LF AI & DATA

MODEL OPENNESS FRAMEWORK

The Model Openness Framework (MOF) Specification

LF AI & Data - Generative AI Commons

Dec 17, 2024

Version 1.0

Status of this Document

This is a **stable document**. Its content has gone through extensive review within and outside LF AI & Data and is free of any known major issues. This document is a product of the LF AI & Data Generative AI Commons.

1. Introduction

This specification is based on The Model Openness Framework (MOF) whitepaper, which defines a ranked classification system that rates machine learning models based on their completeness and openness, following principles of open science, open source, open data, and open access. The MOF requires specific components of the model development lifecycle to be included and released under appropriate open licenses.

This specification provides a concise version of the MOF in the form of a set of requirements for model distributions to fulfill to be compliant. It focuses on the *what* rather than the *why*. For additional background and motivation for the different elements included in the MOF, please, refer to the whitepaper.

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC2119.

2. MOF Classes

The MOF defines three classes (Class III: Open Model, Class II: Open Tooling Model, and Class I: Open Science Model), each building upon the previous one, representing ascending levels of model completeness and openness.

Each class requires a set of components which, unless otherwise noted, MUST all be included and released under type-appropriate open licenses for the model to qualify for that class.

An ***open license*** is a license allowing unrestricted usage, study, modification, and redistribution for any purpose. ***A type-appropriate open license*** is an open license that is appropriate for the type of component it is covering: code, data, or documentation. An ***open source license*** is an open license appropriate for code. An ***open data license*** is an open license appropriate for data. An ***open content license*** is an open license appropriate for documentation. Recommendations of open licenses appropriate for each component type are given in Section 4.

MOF Class	Components Included	Usage
Class III – Open Model	1. Model Architecture 2. Final Model Parameters 3. Technical Report or Research Paper 4. Evaluation Results 5. Model Card 6. Data Card 7. Sample Model Outputs *(optional)*	• Unrestricted usage (access, use, modify, redistribute) • Create a product or service • Fine tune and align • Model optimizations
Class II – Open Tooling Model	1. All Class III Components 2. Training, Validation, and Testing Code 3. Inference Code 4. Evaluation Code 5. Evaluation Data 6. Supporting Libraries & Tools	• Understand training process • Validate benchmark claims • Inference optimizations
Class I – Open Science Model	1. All Class II and III Components 2. Research Paper 3. Datasets 4. Data Preprocessing Code 5. Intermediate Model Parameters 6. Model Metadata *(optional)*	• End to end analysis and auditing • Reproduction of a similar model • Data exploration and experimentation

3. MOF Components

The following defines the 17 components included in the above classes of model. They cover the degree of completeness and openness across all aspects of the development process of an ML model, including training data, model architecture, model parameters, evaluation benchmarks, and documentation.

Not all components are required for all classes. Each component section below specifies the classes and it applies to and the above table lists the components required for each class.

Not all components need to be distributed separately, some MAY be combined. For example, evaluation results MAY be included in the research paper, technical report, or model card rather than published as a standalone artifact. This sort of combination SHOULD however be limited to combining component types that are covered by the LICENSE file for that component or the whole distribution (see Section 5 LICENSE file).

The model distribution SHOULD include one additional component to be compliant with this specification: the MOF configuration file described in Section 6.

Model Architecture (III.1) - applies to all classes

The model architecture includes the ML algorithms, neural network layout, connectivity patterns, activation functions, and other architectural elements. Examples include transformers (e.g., GPT, BERT), convolutional neural networks (CNNs), recurrent neural networks (RNNs), and graph neural networks (GNNs).

The model architecture SHOULD be fully described in the research paper, technical report, or model card, and MUST be distributed as open source code under an OSI-approved open source software license.

Final Model Parameters (Checkpoints and Optimizer State) (III.2) - applies to all classes

Trained model parameters MUST be released under an open license. In the case of deep learning models, checkpoints from key intermediate stages of training as well as the final optimizer state SHOULD be included. At a minimum the final model parameters and optimizer state (when applicable) MUST be distributed, whether compressed or uncompressed, in a format compatible with popular deep learning frameworks such as TensorFlow, Keras, PyTorch or the framework independent ONNX file format.

Model parameters (weights and biases) are data for which an open source software license like Apache 2.0 and MIT are not well suited. Model parameters SHOULD be distributed under an open data license, like CDLA-Permissive-2.0.

The model architecture and the model parameters SHOULD be distributed separately because this separation allows each component to be studied, modified, redistributed, and used independently of the other.

Technical Report (III.3) - applies to all classes

The technical report, which MAY be in the form of a white paper, provides the necessary documentation for the model consumer to understand the performance, usage, and implications of using the model, but not necessarily enough details to reproduce the model and replicate its results.

The technical report MUST be included in the distribution or made available on a permanent open access platform such as arXiv, and SHOULD be linked from the model card. The technical report MAY be omitted if a research paper is provided.

The technical report MUST be distributed under an open license that SHOULD be appropriate for documentation, ideally CC-BY-4.0.

Evaluation Results (III.4) - applies to all classes

Detailed quantitative metrics and qualitative results from evaluating the model MUST be reported. They MAY be included in the technical report, the research paper, or the model card.

Tests can evaluate any factor, not limited to model efficiency, accuracy, performance, fairness and bias evaluations, toxicity, truthfulness and so forth.

Producers MUST include benchmark test results, whether industry standard benchmarks or custom benchmark tests that were developed. If industry standard benchmark tests or test suites are used, the test suite name, test name and version number MUST be included with the results. If custom benchmarks were developed, whether in code or any form of media including text, images, the custom benchmarks MUST be included in full for validation.

The raw outputs of the model evaluation MUST be distributed under an open license that SHOULD be appropriate for content like CC-BY-4.0.

Model Card (III.5) - applies to all classes

A model card provides metrics, usage guidance, and details about a model. Model cards SHOULD cover model details, intended uses, factors, evaluation, risks, and mitigations related to the model (see Model cards for model reporting).

The model card itself MUST be distributed under an open license that SHOULD be appropriate for documentation, ideally CC-BY-4.0.

Data Card (III.6) - applies to all classes

A data card provides summary statistics and other details about a dataset. Data cards SHOULD describe metrics about the features, instances, intended uses, motivation, and collection process.

The data card MAY be combined with the model card.

The data card MUST be distributed under an open license that SHOULD be appropriate for documentation, ideally CC-BY-4.0.

Sample Model Outputs (III.7) - applies to all classes

Sample outputs are text samples, images, videos, software code, audio, 3D assets, metadata or any other potential output generated from the model, including predictions and probabilities.

Sample outputs are RECOMMENDED but not required. They MAY be omitted. If they are included in the distribution, they MUST be shared publicly without copyright or restrictions where legally permitted so that they can be redistributed with the release.

For certain sensitive domains, generated examples MAY be anonymized or simulated if needed. In the event where model outputs are not copyrightable, the outputs SHOULD be released without a license, and this SHOULD be noted in the LICENSE file.

Note that actual model outputs that are generated once the model consumer performs inference are not considered in the MOF.

Training, Validation, and Testing Code (II.2) - applies to classes II and I

The full code for training, validating and testing the model, including model construction, training loop, hyperparameter selection, and checkpointing SHOULD be distributed as open source software. Any fine-tuning code, reinforcement learning code or any other method that otherwise modifies that model parameters or code that implements adapters that ultimately affect the performance of the model MUST be included.

Comments explaining the approach SHOULD be included in the code, ideally following PEP 8 style guide for Python code. It is also RECOMMENDED to include with the release the log files generated during training.

The training, validation, and testing code itself MUST be released under an OSI-approved open source software license while any logs SHOULD be covered by an open content license like CC-BY-4.0.

Inference Code (II.3) - applies to classes II and I

Code for performing inference includes any data preprocessing or postprocessing required during inference and possibly any model optimizations and dependencies like external libraries. It MUST include any code required to fully replicate the benchmark results for the model.

The inference code MUST be released under an OSI-approved open source software license.

Evaluation Code (II.4) - applies to classes II and I

Any code used for model evaluation and benchmarking MUST be included and distributed under an OSI-approved open source software license.

Evaluation Data (II.5) - applies to classes II and I

When the model is evaluated with data (be it any media format including text, images, videos, audio, 3D data, and so forth), that evaluation data MUST be included with the distribution.

Where the model producer relies on standard benchmark tests that are widely disseminated, they MAY be omitted from the distribution, but they MUST be described in the technical report, research paper, or model card, along with the version of the test.

The evaluation data MUST be released under a data or content appropriate open license like CDLA-Permissive-2.0 or CC-BY-4.0.

Supporting Libraries and Tools (II.6) - applies to classes II and I

Any supporting code libraries, utilities, or tools developed in the course of the development of the model SHOULD be distributed under an OSI-approved open source software license. This includes data loaders, visualization code, simulation environments, etc. Use of existing and custom open source tools SHOULD also be documented.

Any of the following tools and libraries SHOULD also be included:

- Software libraries and frameworks used in model development along with version details.
- Tokenizers – Any code used to tokenize text and any data used to train the tokenizer.
- Hyperparameter search code – Any code used for automating hyperparameter tuning.
- Compute infrastructure code – Any setup code for specialized compute infrastructure used to scale training.
- Monitoring code – Code for tracking experiments, metrics, artifacts etc. used during model development.
- Containerization files – Dockerfiles or other container packaging to distribute the model.
- Frontend/visualization – Any web/mobile frontends or visualizations built on top of the model outputs.
- Deployment orchestration – Infrastructure-as-Code templates for deploying the model to production.
- Model integration code – Wrapper code/SDKs to integrate the model into downstream applications.
- Interactive demos – Links to hosted interactive demos of the model through Jupyter, Streamlit, etc.

Presumably most libraries and tools used already have their own licenses, but if the model producer created their own libraries or tools they MUST include them with the distribution under an OSI-approved open source software license.

Research Paper (I.2) - applies to class I

The research paper detailing the model methodology, results, and analysis MUST be included in the distribution or made available on a permanent open access platform such as arXiv, and SHOULD be linked from the model card.

The research paper MUST be released under an open license appropriate for documentation, ideally CC-BY-4.0.

The following structure is RECOMMENDED but not required: abstract, introduction, related work, methods, results, discussion, conclusion, references.

The research paper MAY be replaced with a detailed technical report providing the same information and distributed under an open license appropriate for documentation, ideally CC-BY-4.0.

Datasets (I.3) - applies to class I

Datasets include training data which is data used for any form of model training including pre-training, fine-tuning, alignment using reinforcement learning techniques or data used for other methods that otherwise modify the weights of the model. Datasets also include data used for model validation and testing as well as data used with benchmark tests. The datasets component also includes tokenized datasets when present.

Data can be any form or combination of media, whether text, code, images, videos, audio, 3D objects, URIs and any other data used for training, validation and testing purposes.

Datasets also include any metadata. This includes anything from annotation data like labels, bounding boxes and key points to attribution, bitrates, resolution and other metadata relevant to a dataset used in the model development process.

The datasets used to develop the model MUST be provided, in the public domain, as copyrighted data, or under any form of license. They SHOULD be released under an open license, preferably CC-BY-4.0 or CC-0.

Any limits on sharing due to privacy or sensitivity SHOULD be documented. Both pre- and post-processed data SHOULD be supplied, however producers MAY provide instead links to any curated raw datasets online if they are accompanied by data preprocessing code.

Data Preprocessing Code (I.4) - applies to class I

The data preprocessing code is all code used for preprocessing, cleaning, and formatting the training, validation, and testing data for a model. It also includes code used to transform fine-tuning data and code that is used for alignment tasks like Reinforcement Learning from Human Feedback (RLHF). Other data preprocessing code such as code for data ingestion when appropriate, feature engineering, data augmentation and tokenization is also included.

The data preprocessing code MUST be released using an OSI-approved open source software license.

Intermediate Model Parameters (Checkpoints and Optimizer States, log files) (I.5) - applies to class I

In addition to the final checkpoints and optimizer states, for Class I models, the checkpoints and optimizer states (when applicable) from key intermediate stages of training along with the log files MUST be included and distributed under an open license.

Intermediate model parameters SHOULD be distributed under an open data license, such as CDLA-Permissive-2.0.

Model Metadata (I.6) - applies to class I

There are other forms of metadata that can provide additional context about the model, such as the version of the framework used to create it and custom tags or descriptions provided by the developer including model and data lineage information. There is no particular requirement or profile for this type of metadata and it can reveal anything the developer would like to include with the shipped model. This information can help with model management, especially when working with multiple versions of models or conducting experiments. Often the metadata is exported from or loaded by a metadata store.

The model metadata MAY be included in the model card, research paper, or technical report.

Any model metadata SHOULD be covered by an open data license such as CDLA-Permissive-2.0.

Model Openness Configuration File - applies to all classes

The MOF configuration file MUST be included in any distribution. It describes what model components are included in the release and what license covers each component.

The file itself MUST be distributed under an open license and SHOULD be distributed under the CC-BY-4.0 license.

See Section 6 for further details.

4. MOF Recommended Licenses

Each component is of one of three possible types: data, code, or documentation. The table below specifies standard open licenses that SHOULD be used for each component - with some flexibility to allow equivalent licenses.

Component	Type	Recommended Open License
Datasets	Data	Preferred: CDLA-Permissive-2.0, CC-BY-4.0 Acceptable: Any including unlicensed
Data Preprocessing Code	Code	Acceptable: OSI-approved
Model Architecture	Code	Acceptable: OSI-approved
Model Parameters	Data	Preferred: CDLA-Permissive-2.0 Acceptable: OSI-Approved, Open Data Licenses
Model Metadata	Data	Preferred: CDLA-Permissive-2.0 Acceptable: CC-BY-4.0, Open Data Licenses
Training Code	Code	Acceptable: OSI-approved
Inference Code	Code	Acceptable: OSI-approved
Evaluation Code	Code	Acceptable: OSI-approved
Evaluation Data	Data	Preferred: CDLA-Permissive-2.0 Acceptable: CC-BY-4.0, Open Data Licenses
Evaluation Results	Documentation	Preferred: CC-BY-4.0 Acceptable: Open Content Licenses
Supporting libraries and Tools	Code	Acceptable: OSI-approved
Model Card	Documentation	Preferred: CC-BY-4.0 Acceptable: Open Content Licenses
Data Card	Documentation	Preferred: CC-BY-4.0 Acceptable: Open Content Licenses

Component	Type	Recommended Open License
Technical Report	Documentation	Preferred: CC-BY-4.0 Acceptable: Open Content Licenses
Research Paper	Documentation	Preferred: CC-BY-4.0 Acceptable: Open Content Licenses
Sample Model Outputs	Data or Code	Unlicensed

5. LICENSE file

All distributions MUST include a LICENSE file that describes the licenses used for the model. It is RECOMMENDED that the LICENSE file includes all licenses that apply to the model. For instance if software is covered under Apache 2.0 and all documentation and data under CC-BY-4.0, then the text of both licenses SHOULD be included in the LICENSE file in their entirety including the license heading in order to distinguish what text belongs to which license.

Alternatively, a distribution MAY contain different LICENSE files that are bound to the different components included in the distribution. The LICENSE file for each component SHOULD be located in the base directory of the component that it covers. When different types of components are combined the LICENSE file SHOULD include all the licenses that apply to the different types of components being combined.

The MOF Configuration file contains the path to the appropriate LICENSE file for each component included in the distribution and supports both the per component LICENSE method and the single LICENSE file method.

6. Configuration file

The MOF configuration file is a file in a machine readable format such as JSON, YAML, or an AIBOM format, which contains the following information:

- **Framework:** This object contains the details related to the framework itself, including the following required attributes:
  - **name:** name of the framework.
    The attribute type is string.
  - **version:** version number of the framework.
    The attribute type is string.
  - **date:** date the framework version was published.
    The attribute type is string in the format "YYYY-MM-DD".
- **Release:** This object contains the details of the model being released. There are a number of attributes:
  - **name:** the name of the release. The attribute type is string.
  - **version:** the version for the release, which can be the parameter count or some other identifier that distinguishes the model from both previous versions and versions of the same model with different parameter counts. The attribute type is string.
  - **date:** the date of the release. The attribute type is string in format "YYYY-MM-DD".
  - **type:** the nature of the model, i.e., language model, image generation, audio generation, image classification, statistical ML, or any number of other types of models. The attribute type is string.
  - **architecture:** the model architecture employed, i.e., transformer, diffusion, GAN, NERF, VGG, Resnet, K-means, or any other type of model architecture. The attribute type is string.

- **treatment:** any type of post-training treatment, like fine-tuning, constitutional alignment, RLHF or any other treatment that otherwise modifies the parameters of the original model. If no treatment has been applied then this attribute is an empty string. The attribute type is string.
- **origin:** the original model, generally this is the foundation model. If this is not a foundation model in the release, then this attribute contains the name and version of the model that was modified. The attribute type is string or left empty for foundation or non-derivative models.
- **producer:** the name model producer or publisher, could be a company, organization, group or individual. The attribute type is string.
- **contact:** an email address for the model producer or publisher. The type is string, the format is an email address.
- **mof_class:** the qualifying Model Openness Framework class of the release as generated by the Model Openness Checker. The attribute type is integer.

- **Components:** This object contains a list of components that are included with the model distribution, as well as each component's details:
  - **description:** A text description of the component, using the default values is acceptable. When introducing a new component that is outside of the standard components it is important to include a description of the component.
  - **location:** the location of the component within the distribution, full path is required in UNIX format with leading slash for the root directory. The attribute type is string.
  - **license:** the SPDX identifier of the license or licenses used for the component. In the case where multiple licenses are used for a single component, which is often the case for libraries and tools, they must be provided in a comma-separated list. The license field must use a valid SPDX license identifier, found here: https://spdx.org/licenses/. The attribute type is string.
  - **license_path:** the location of the LICENSE file for the component within the distribution, full path is required in POSIX format with leading slash for the root directory. More than one component can point to the same LICENSE file. In the event the component employs multiple licenses, the LICENSE file should contain the text for all the licenses used. Alternatively, multiple license files may be specified, each separated by a comma. However they must correspond in order to the comma separated list of license names provided in the license attribute. The attribute type is string.

The Model Openness Tool (MOT) can be used to easily generate the model configuration file.

7. Out of Scope

The MOF does not intend to address any of the following: AI safety (including bias, fairness, and trustworthiness), performance testing, red-teaming, security and privacy, components related to model serving, and model provenance.

References

The Model Openness Framework (MOF) whitepaper: https://arxiv.org/abs/2403.13784

The Model Openness Tool (MOT): https://mot.isitopen.ai

Model cards for model reporting: https://doi.org/10.1145/3287560.3287596

OSI-approved licenses: https://opensource.org/licenses

Apache 2.0: https://www.apache.org/licenses/LICENSE-2.0.html

MIT: https://opensource.org/license/MIT

CC-BY-4.0: https://creativecommons.org/licenses/by/4.0/

CDLA-Permissive-2.0: https://cdla.dev/permissive-2-0/

Acknowledgements

Beside the authors of the MOF whitepaper - Matt White, Ibrahim Haddad, Cailean Osborne, Xiao-Yang (Yanglet) Liu, Ahmed Abdelmonsef, Sachin Varghese, Arnaud Le Hors - from which most of the content in this document comes from, the following individuals deserve extra credit for their contribution to this specification: Arnaud Le Hors (editor) and Cailean Osborne.

License

A. Deviations from the MOF paper

This specification is intended to be largely aligned with the requirements set by the MOF paper v6 from October 18th 2024 but it deviates from it in a few areas. This section highlights those deviations.

- Section 3: The specification recommends having a configuration file but doesn't require it (i.e., it's a SHOULD instead of MUST).
- Section 3 - Model card: The specification allows for the data card to be combined with the model card.
- Section 5 - LICENSE File: The specification covers the case for a per component file where different types are combined.
- Section 6: The specification requires the configuration file to be in a machine readable format but allows for the use of other formats than JSON.

B. About The Generative AI Commons

The Generative AI Commons is a community-driven initiative at the Linux Foundation's AI & Data Foundation. It is a vendor neutral forum and open participation initiative focused on advancing principles of open science and open source in generative AI. The Generative AI Commons is dedicated to fostering the democratization, advancement and adoption of efficient, secure, reliable, and ethical Generative AI open source innovations through neutral governance, open and transparent collaboration and education.

More about the Generative AI Commons, as well as details and links to join the community can be found at https://genaicommons.org.

C. About The LF AI & Data Foundation

The LF AI & Data Foundation is a global not-for-profit foundation that hosts critical components of the global AI & data technology infrastructure at the Linux Foundation.

It brings together the world's top developers, end users, and vendors to identify and contribute to the projects and initiatives that address industry challenges for the benefit of all participants.

More about the LF AI & Data Foundation can be found at https://lfaidata.foundation/