Provenance Graph Integration in Scientific Machine Learning Multi-workflows

Débora Pina[1], Lucas Moreira Dias[2], Maria Célia Santos Lopes[2], Daniel de Oliveira[3], Marta Mattoso[1]

[1]PESC/COPPE – Universidade Federal do Rio de Janeiro (UFRJ)

[2]LAMCE/COPPE – Universidade Federal do Rio de Janeiro (UFRJ)

[3]Institute of Computing - Universidade Federal Fluminense (UFF)

{dbpina,marta}@cos.ufrj.br, {ldias,celia}@lamce.coppe.ufrj.br,

danielcmo@ic.uff.br

Abstract. *Scientific Machine Learning (SciML) multi-workflows consist of multiple independent workflows, often implemented as fragmented Jupyter Notebooks, which hinder provenance tracking and limit end-to-end querying and reproducibility. In this paper, we present a real-world SciML multi-workflow that advances methods for capturing and integrating provenance in SciML applications. The proposed approach combines scientific domain modeling of multi-workflows as prospective provenance, provenance capture mechanisms, and shared execution identifiers to enable the construction of an end-to-end provenance graph. We demonstrate the approach through queries over the integrated provenance graph of a multi-workflow seismic binary data segmentation application, highlighting its potential for long-term provenance management.*

1. Introduction

Data Science workflows, particularly in scientific domains that rely on standard domain-specific data representations, *e.g.*, seismic image analysis [Islam and Wali 2024], are increasingly adopting Artificial Intelligence (AI)-based models. In the context of seismic image analysis, Machine Learning (ML) U-Net neural networks are used to learn features from seismic images [AlSalmi and Elsheikh 2023]. These U-Net models are then incorporated into multiple workflows to detect segmentation faults in seismic images, thus making the seismic image analysis a *Multi-workflow*. Establishing trust in the resulting models produced by these multiple, federated workflows is a top priority [Liu et al. 2022].

Provenance graphs [Freire et al. 2008] have been proposed to improve trust by tracking the lifecycle of workflows. In the context of seismic analysis multi-workflow, it includes tracking the training data preparation and prediction validation workflows [Procko et al. 2025, Souza et al. 2022]. By adopting a data model based on W3C PROV [Moreau et al. 2015] to represent the transformations involved in the workflows, *e.g.*, neural network generation, the resulting provenance graph can be recursively queried through standard relationships, *e.g.*, *used* and *wasGeneratedBy*. These graphs are essential to support explainability through end-to-end provenance queries [Missier et al. 2025].

Despite advances in workflow provenance and the adoption of PROV as a *de facto* standard, several open, problems remain in integrating provenance data across ML-based

multi-workflows [Souza et al. 2024, Ferreira da Silva et al. 2024]. The Workflows Community Summit [Ferreira da Silva et al. 2024] highlights the need for robust solutions and explicitly recommends *"establishing best practices and frameworks for provenance tracking in ML workflows, with emphasis on regulated environments and long-term data retention requirements"*. This recommendation reinforces the importance of scalable and interoperable, provenance management to support complex ML multi-workflow ecosystems.

We present a set of best practices for the integrated capture of provenance in ML multi-workflows executed in High Performance Computing (HPC) environments, as well as for exporting the integrated provenance graph for selected ML models. The adoption of PROV-N (PROV notation) supports long-term model trustworthiness, which is important when these models are deployed in production environments to support complex decision-making processes. To show these best practices, we present FaultSeg, a real-world Scientific Machine Learning (SciML) multi-workflow, for fault segmentation in seismic images.

Due to the use of domain-specific data formats and specialized data management requirements, the FaultSeg ML lifecycle, like most ML workflows, demands careful modeling of provenance data to represent data preparation steps and enable domain-oriented end-to-end queries. The different phases of the FaultSeg ML lifecycle are executed as independent workflows, each implemented as a different Jupyter Notebook. Finally, after selecting ML models, we export their integrated multi-workflow provenance graph to support querying, auditing, and long-term preservation of the model and its execution history. This paper is organized as follows: the FaultSeg workflow and the proposed best practices are presented in Section 2. Section 3 explores end-to-end queries on the integrated provenance graph. Related work is discussed in Section 4, and Section 5 concludes.

2. Best Practices in Multi-Workflow Systems: The FaultSeg Case

We propose four steps as best practices and demonstrate their application with the FaultSeg multi-workflow, as shown in Figure 1. The following subsections illustrate these steps.

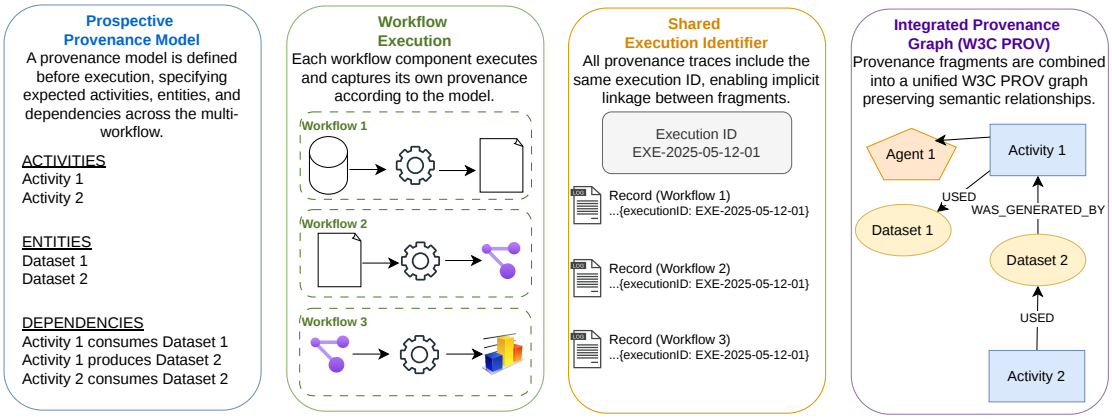

Figure 1. Provenance integration in multi-workflows.

Multi-workflows consist of a set of independently implemented workflows that can be connected, with data produced by one workflow is used as input to subsequent ones [Souza et al. 2022]. Each workflow is executed separately and requires human decisions before executing the next, which may run under a different ML environment, yet guided by

a multi-workflow leader. Instead of treating multi-workflow boundaries as isolation points, entities produced in one workflow should be integrated into subsequent ones. This preserves the continuity of provenance derivation across multi-workflow boundaries and enables the construction of an integrated provenance graph without post-hoc processing.

2.1. Multi-workflow Prospective Provenance Modeling

Prospective provenance (*p-prov*) [Freire et al. 2008] modeling is the initial step for provenance capture, as it defines an abstract representation of the execution process. In *p-prov*, users define, through the domain model (the conceptual model of a specific domain), the relevant PROV elements and relationships (*i.e.*, *entities*, *activities*, *used*), acting as a semantic contract that guides domain-aware provenance capture (and posterior analysis) in each independent workflow component. The conceptual model uses stereotypes that associate each PROV element with a domain-specific concept. These mappings are injected into the script of each workflow composing the multi-workflow. By explicitly embedding provenance semantics into the multi-workflow specification, this model guides provenance capture during execution. Persisting this representation also makes the main concepts explicit, preserving the original multi-workflow specification as provenance data.

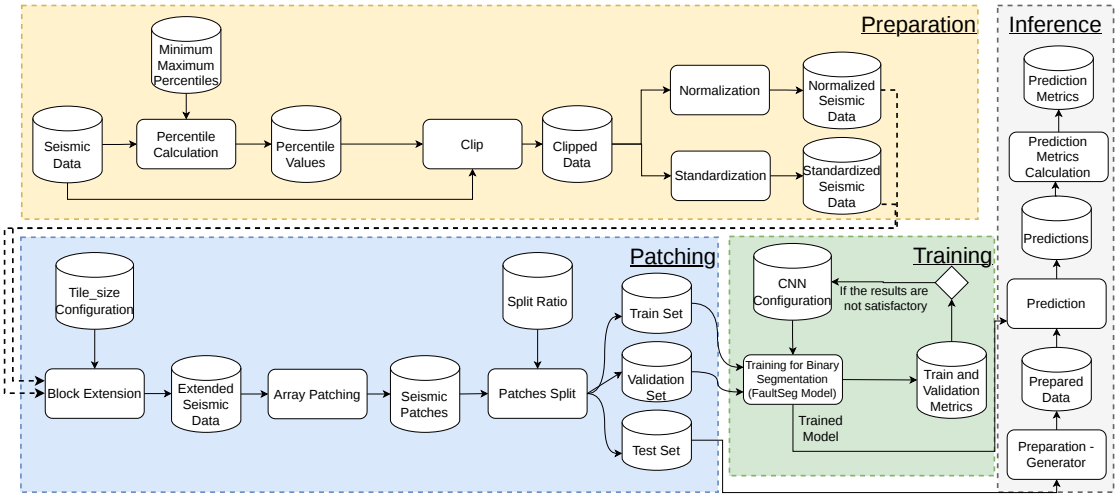

Figure 2. FaultSeg multi-workflow modeling with PROV entities and activities.

Figure 2 presents the specification of the FaultSeg multi-workflow with seismic domain classes associated with PROV elements, where rectangles denote activities, cylinders denote entities, and arrows represent relationships between them. The four FaultSeg workflows are highlighted by different regions: *(i)* Preparation (yellow), where the activities *Percentile Calculation* and *Clip* are applied to the seismic data, producing percentile values used in the *Clip* activity. The clipped data is then transformed through either *Normalization* or *Standardization*. *(ii)* Patching (blue), where seismic patches are created through the *Block Extension* and *Array Patching* activities based on the *Tile_size* configuration, and divided by the *Patches Split* activity into training, validation, and test sets. *(iii)* Training (green), where a U-Net model is built through *Training for Binary Segmentation* and evaluated using validation metrics. *(iv)* Inference (gray), where the data undergoes *Preparation-Generator*, followed by *Prediction*, and the results are evaluated by *Prediction Metrics Calculation*.

2.2. Integration of Provenance Graphs

Figure 1 presents how integration can be semantically achieved through a combination of *p-prov* modeling and shared execution identifiers, which are part of Retrospective Provenance (*r-prov*) [Freire et al. 2008], enabling the reconstruction of a unified provenance graph across multi-workflow executions. This identifier is generated at the beginning of the execution of the first workflow composing the multi-workflow, and subsequent workflows automatically retrieve it, ensuring that all executions are associated with the same multi-workflow, *i.e.*, *r-prov* is automatically integrated. The integrated provenance graph is materialized in a PROV-N representation, preserving relationships such as *used*, *wasGeneratedBy*, and *wasAssociatedWith*. PROV provides database connectors for ingesting provenance graphs into Neo4j (https://neo4j.com) and MongoDB (https://www.mongodb.com).

We used DLProv [Pina et al. 2024, Pina et al. 2025], a provenance-centric service that supports ML workflow analyses. DLProv captures multi-workflow provenance and follows the best practices to automatically deliver the integrated provenance graph of the multi-workflow execution. FaultSeg was executed on the Santos Dumont supercomputer (https://sdumont.lncc.br), in a *sequana_gpu* node using public real data from the F3 block of the North Sea (an offshore area used for oil & gas exploration). DLProv delivered the provenance graph of what happened after the four FaultSeg workflows executions.

Figure 3 presents a fragment of the FaultSeg PROV graph, corresponding to the execution of *Preparation* workflow and part of the *Patching* workflow (Figure 2). The PROV-N representation adopts yellow ellipses for entities, blue rectangles for activities, and white rectangles to show their attributes with execution values. Arrows trace back how entities were generated or used by activities linking the multi-workflow. Since provenance represents the history of the execution, the graph is traversed from the final result to the first input of the multi-workflow. The provenance graph in Figure 3 shows that the first activity, *percentilecalc*, *used* the input entity *seismic-entire-volume.npy* and generated *percentil* values. These values were then used by the *Clip* activity to generate a clipped dataset. This dataset was subsequently *used* by the *featurescaling* activity, where either normalization or standardization could be applied. This provenance shows that, in this *Preparation* workflow execution, normalization was chosen. As a result, the FaultSeg provenance graph supports comparative analyses in which model performance can be associated with variations in feature scaling. Next, Figure 3 shows the provenance of the *Patching* workflow execution (within the dotted region), using the scaled data generated by the previous workflow as if both workflows were part of the same execution. The *BlockExtension* activity then adjusts the data dimensions, which are *used* by the *ArrayPatching* activity to generate the seismic patches as part of the Patching workflow.

3. FaultSeg Queries with the Integrated Provenance Graph

The provenance of FaultSeg encoded in PROV-N can be considered a long-term document due to its compliance validated with PROV and its database connectors, which enable automatic transformation into independent representations. To query the FaultSeg multi-workflow execution, we convert its PROV-N graph to a queryable graph in Neo4j using a modified version of the PROV Database Connector (https://github.com/DLR-SC/prov-db-connector) to support more recent versions of Neo4j. Due to space limitations, we present only two Neo4j queries that traverse the data derivation paths of the FaultSeg multi-

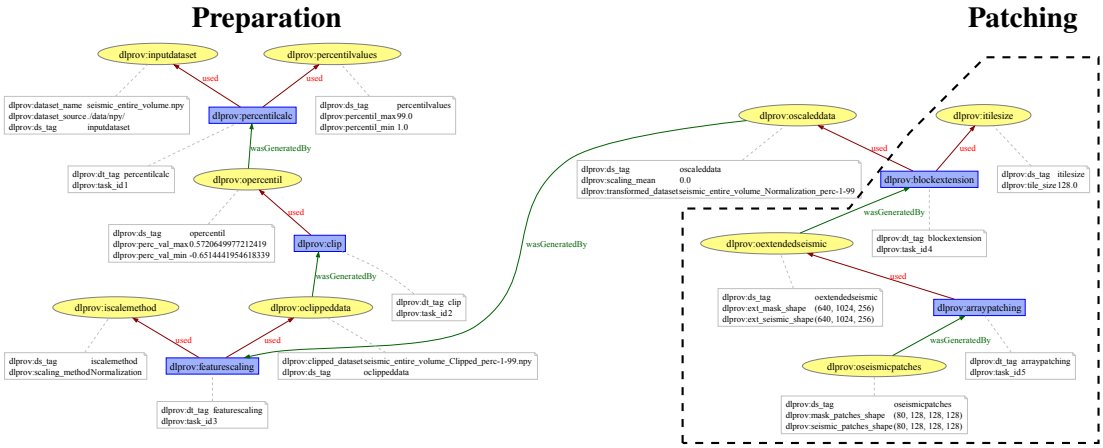

Figure 3. Fragment of the provenance graph represented in PROV-N.

workflow. The first query reconstructs the generation of the trained U-Net model, associating all involved activities and entities. The resulting graph (Figure 4) follows the Neo4j representation and provides an end-to-end view of the training process for the multi-workflow execution based on the outcomes of the query `MATCH path=(model:Entity ds_tag: 'otrainedmodel')-[:WAS_GENERATED_BY|USED*]->(n) RETURN path`. This resulting graph is traversed backwards, starting from the entity generated by the final *Train* activity, marked with a red circle, towards the input data entity, marked with a green circle. This traversal follows the chained *wasGeneratedBy* and *used* relationships, reflecting how data dependencies alternate between activities and entities across the multi-workflow.

The second query reconstructs the derivation of the intermediate entity *oscaleddata* by identifying the activities and entities that contributed to its generation. In particular, it determines which method was selected by the *FeatureScaling* activity, *i.e.*, standardized or normalized seismic data, by starting from the entity produced by *FeatureScaling* and traversing the graph backwards. The query is `MATCH path=(n)<-[:WAS_GENERATED_BY|USED*]-(scaled:Entity ds_tag: 'oscaleddata') RETURN path`. The result is depicted in Figure 4, highlighted by the dotted region. Due to space constraints, the same figure is reused to illustrate the outcomes of different queries. This result reveals the sequence of steps, such as clipping and percentile computation, enabling users to interpret the multi-workflow execution.

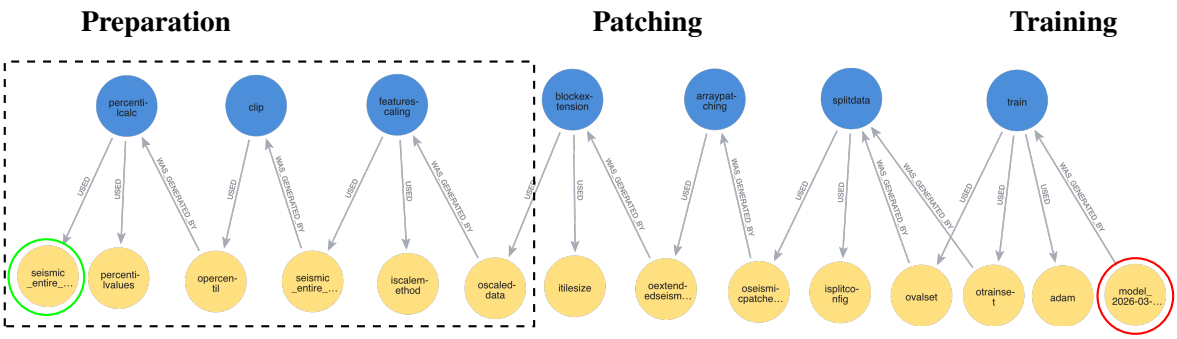

Figure 4. Results of the provenance queries over the integrated graph in Neo4j.

4. Related work

In previous work, we evaluated DLProv's querying capabilities against other approaches, showing its ability to answer queries that require traversing relationships with negligible overhead introduced by provenance capture [Pina et al. 2024, Pina et al. 2025]. However, extending this capability to long-term querying of multi-workflow provenance remained an open issue [Procko et al. 2025], which stems from two main limitations in current solutions.

The first limitation is with respect to not exporting the provenance graph to a representation that does not depend on having access to the provenance data of the ML environment. This limitation occurs even for solutions that provide single workflow provenance. For example, after the ML model is used in production, if the model users want to interpret the model generation, they have to resort to the ML framework where provenance was persisted by the ML team that produced the model. This is not long-term aware. ML frameworks such as MLflow (https://mlflow.org) and WandB (https://wandb.ai), as well as multi-workflow solutions such as PROV-ML [Souza et al. 2022] and SageMaker (https://aws.amazon.com/sagemaker/), require access to their provenance repositories for querying, which may be unavailable in the long term. MLflow2PROV [Schlegel and Sattler 2025] exports provenance graphs from MLflow enriched with versioning data. However, the entire ML multi-workflow must be executed under MLflow control, which may not be feasible.

The second limitation is the lack of native multi-workflow integration, which is important when data preparation and model training happen in isolated environments. Current solutions do not automatically bridge this gap, delivering independent provenance graphs for each workflow and leaving integration to be done manually by the ML team, or not at all. While MIDA [Souza et al. 2023] advances multi-workflow provenance integration using a PROV-compliant DBMS, exporting the integrated graph remains necessary for long-term retention. Our approach complements ML provenance systems by providing best practices for generating and exporting long-term multi-workflow provenance graphs that can be queried independently of the ML environment and its local provenance repository.

5. Conclusion and future work

In this paper, we highlight the importance of exporting an integrated provenance graph for long-term model preservation in ML multi-workflows. By enabling end-to-end querying through popular query systems, the proposed provenance graph generation is independent of the ML workflow environments in which the different workflows were executed. Our approach requires workflow modeling using PROV elements and manual instrumentation of the workflow script to embed both *p-prov* and *r-prov*, but recent advances in LLMs [Woyames et al. 2025, Gregori et al. 2025] and provenance support for agentic AI [Souza et al. 2025] can enable automatic provenance generation. Our best practices differ from *post-mortem* approaches by achieving integration by design through a shared semantic model and execution identifiers, allowing provenance generated by independent workflows to be directly integrated. This supports reproducibility and auditing in distributed and HPC-based scientific ML multi-workflow executions.

Acknowledgments

This work was partially funded by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior-Brasil (CAPES)-Finance Code 001, CNPq, FAPERJ, and Petrobras. Computer time on Santos Dumont machine at the National Scientific Computing Laboratory (LNCC). The authors declare that LLM-based tools were used exclusively for text revision.

References

AlSalmi, H. and Elsheikh, A. H. (2023). Automated seismic semantic segmentation using attention u-net. *Geophysics*, 89(1):WA247–WA263.

Ferreira da Silva, R. et al. (2024). Workflows community summit 2024: Future trends and challenges in scientific workflows. (ORNL/TM-2024/3573).

Freire, J., Koop, D., Santos, E., and Silva, C. T. (2008). Provenance for Computational Tasks: A Survey. *Computing in Science and Engineering*, 10:11–21.

Gregori, L. et al. (2025). An llm-guided platform for multi-granular collection and management of data provenance. *Journal of Big Data*, 12(1):187.

Islam, M. S. U. and Wali, A. (2024). A comprehensive review of deep learning techniques for salt dome segmentation in seismic images. *Journal of Applied Geophysics*, 230:105504.

Liu, H., Wang, Y., Fan, W., Liu, X., Li, Y., Jain, S., Liu, Y., Jain, A., and Tang, J. (2022). Trustworthy ai: A computational perspective. *ACM Trans. Intell. Syst. Technol.*, 14(1).

Missier, P. et al. (2025). From XAI to XEE: explainable end-to-end using influence and provenance. In *Symposium on Advanced Database Systems*, pages 560–569.

Moreau, L., Groth, P., Cheney, J., Lebo, T., and Miles, S. (2015). The rationale of PROV. *J. Web Semant.*, 35:235–257.

Pina, D., Chapman, A., Kunstmann, L., de Oliveira, D., and Mattoso, M. (2024). Dlprov: A data-centric support for deep learning workflow analyses. In *Proceedings of the Eighth Workshop on Data Management for End-to-End Machine Learning*, page 77–85. ACM.

Pina, D., Kunstmann, L., et al. (2025). Dlprov: a suite of provenance services for deep learning workflow analyses. *PeerJ Comp. Sci.*, 11:e2985.

Procko, T., Vonder Haar, L., and Ochoa, O. (2025). A survey of machine learning lifecycle provenance: Models, approaches and tools.

Schlegel, M. and Sattler, K.-U. (2025). Capturing end-to-end provenance for machine learning pipelines. *Information Systems*, 132:102495.

Souza, R. et al. (2022). Workflow provenance in the lifecycle of scientific machine learning. *Concurrency and Computation: Practice and Experience*, 34(14):e6544.

Souza, R. et al. (2023). Towards lightweight data integration using multi-workflow provenance and data observability. In *International Conference on e-Science*, pages 1–10.

Souza, R. et al. (2024). Workflow provenance in the computing continuum for responsible, trustworthy, and energy-efficient ai. In *International Conference on e-Science*, pages 1–7.

Souza, R. et al. (2025). Prov-agent: Unified provenance for tracking ai agent interactions in agentic workflows. In *International Conference on eScience*, pages 467–473.

Woyames, P. et al. (2025). Avaliação da capacidade de llms para especificar workflows. In *Anais do XIX Brazilian e-Science Workshop*, pages 81–88.