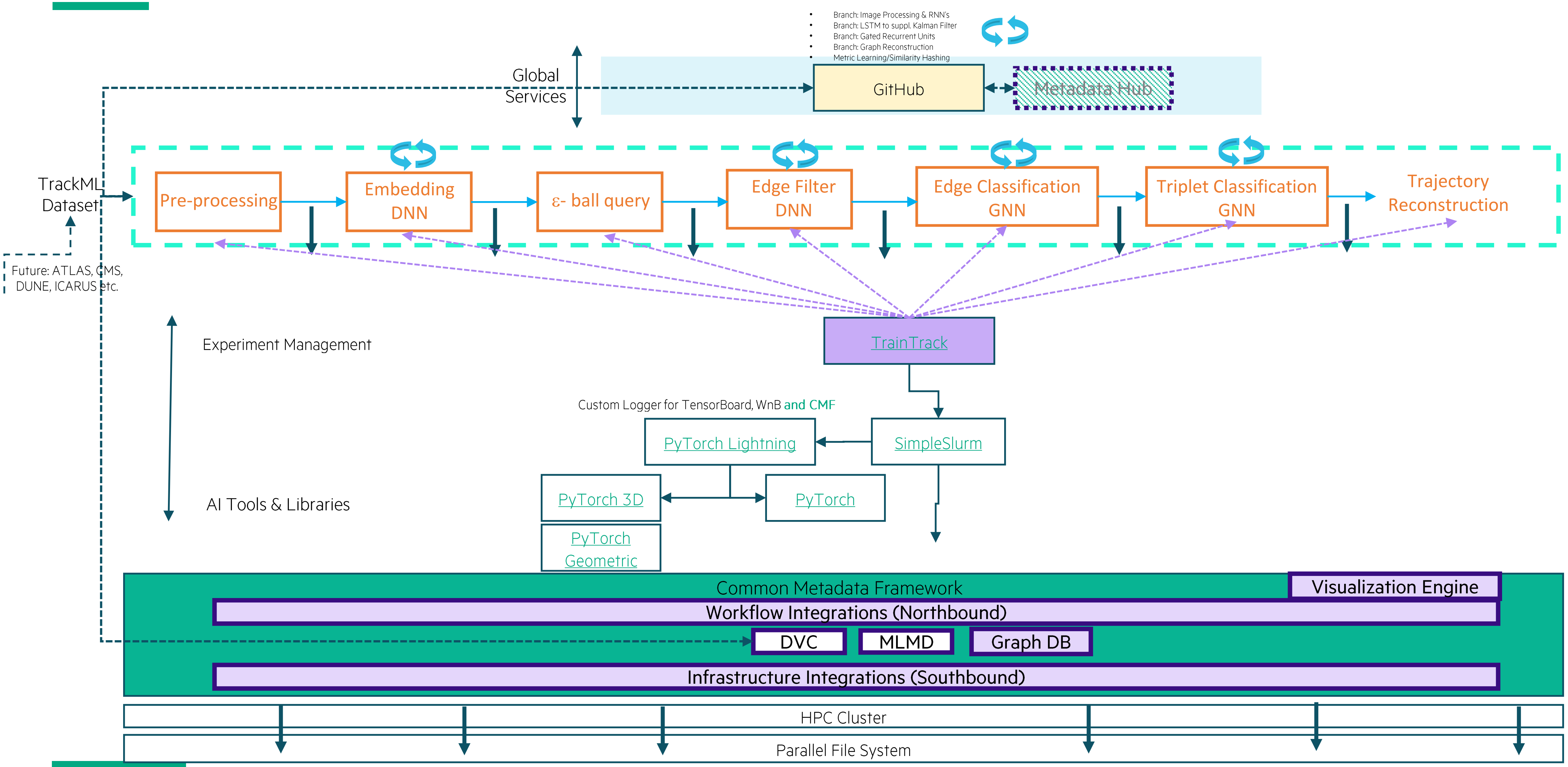
Offline Particle Tracking
(Physics-based/Deterministic. Large computational cost. Slow)



Online Particle Tracking
(GNN/Metric Learning Particle Tracking Algorithms)

- Particle Tracking algorithm
  - Linked experiment stages: Many **execution lineages, artifact lineages, metrics, models, and config parameters**
  - Long experiment times
  - Purity-efficiency tradeoff

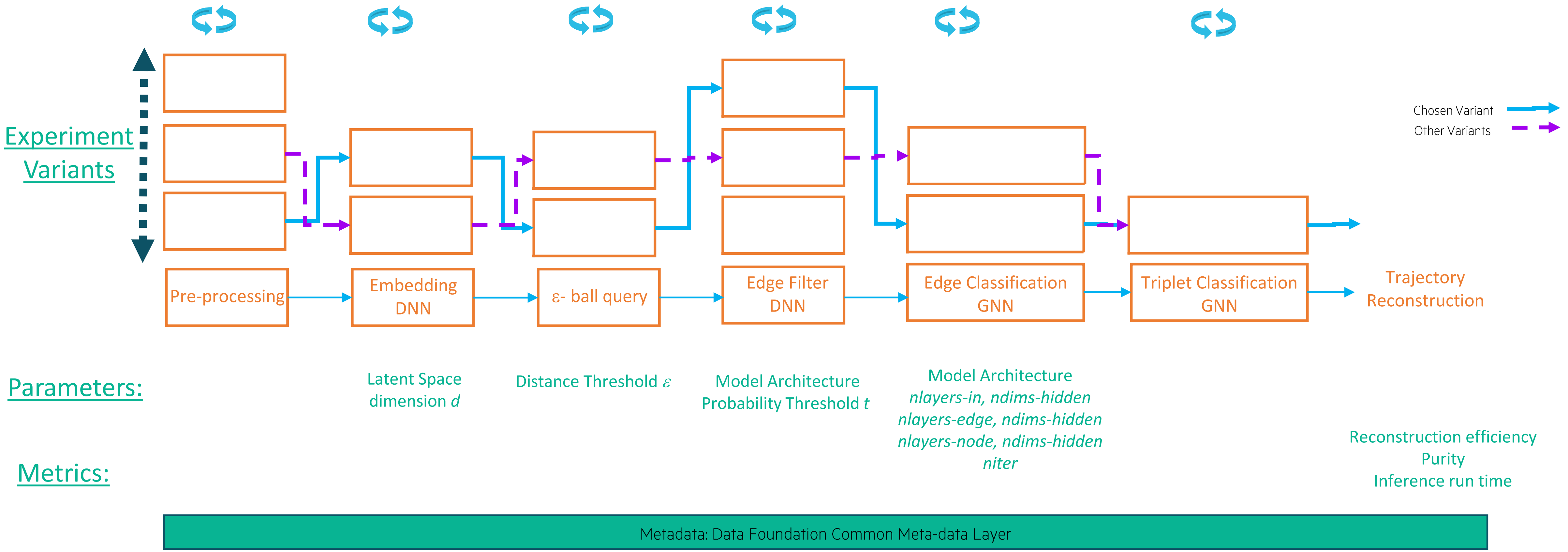
Instrumenting Exa.trkX HEP Particle Reconstruction Pipeline with CMF



ENABLING AI MODEL OPTIMIZATION FOR COMPLEX LINKED AI PIPELINES

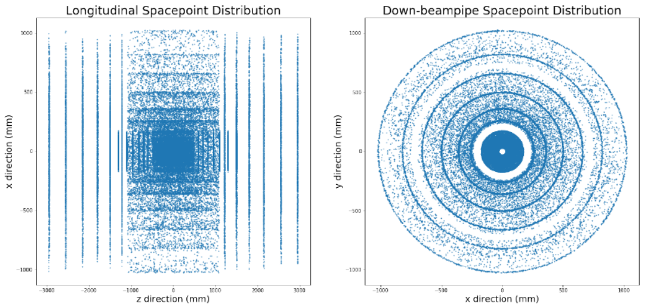
Faster development of Models, increasing efficiency for Data Scientist, Accelerating new Science

- CMF helps tracks dependencies of output Metrics on Parameters across **all variations of pipeline stages**
- Capturing **data lineage**, **metrics**, **network architecture** & **parameters** enables <u>end-to-end visibility</u>, <u>reproducibility</u>, and drives <u>optimization</u>

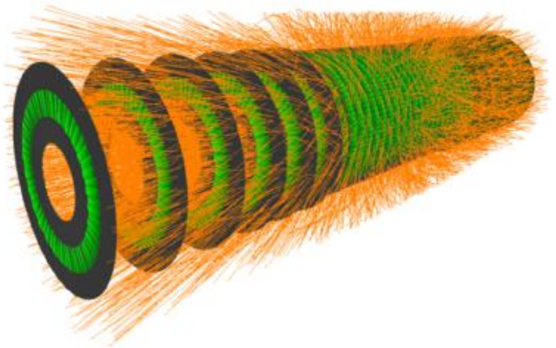


EXAMPLE USE CASE 2

- Faster development of Trustworthy AI models in complex pipelines



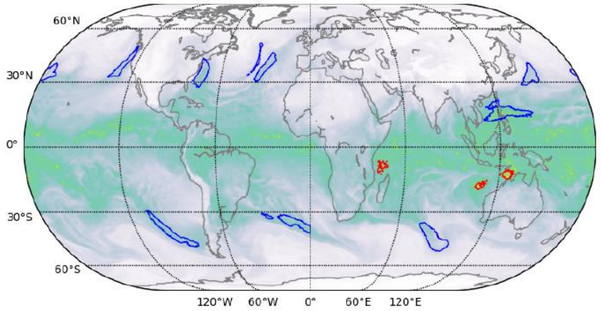
Exa.trkX
High Energy Physics
Particle Trajectory Reconstruction



(Problem example from
Eur. Phys. J. C **81** (2021), 876
and arXiv 2103.06695 by Ju et. al.)

- Reduction of AI model training time

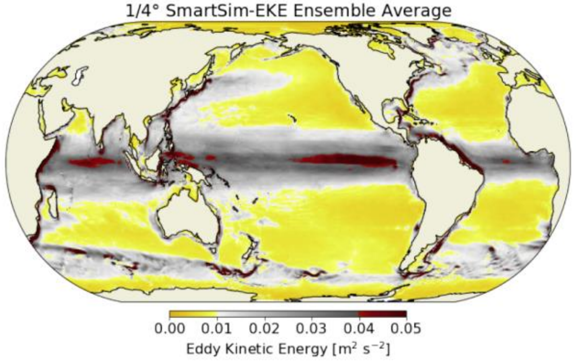
DeepCam
Extreme Weather
Feature Identification



(Problem example from SC'18 and arXiv 1810.01993 by Kurth et. al.)

- Trustworthy AI models for acceleration of complex simulations

Ocean Climate Modeling
MOM6 with EKE Surrogate Models



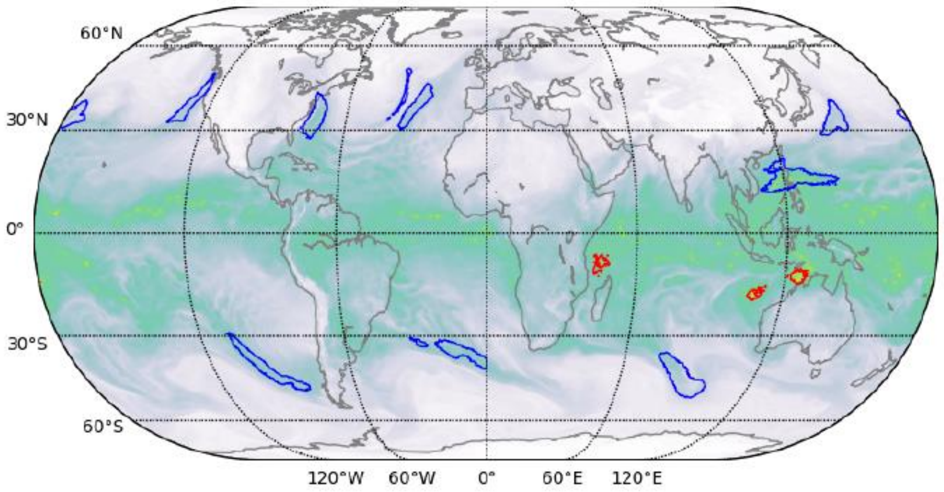
(Problem example from arXiv 2104.09355 by Partee et. al.)

- Reduction of data labeling effort for supervised learning (looking for collaboration opportunities)
- Reduction of experiment or simulation data volume required to build Trustworthy AI models

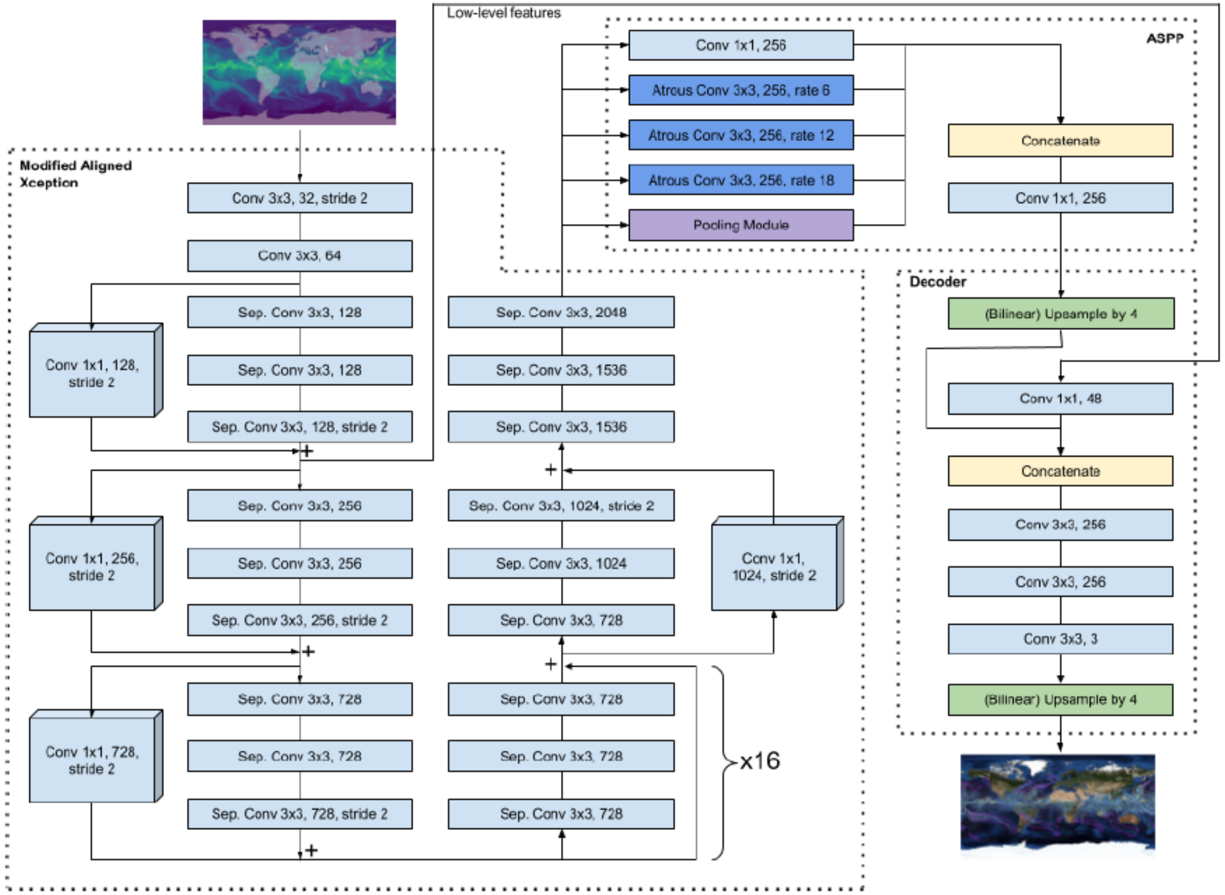
DeepCam atmospheric image segmentation

Challenges

- Long model training time
  - I/O data movement bottleneck: 57 MB/image, 6.3 TB/dataset
  - Complex model: >6 GB GPU memory, ~3 TFLOP forward pass
- High labeling effort and Lack of uncertainty quantification
  - Disparity of labels from different auto-labeling tools
  - Inference mean IoU quality of only ~0.73



Labeled Atmospheric rivers (blue) and tropical cyclones (red)
(Kurth et. al., SC'18 and arXiv 1810.01993)

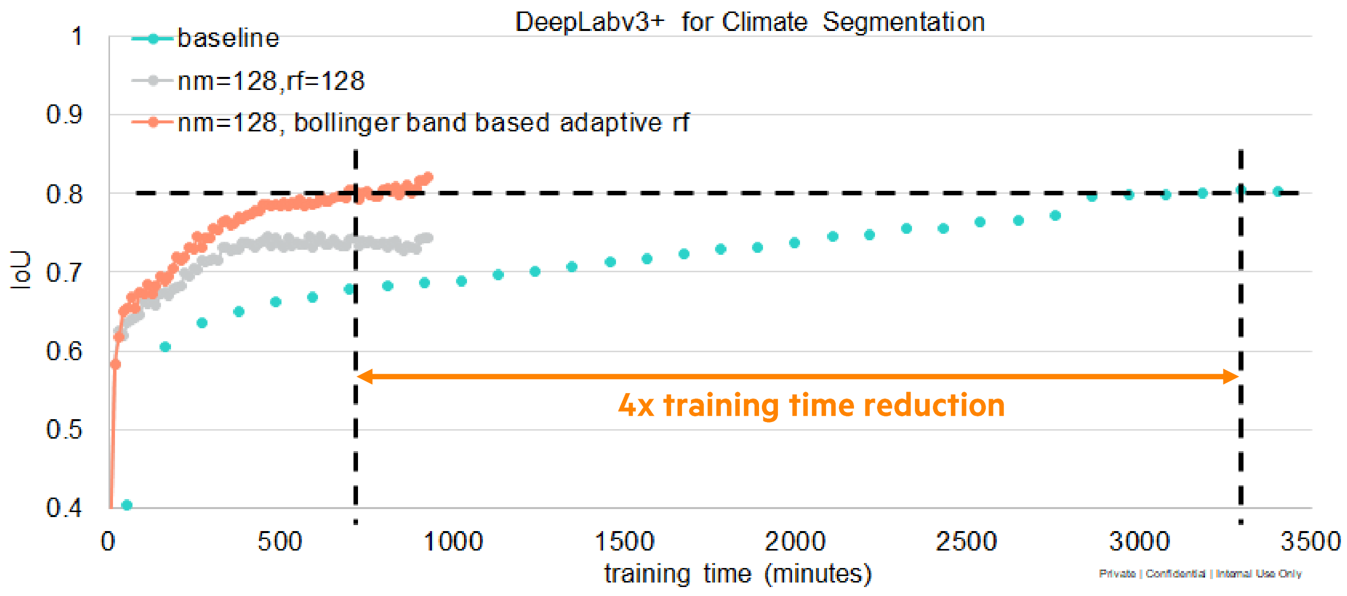


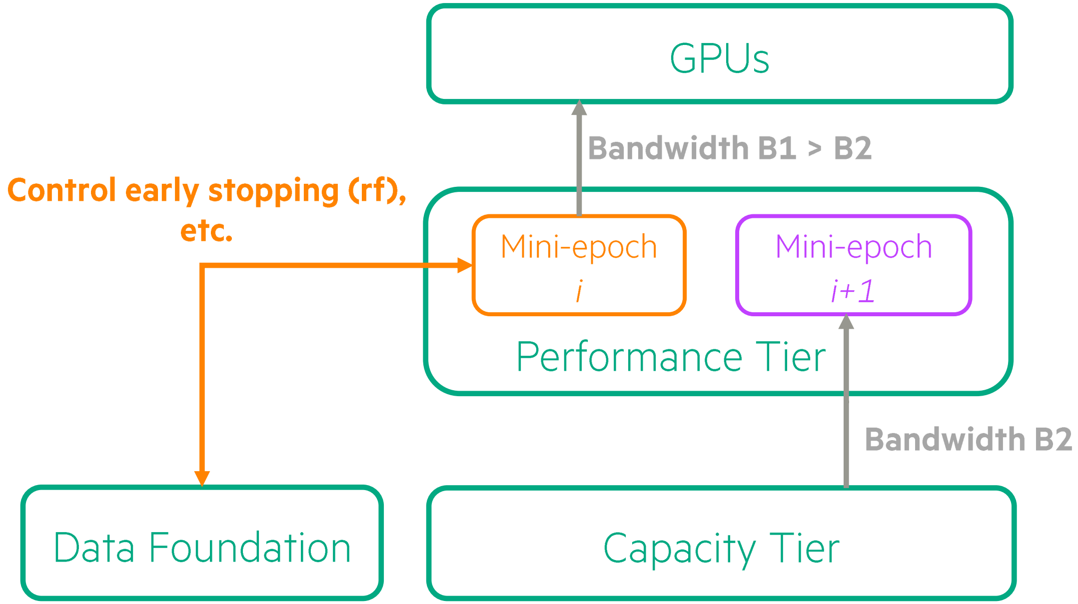
Complex climate segmentation model - DeepLabv3+
(Geosci. Model Dev. 14, 107-124, 2021 : gmd-14-107-2021)

DeepCam atmospheric image segmentation

Reduce model training time

- Model training can be storage I/O bandwidth limited
- Bandwidth and data aware, and training convergence conscious training computations hide data movement latency
- Data Foundation intelligence monitors accuracy to adjust computation repeating factor dynamically
- Example: ~4x training time reduction on 4-nodes from Cori-GPU: 930 GB, 6.8 GB/s NVMe / node, 500 MB/s HDD / node

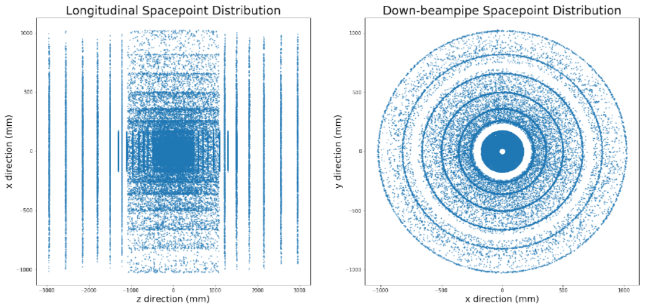




(http://www.pdsw.org/pdsw21/papers/ws_pdsw_paper_S3_P1_paper-xu.pdf)

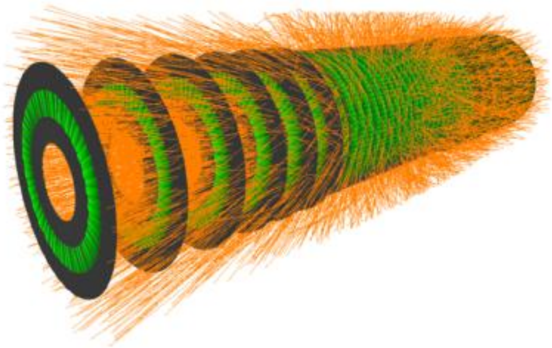
EXAMPLE USE CASE 3

- Faster development of Trustworthy AI models in complex pipelines



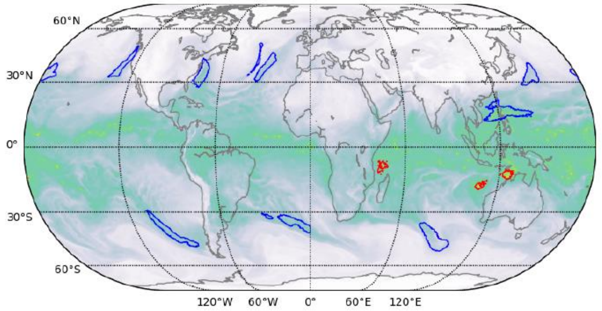
Exa.trkX
High Energy Physics
Particle Trajectory Reconstruction

(Problem example from
Eur. Phys. J. C **81** (2021), 876
and arXiv 2103.06695 by Ju et. al.)



- Reduction of AI model training time

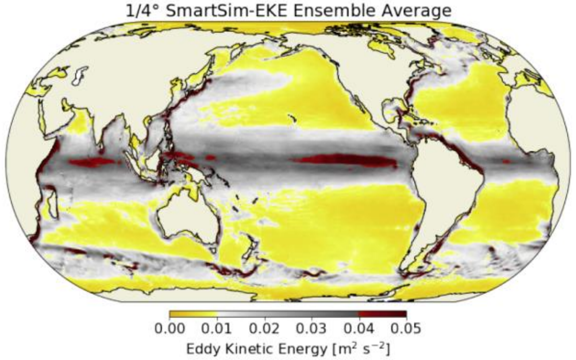
DeepCam
Extreme Weather
Feature Identification



(Problem example from SC'18 and arXiv 1810.01993 by Kurth et. al.)

- Trustworthy AI models for acceleration of complex simulations

Ocean Climate Modeling
MOM6 with EKE Surrogate Models



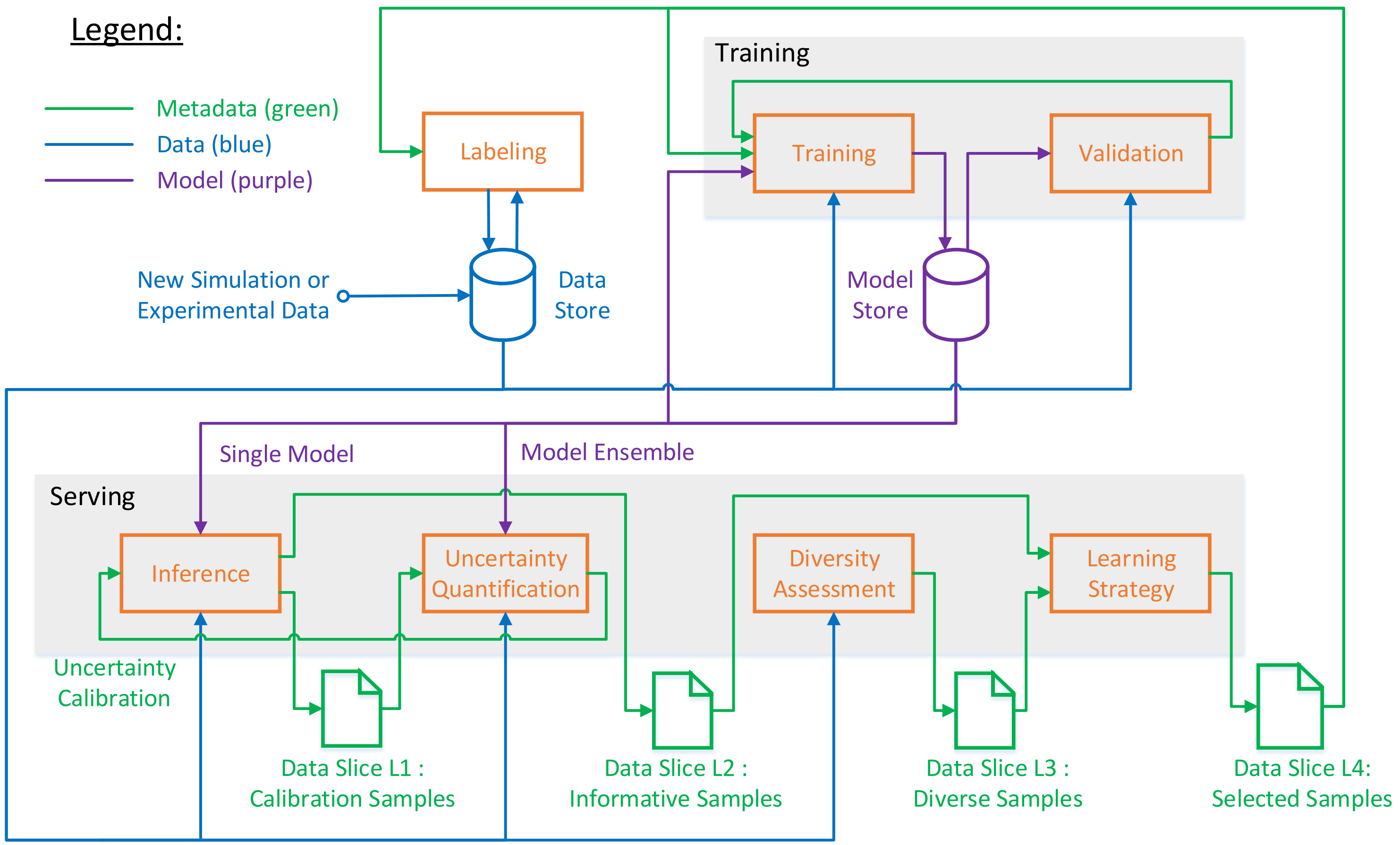
(Problem example from arXiv 2104.09355 by Partee et. al.)

- Reduction of data labeling effort for supervised learning (looking for collaboration opportunities)
- Reduction of experiment or simulation data volume required to build Trustworthy AI models

DeepCam atmospheric image segmentation

Reduce data labeling effort. Quantify prediction uncertainty

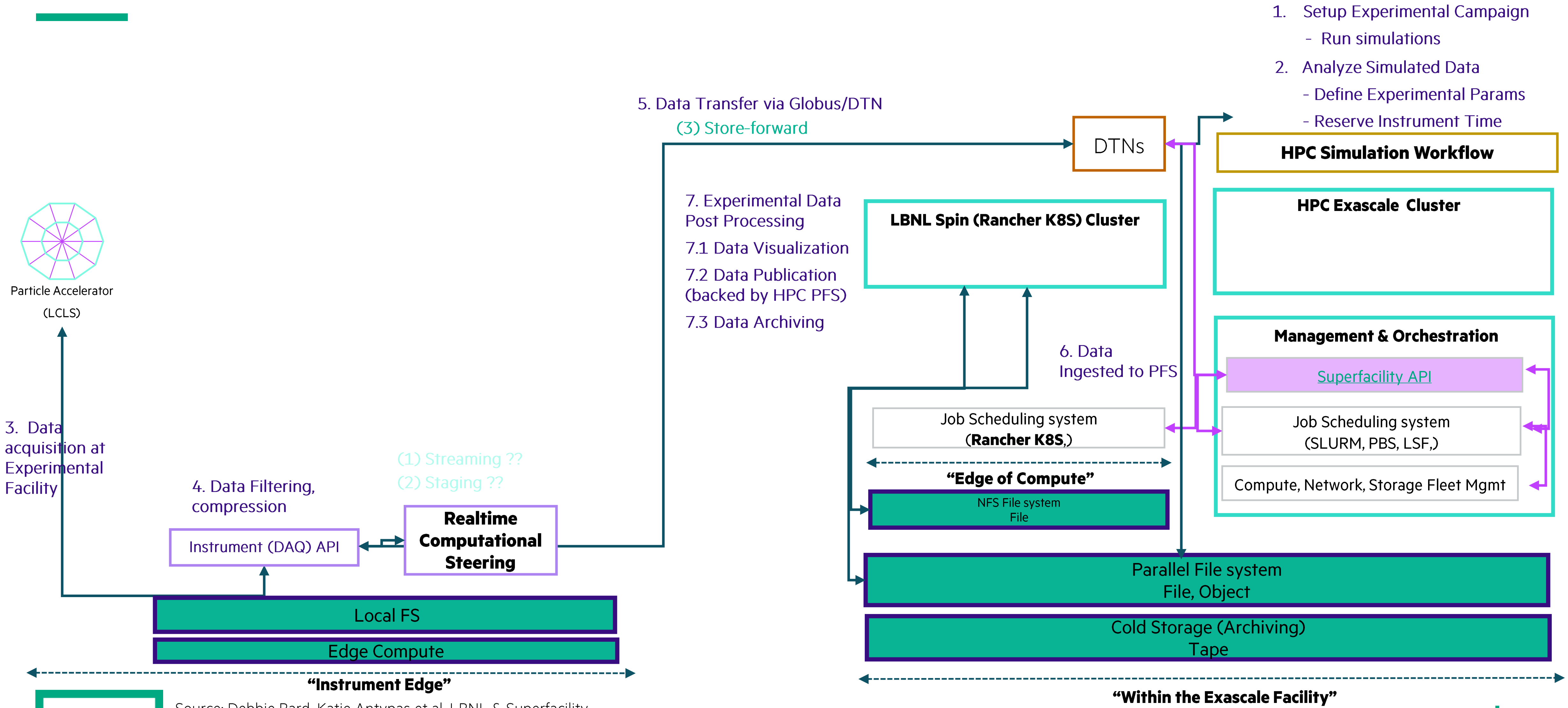
- Active Learning: start with baseline data set, incrementally improve model quality
- Provides guidance which input samples are of highest value for labeling and retraining
- Will be combined with weakly-supervised learning to account for labeling uncertainty
- Data Foundation benefits
  - maintains audit trail
  - enables model rewind
  - performs UQ calibration in the background
  - in the future, will update Learning Strategy based on performance of similar models and data
- Target: >75% reduction in data labeling effort
- Looking for future collaboration opportunities



REALIZING CMF IN END TO END HPC ENVIRONMENTS

END-TO-END DATA WORKFLOWS: SUPERFACILITY MODEL

Experiment Campaign & Data Analysis at “Edge of Compute” (Today)



Source: Debbie Bard, Katie Antypas et al. LBNL & Superfacility Publications

CMF PROTOTYPE ON NERSC SPIN

Making CMF easily accessible to Research Scientists & Instrumentation Engineers

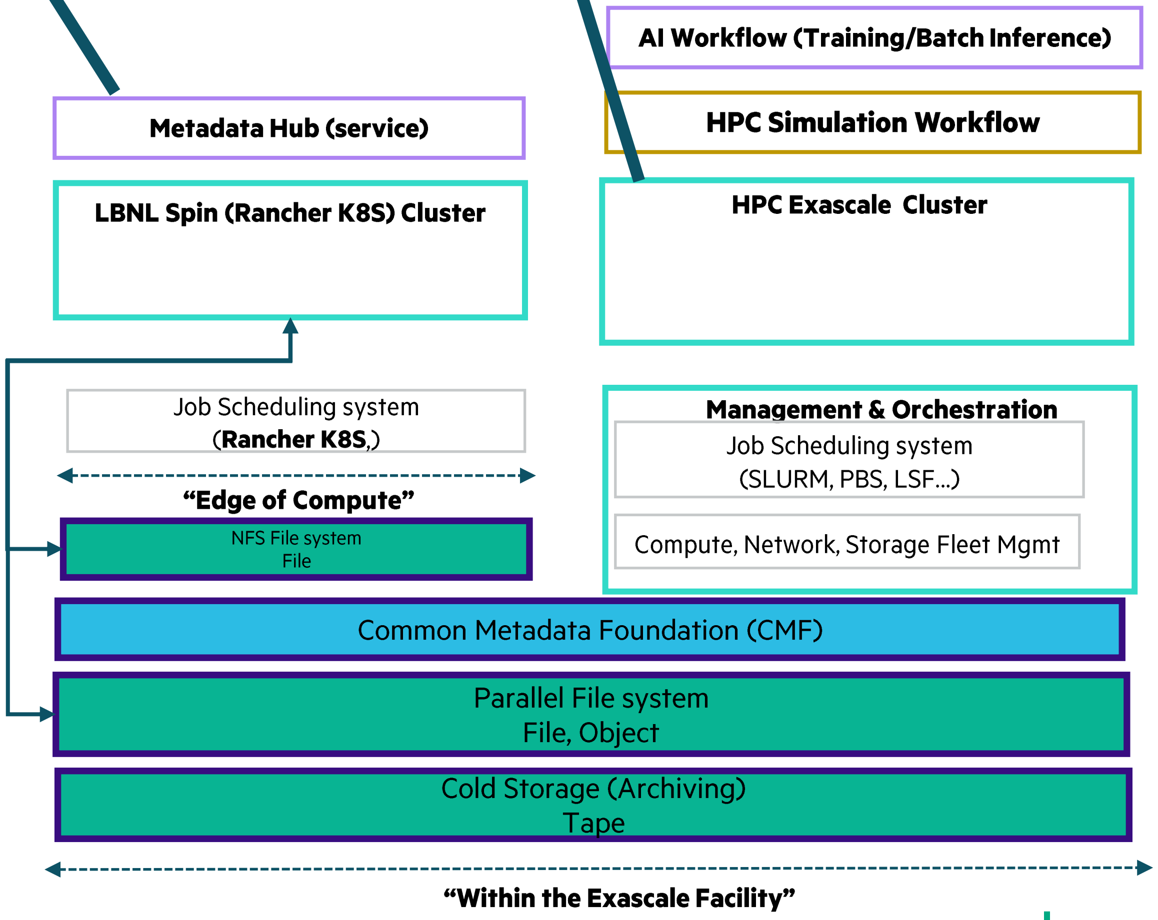
http://jupyterlab.cmf.development.svc.spin.nersc.org/

http://neo4j.cmf.development.svc.spin.nersc.org/browser/

pip install cmflib

- How to make CMF easily accessible to NERSC users?

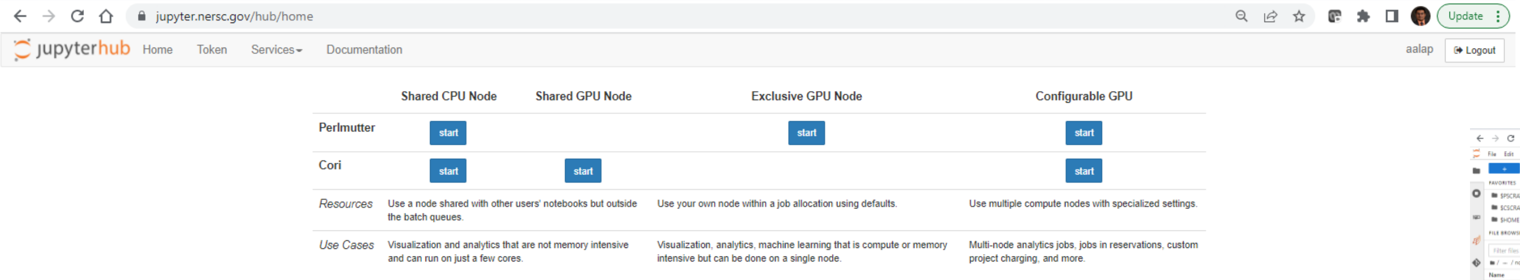
1. From Laptop to Exascale
   - pip install cmflib
   - *In Prototype today*
2. As a Facility-wide service
   - **Metadata Hub** On Spin
   - *In Experiment today*
   - Accessible across users, projects, teams (Future)
3. As a Facility-wide Service
   - With JupyterLab Extension
   - *In Design today*
   - Implicit access to CMF APIs from Notebooks (Future)
   - Linked with **Metadata Hub** & jupyter.nersc.gov

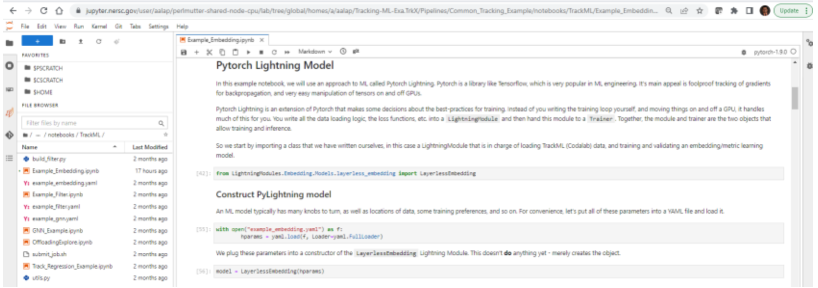


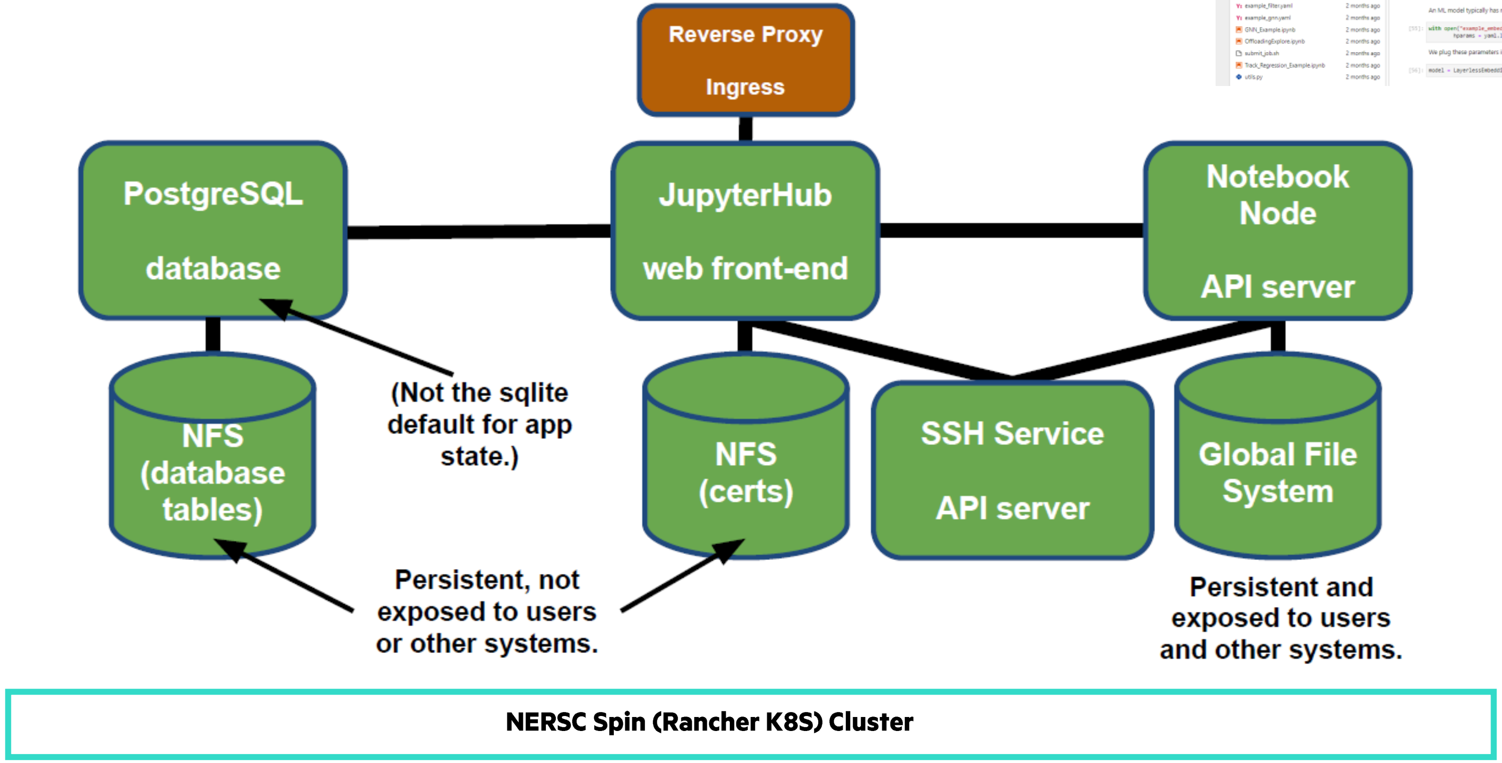
Acknowledgements: Suren Byna (LBNL Scientific Data Management), Johannes Blaschke (NERSC Data Science Engagement), Cory Snavely (NERSC Infrastructure Services)

JUPYTERLAB AT NERSC

Taking Inspiration from how JupyterLab is delivered to users at NERSC







Source: [NERSC Spin Training]

METADATA HUB AT NERSC?

How to deliver a Metadata Hub for users at NERSC (Future)