AN ANALYSIS OF THE MACHINE LEARNING SUPPLY CHAIN ON HUGGING FACE

Trevor Stalnaker

William and Mary

Abstract

The last decade has seen widespread adoption of Machine Learning (ML) components in software systems. This has occurred in nearly every domain, from natural language processing to computer vision. These ML components range from relatively simple neural networks to complex and resource-intensive large language models. However, despite this widespread adoption, little is known about the supply chain relationships that produce these models, which can have implications for compliance and security. In this work*, we conduct an extensive analysis of 760,460 models and 175,000 datasets mined from the popular model-sharing site Hugging Face. First, we evaluate the current state of documentation in the Hugging Face supply chain, report real-world examples of shortcomings, and offer actionable suggestions for improvement. Next, we analyze the underlying structure of the extant supply chain. Finally, we explore the current licensing landscape against what was reported in prior work and discuss the unique challenges posed in this domain. Our results motivate multiple research avenues, including the need for better license management for ML models/datasets, better support for model documentation, and automated inconsistency checking and validation. We make our research infrastructure and dataset available to facilitate future research.

Introduction

The use of machine learning (ML) models in software applications has increased dramatically over the last decade, including in recommendation systems, computer vision, chatbots, image generation, automated software engineering, and more. As creating and training new models has become increasingly more expensive[28], developers have turned to fine-tuning pre-existing models. This approach may save time and effort, but it also introduces the complexity of a new ML supply chain, complete with novel challenges. While supply chains for conventional software typically consist of software components, libraries, configuration files, and processes[11,31], for ML-intensive systems, the supply chain is even more complex. Creating a modern ML system involves the integration of ML models and traditional software components. Additionally, one must account for dependencies between models and between models and their training datasets. Further, the training of ML models can rely on multiple datasets or dataset aggregates, and datasets, in turn, each have their own supply chain[20].

Adaptation or reuse of existing models can also introduce degrees of complexity. Models can be fine-tuned[36] or quantized[12] from existing models. Different ML models can even be combined to form a single architecture that can, in turn, be reused or adapted[1]. Finally, the outputs from one model can be used to train another model, such as through synthetic data[21] or the use of a teacher-student approach[15].

ML supply chains, particularly those involving generative models[20], are not necessarily linear, progressing cleanly from one dependency or stage to the next. Instead, as outlined by Lee et al.[20], there can be branches and even cycles in the supply chain. For example, datasets can be used not only in the initial training of models but also for the fine-tuning of pre-existing base models, sometimes in multiple instances and by different developers. Additionally, some stages of the supply chain can back feed into others, making relationships increasingly complex, if not recursive. For example, the outputs of a generative model can be added to pre-existing datasets and used for training future versions. The sheer volume of content from generative models makes this situation increasingly likely, if not inevitable.

Understanding all these dependency relationships is critical not only for license compliance, which requires a full understanding of the components used in a project and their associated licenses, but also for detecting, mitigating, and managing security threats involved with model reuse, such as weight poisoning attacks[19], data poisoning[13], and even malware hidden in model weights and assembled at runtime[34].

* The content of this report is supported by research that is available as a preprint on arxiv[39].

While previous work has considered the documentation[29], evolution[17], and environmental impact[4] of ML models, the complexity and challenges of the ML supply chain as a whole have not yet been fully explored. The supply chain relationships between models and dependent GitHub repositories have been explored by Pepe et al.[29] and Jiang et al.[18], but the relationships between ML models remain understudied.

To contribute to bridging this gap, we aim to investigate the emerging ML model supply chain on Hugging Face (HF) (hereafter referred to as the 'ML supply chain'), specifically the documentation practices of model owners, the structure and complexity of the supply chain itself, and the existence and prevalence of potential compliance issues. We note that proper documentation and documentation practices are foundational to mapping and thus managing the ML supply chain. Launched in 2016, HF is, as of January 2025, the largest repository of ML models and datasets. HF provides a central hub for model developers and data scientists to explore, share, and experiment with ML models. While other model forges—such as TensorFlow Hub and PyTorch Hub—exist, the HF platform has the widest reach, hosting more than 750K models and 175K datasets across many different use cases. HF therefore represents a robust opportunity to understand ML supply chains and analyze how developers interact with them.

Background

The ML supply chain has emerged as a complex structure with many steps and components, each of which can affect the final output of a given model. Lee et al. identify eight stages in the generative AI supply chain, but most apply more broadly: 1) the creation of expressive works that will eventually be used to train a model, 2) the conversion of these expressive works into digital data, 3) the compilation of these data points into training datasets, 4) the creation of an ML model by selecting an architecture, training datasets, and a training algorithm, 5) the fine-tuning of existing "base" models, 6) the deployment of the trained/fine-tuned model as a service, 7) the use of the model to generate output, and finally 8) applying additional alignment to further improve the model or meet user needs[20]. Modern ML models are also becoming increasingly complex with respect to their architectures. For example, Mixture of Expert (MoE) models and ensemble models comprise multiple specialized models trained by dividing the problem space into homogeneous regions[22,32].

Sharing information about models has become critically important. ML components do not, however, always disclose data sources, making it difficult to comply with or even keep track of licensing obligations associated with the data[14]. This creates conditions that could lead to legal disputes over the use of copyrighted material in training ML components[33]. Model cards[26] have become the standard method for sharing information about ML models hosted on HF, including intended uses, limitations, and datasets used in training. Although tools are in development to streamline the process[10], model card creation on HF is primarily a manual process in which a user enters documentation about a model into a predefined markdown template[9]. This template is robust, including spaces for model description, uses, bias, limitations, testing, etc., as well as optional fields for citation, technical specifications, and environmental impact. The manual nature of documentation, however, introduces significant potential for human error, ambiguity, and incompleteness[16,18]. Indeed, the quality and adoption of model cards remains low[3], despite user studies and other efforts by HF[7] to ameliorate the situation, and despite researchers proposing tools such as DocML[3] to create and evolve model cards.

While this study has been informed by prior work, our investigation focuses on the structure and characteristics of the ML supply chain itself and the challenges encountered in managing it. While previous work examined deficiencies in documentation and licensing at the level of individual models, we investigate the ML supply chain at a higher level by examining how it is structured and how models relate to each other.

Study Design

This study's goal was to evaluate the state of documentation, structure, and licensing of the ML supply chain. We aim to address the following research questions:

RQ0: What documentation deficiencies exist on HF that potentially complicate mapping the ML supply chain?
RQ1: What is the structure of the ML supply chain?
RQ2: What is the licensing landscape for models and datasets and what are potential compliance challenges?

Next, we describe our methodology depicted in Fig. 1.

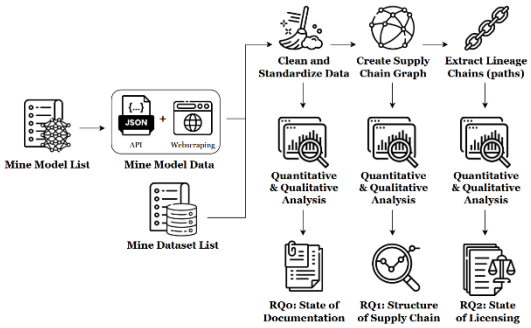

Figure 1: Methodology Overview

Data Collection

We used the HF API to extract a list of all models publicly hosted on the platform. An initial list of 500,000 models was pulled on July 9, 2024, and a full list of all available models was obtained on July 11, 2024. Using the same APIs, we mined data (model cards and metadata) for each model on this list. For models that were unavailable through the API, we employed a web-scraping technique relying on a combination of Python's request module and the BeautifulSoup HTML parser. Mining took place between July 9 and July 12, 2024, yielding 760,460 mined models. We also used the HF API to gather the list of all datasets publicly hosted on the platform and available through the API as of July 9, 2024. We make all the raw data acquired from mining available in our replication package[37].

Data Normalization and Cleaning

Once the data for models and datasets was downloaded, it needed to be cleaned, standardized, and stored in a more usable form. This process involved standardizing the names of declared base models and datasets and mapping to unique identifiers where possible. For example, xlm-roberta-base is a shorthand for FacebookAI/xlm-roberta-base, which has the unique internal ID 621ffdc036468d709f174364. We relied on HF's internal mechanisms to resolve these differences where possible. For example, attempting to load the model page for xlm-roberta-base on HF redirects to the page for FacebookAI/xlm-roberta-base. Additionally, we resolved inconsistencies in field inputs. For example, [], "", and None were all being used to denote no declared licenses. We standardized these variations to the empty list to make later computation simpler. We further elaborate on documentation challenges that required special attention and resolution in our discussion of RQ0.

Extracting Licensing Information

We extracted licensing information from the tags present in model metadata. While licenses can also be declared in the CardData attribute of the model metadata, we found through an analysis of our dataset that the tags were more complete in every instance than the CardData attribute. In nearly all instances, licensing information was in both locations (99.9%). However, for 0.1% of models, the licensing information was exclusively in the tags, and there were no instances that relied only on the CardData attribute. We also looked for discrepancies in those cases where both locations were utilized. We identified only 134 such discrepancies, and in all cases, an additional license was declared in the tags that was not declared in the CardData. There were no irreconcilable differences (i.e., MIT vs GPL). From this, we concluded that the model tags are the most complete machine-readable source for licensing information.

Data Analysis

To address RQ0, we quantitatively and qualitatively analyzed and discussed the different types of documentation issues that we observed in our attempt to understand the ML supply chain on HF.

Once the dataset had been cleaned, to address RQ1, we used the networkx Python library to create a directed graph representing the observable ML supply chain. Each node in the graph represents a distinct model, and an edge represents a dependency relationship. For example, if model A listed model B as a base model, there would be an edge from B to A. We would describe model B as the parent of model A and model A as the child of model B. Due to the inherent issues and incompleteness of model documentation, this graph, by its nature, only represents a subset of the total ML supply chain on HF.

For the sake of analysis, we flatten the individual model supply chains by extracting all longest paths (e.g., lineage chains) from the graph that go from source nodes to sinks. Each of these paths goes from a root base model to a model at the end of the ML supply chain (i.e., a leaf node with no dependents or outgoing connections).

Using the paths resulting from RQ1, we addressed RQ2 by analyzing the state of licensing in the ML supply chain, developing scripts and heuristics to look for parent/child license differences, and potential license compliance problems related to model/dataset dependencies. We define a parent/child license difference to refer to the difference between the license(s) of a parent and

a child model/dataset. Assume, for example, that model B is the child of model A. If model B was licensed under MIT and model A under Apache-2.0, we would say there is a parent/child license difference between A and B. We avoid stronger terms such as "inconsistencies" and "incompatibilities", since some of these differences may in fact be consistent and compliant with licensing terms. Two authors reviewed all distinct licenses in our dataset and grouped them into six categories, which can provide the basis for future research into license compatibility and licensing trends in the ML supply chain.

All scripts used for mining, data clean-up, and analysis can be found in our replication package[37] and can be used to foster further research in this area.

Study Results

RQ1: Documentation Challenges on Hugging Face

Inability to Access Meta Data

Not all model metadata was accessible through the HF API. There were 7,258 model cards (0.95%) that had to be scraped using a combination of Python's requests and BeautifulSoup libraries. The most frequent challenge (i.e., found in 99.4% of cases) resulted from gating based on the acceptance of terms and conditions. This was particularly the case for models owned by or relying on models created by larger entities such as Google or Meta. Twenty-two models and their metadata were gated behind age restrictions imposed by the model owner, requiring a HF account to access. Accessing the full repository, and thus metadata, required logging in with a HF account and then, in most cases, providing an email address. This hurdle to even access the metadata for a model could potentially complicate the creation of fully automated analyzers and ML/AI-BOM (Machine Learning/Artificial Intelligence Bill of Materials) generators for an ML supply chain

Incomplete Metadata

We observed that the documentation provided by model owners on HF is often incomplete. This is a trend not just for smaller, infrequently used projects but also for large projects with tens of thousands of downloads. For example, OpenAI's clip-vit-large-patch14-336 model, with 5,827,027 downloads, has a model card that is almost entirely incomplete, save for basic hyper-parameter and framework version information[27]. Only 37.8% of models and 27.6% of datasets declared any kind of licensing information in a machine-readable way. Some models may provide information on training data and architecture by linking to scientific papers that include these details, but such a practice introduces an additional hurdle in easily and programmatically obtaining that data. Ultimately, the reason why many models do not even mention the datasets used would require further investigation, including interviewing/surveying developers, which is out of the scope of this work.

During the manual analysis of 100 model cards we observed two examples where the HF team wrote documentation on behalf of model owners. The first instance was an image-to-text model uploaded by Microsoft with the disclaimer: "The team releasing TrOCR did not write a model card for this model so this model card has been written by the HF team"[24]. The second was a larger version of the same model, also owned by Microsoft[25]. In both cases, the models were described by a research paper and released on GitHub. Understanding the criteria that the HF team uses to determine when and how to intervene with model cards requires further investigation, but we do note that according to a blog written by members of the HF team, they created/updated model cards in some instances to inform design decisions surrounding a new model card template[6].

The Unknown License

The HF platform's documentation provides a list of recognized licenses each with a unique, standardized shorthand identifier that can be utilized by users when creating their model/data cards[8]. This list includes common Open-Source Software (OSS), Creative Commons (CC), and ML-specific licenses as well as an "Other" catch-all category which encompasses less common, custom, or modified licenses not included in the predefined license list. An "unknown" licensing option is also provided, but there is no guidance provided as to when this option should be selected. Future work will need to explore when and why model owners select this option, particularly given that acknowledging that the license is "unknown" suggests, at best, a lack of due diligence or understanding on the part of the owner or, at worst, that the model/dataset raises copyright infringement issues. We observed 4,419 (1.5%) models and 2,194 (4.5%) datasets that used this "unknown" license tag, which inherently raises compliance challenges for dependent models.

Naming Problems

HF users primarily use human-readable names, not unique identifiers, when supplying the metadata information for their base models/datasets. Unique

identifiers do exist for all models and datasets on HF, but we did not observe any instances where this information was being used by model/dataset owners when referring to models or datasets they depended on. Instead, these references typically, but not always, followed the "owner/model" naming convention. For example, a model owner would refer to a base model as "FacebookAI/xlm-roberta-base" and not by its unique HF ID: 621ffdc036468d709f174364. This could be problematic because these model references are not automatically updated when name changes occur. A change to the model, dataset, or owner name results in a new human-readable identifier that no longer matches the previous references to the model/dataset. As a result, documentation can easily become outdated, unhelpful, and potentially confusing.

We observed 596 models that had their human-readable identifiers changed. Instances such as 222gate/Blurdus-7b-v0.1 → gate369/Blurdus-7b-v0.1 involve a change to the name of the model owner. Other instances, such as aaditya/openbiollm-llama3-70b → aaditya/Llama3-OpenBioLLM-70B involve a change to the model name itself. In both cases, manual effort is required to track down and map the original references to the new names, which can make dependency management tasks increasingly difficult. Often the only way to determine the correct mapping is by relying on HF to redirect the page associated with a former name to the one associated with the most current name. This suggests that HF maintains some internal mapping, but it is unclear how long this mapping persists or what might happen in the event of a name collision.

Model names, without the additional owner information, are not guaranteed to be unique. Similar to how repository forks have the same name (but different owners) on GitHub, fine-tuned or forked models may also have the same name yet different owners on HF. For this reason, developers often specify models using the "owner/model" naming convention described previously. If this happens consistently, ambiguities can be avoided. However, we observed 16,477 cases where developers referred to base models only by the model name, excluding owner information. For example, a developer might use "roberta-base" to refer to "FacebookAI/roberta-base." While it could be reasonable to assume that this shorthand refers to the popular model, this may not be the case: in our dataset, there were 39 other models also named "roberta-base," all with different owners. If one of these models were referred to by name only, it would be nearly impossible to determine which model was being referenced. For example, more than 5,600 models share the model name "ppo-LunarLander-v2." This further stresses the need for widespread adoption of unique model identifiers.

Missing or Nonsensical References

There were 34,159 instances where we were unable to map a declared dataset to a dataset found publicly on HF. These consisted of 4,755 unique declarations. It is impossible to know with certainty, but such declarations likely consist of datasets that have been removed or made private, are from external sources, are actually dataset descriptors, contain typos, or, in some cases, are HF usernames.

We were also unable to map 2,501 base model declarations to known IDs, representing a possible 1,371 unique missing models in total. As with the datasets, these models may have been removed, made private, renamed in a way difficult to trace, or may reference a model outside the ecosystem. It is also possible that developer typos result in the inability to identify the models.

In other cases, some of the references were nonsensical: in our analysis of the graph of all collected components in the ML supply chain, we detected 684 cycles in total, including 675 (98.7%) trivial cycles in which a model declares itself as a base model, despite the fact that a model should not be able to be its own ancestor. All of this further motivates the need to use unique and standardized identifiers when referring to models and datasets, as well as a need for validating reference information.

Models as Datasets

We observed 1,416 instances in which a model was listed as a dataset in the datasets field. In other words, a model reference was included in the metadata field specific to training datasets. Without more information, it is impossible to know whether this indicates a mistake on the part of the model owner or that the output of the declared model was somehow used for training.

Shortcomings of Hugging Face

While base models are declared by some model owners, in many cases, the precise relationship between a derived model and its base model is left ambiguous, at least in the available metadata. HF does provide a separate field for architecture relationships, but there is no standardized way to specify situations involving fine-tuning, quantization, or using outputs for training. This ambiguity can have significant consequences since,

particularly in the licensing context, the nature of the relationship is important in determining how licensing terms should be applied. For example, it is permissible to use a model licensed under the llama3 license for fine-tuning but impermissible to use its outputs to train a competing model[23]. Additional fields specifying the nature of relationships between models would be useful, but their inclusion would also introduce additional overhead for model owners.

While uncommon and not a direct shortcoming of HF, we also observed examples where model owners would declare the same dataset or base model multiple times (i.e., the same model/dataset had multiple entries in the respective metadata field). Specifically, 178 models declared a dataset at least twice, and 310 models declared a base model at least twice. One model, with otherwise robust documentation, declared the same base model 64 times. Again, while there are few instances of this duplication, they still serve as something of a canary in the coal mine, indicating a lack of validation on behalf of HF.

RQ2: Structure of the ML Supply Chain

Lengths of model supply chains

We define a lineage chain as a path from some root base model (i.e., one with no model dependencies) to a final sink node model (i.e., one without any model dependents). We examine 53,151 lineage chains for models that declare at least one base model, including cases where multiple chains lead to the same model. Of these, the average chain length to reach that model is 6.2 models. The most frequent chain length is of three models (12,480 chains). The longest chain, beginning with cohereforai/c4ai-command-r-v01 and ending with Citaman/command-r-1-layer, contains 40 models. Excluding the first model, all models in this chain are owned by Citaman and incrementally count down from 39. That is, the models in the chain range from Citaman/command-r-39-layer down to Citaman/command-r-1-layer, suggesting that each model may represent an incremental improvement or an updated version.

Model ownership

We also observe that developers using HF often build off each other's work rather than off their own models. The owners of nodes at the end of a chain appeared within that node's chain only 1.4 times on average, showing that base models are frequently sourced from the community and supporting the idea that a common approach to training a model involves building off of previous work. Another interpretation could be that model trainers do not fill in base model information when building on their own work, at least some of the time.

In 82.9% of chains (43,507), the owner of the final model in the chain owned no other model in that chain. In the maximum case, the chain contained 39 models from the final model's owner. Illustrating this further, the first quartile, median, and third quartile are all a single occurrence of the final model's owner in a given chain: only the final model itself.

Based on our observations, a given owner publishes relatively few models to HF, owning, on average, 4 models in our dataset, though the data is skewed heavily toward relatively few highly prolific owners. Of 190,136 distinct model owners in our dataset, 61.5% owned just one public model (117,016), while the most prolific account owned 4,610 models. The first quartile and median are both just 1 owned model, and the third quartile is 2 models. Notably, the large players mentioned above, including Meta and OpenAI, are not present in the top 10 most prolific model owners: those spots instead go to smaller stakeholders in the AI market.

RQ3: Licensing of Models/Datasets on Hugging Face

The majority of models (62.2%) and datasets (72.4%) did not declare licensing information in a machine-readable way. To better illustrate how models and datasets are licensed, we organize the licenses that we observed into six categories. The categories are intended to be indicative of the origin or purpose of the licenses (whether the licenses were designed for open-source software, for ML components specifically, or other purposes) and are not intended to convey any legal characteristics, particularly given the uncertainty involved in applying these licenses in the ML context. We define the following categories:

- OSS: open-source software
- CC: Creative Commons licenses
- ML: ML-specific licenses (e.g. open-rail)
- Data: licenses to protect data (e.g. ODBL)
- Other: the "other" license category on HF
- Unknown: the "unknown" category on HF

Most common licenses

There were 71 unique declared licenses for models and 70 for datasets. Tables 1 and 2 provide an overview of the most frequently observed licenses.

Models			
License	**Type**	**Count**	**Percent (of licensed)**
apache-2.0	OSS	119,449	41.6%
mit	OSS	51,184	17.8%
openrail	ML	30,095	10.5%
creativeml-openrail-m	ML	20,396	7.1%
other	Other	16,447	5.7%
cc-by-nc-4.0	CC	8,318	2.9%
llama2	ML	6,158	2.1%
unknown	Unknown	4,419	1.5%
cc-by-4.0	CC	4,269	1.5%
llama3	ML	3,335	1.2%

Table 1: Top 10 licenses for models

Datasets			
License	**Type**	**Count**	**Percent (of licensed)**
mit	OSS	13,501	27.9%
apache-2.0	OSS	12,372	25.6%
openrail	ML	5,501	11.4%
cc-by-4.0	CC	2,673	5.5%
unknown	Unknown	2,194	4.5%
other	Other	1,837	3.8%
cc-by-sa-4.0	CC	1,256	2.6%
cc-by-nc-4.0	CC	1,014	2.1%
cc-by-nc-sa-4.0	CC	946	2.0%
cc0-1.0	CC	799	1.7%

Table 2: Top 10 licenses for datasets

Models			Datasets		
License Class	**Count**	**Percent**	**License Class**	**Count**	**Percent**
OSS	179,421	62.5%	OSS	28,047	58%
ML	66,540	23.2%	CC	9,151	18.9%
CC	19,856	6.9%	ML	6,481	13.4%
Other	16,447	5.7%	Unknown	2,194	4.5%
Unknown	4,419	1.5%	Other	1,837	3.8%
Data	326	0.1%	Data	559	1.2%
Combination	134	0.05%	Combination	87	0.2%
Total Licensed	287,143	100.0%	Total Licensed	48356	100.0%

Table 3: Prevalence of license classes

We classify licenses by the classes defined above. A breakdown of the prevalence of these classes can be found in Table 3. Both models (62.5%) and datasets (58%) are most likely to be licensed under approved OSS licenses. This is particularly interesting for datasets, where we would suspect to see mostly CC or Data licenses, as datasets are not "software" in a strict sense of the word. This may reflect that HF developers view models and datasets more similarly to programs than to data or that developers rely on OSS licenses because they are already familiar with their terms. However, the relationships between these preexisting licenses and these new types of components in the ML supply chain, as well as the implications of licensing such components with such licenses, are currently unknown.

Both models and datasets are distributed with (i) OSS licenses specific to software (both restrictive and permissive), (ii) Creative Commons (CC) licenses, that are not software-specific yet are being used for software (e.g., Stack Overflow's adoption of CC licenses), and ML-specific licenses. For models, our findings are consistent with those of Pepe et al.[29], showing that permissive OSS licenses such as Apache-2.0 and MIT are popular in this space. This is not surprising, as such licenses are also among the most popular for open-source projects[38] and allow commercial/closed-source exploitation. We note that, in general, the top licenses adopted on HF are traditionally understood to be more permissive, even if their exact application in this novel context is not fully understood.

We also note the adoption (for models, but especially for datasets) of different variants of the CC license, including a permissive one (CC-BY), restrictive ones (CC-BY-SA and CC-BY-NC-SA), and licenses limiting non-commercial use only (CC-BY-NC).

Besides OSS licenses, we see a large proportion of different kinds of ML-specific licenses. These include both licenses originating from open-source initiatives (openrail, creativeml-openrail-m), and licenses originating from companies (e.g., Meta's llama2 and llama3 licenses). Such licenses tend to differ from typical redistribution terms established by software licenses by, for example, introducing "behavioral" constraints; requiring responsible model usage; and prohibiting adoption for harmful or unethical uses or uses that do not take models' limitations into account. Likewise, the llama2 and llama3 licenses impose limitations related to commercial use, use in products or services having more than 700 million monthly active users, and use to train other models, unless such models are redistributed, under the same licenses, as derivative works of llama models.

Finally, we note the increased prevalence of HF's "Other" license which was ranked 7th in Pepe et al.[29] but has moved to the 5th spot since. In fact, 3.8% of datasets and 5.7% of models are licensed with some "Other" license. Prior work has found that such license proliferation can make license compliance tasks more time-consuming and difficult, since compliance teams are no longer dealing with known quantities[35].

Parent/child license differences

Our dataset contains 274,104 distinct parent/child relationships. We observed 66,460 instances (24%) where the licensing of a child model (i.e., derivative model) was different from that of its parent (i.e., base model). In nearly a third of cases, child models specified no

licensing information despite information being available for their parent(s), leading to a potential compliance violation. We also observe that once license information has been dropped, it is unlikely to be restored by later links in the chain, with license restoration behavior observed in only 15.9% of instances. A complete shift in licensing between parent and child was seen in 54.8% of cases (i.e., no licenses were shared between the two). Of these shifts, 18.5% involved the "unknown" or "other" license category. Table 4 shows the top ten such shifts, and Figure 2 provides more information on how licensing decisions shifted across class boundaries between parents and their children. While ML-specific licenses seem a good fit for models, model owners may be opting for licenses they are more familiar with (OSS for software developers and CC for data scientists). We observe that most shifts occur across the OSS and CC boundary (68.3%), perhaps suggesting some tension between the licensing preferences of those with data science and software development backgrounds. Additionally, depending on the specifics of the situation and the licenses involved, these shifts can also potentially introduce license incompatibilities, such as dropping non-commercial requirements or failing to apply the requirements of a copyleft license.

Parent License	Child License	Count	Percentage
cc-by-nc-4.0	apache-2.0	11,001	30.19%
apache-2.0	cc-by-nc-4.0	7,427	20.38%
cc-by-4.0	cc-by-nc-4.0	2,626	7.21%
other	llama3	1,720	4.72%
cc-by-nc-4.0	cc-by-4.0	1,444	3.96%
other	apache-2.0	1348	3.70%
apache-2.0	mit	1,241	3.41%
cc-by-nc-nd-4.0	cc-by-nc-4.0	1,180	3.24%
llama3	other	1,083	2.97%
apache-2.0	cc-by-4.0	782	2.15%

Table 4: Most Common parent/child license differences

Dataset license and model license combinations

The licenses of datasets that are used during training can have an impact on the licensing decisions for the final ML model. Here we consider the pairings of dataset licenses and the resultant model licenses. There are 623 distinct model/dataset license combinations within our dataset across 43,455 such pairings. (The top ten most frequent can be found in Table 5.) However, since we (and HF) aggregated all "other" licenses and treated them as one quantity, these numbers likely overestimate the consistency in the space. In 41 of the 623 combinations, the license of the model exactly matches the license of at least one dataset it was trained on. The most common combination is a model licensed under the apache-2.0 license and a dataset under a custom or "other" license. In total, 11,731 (27%) pairs involve a dataset under a custom or "other" license. This can make using these datasets and models problematic since retrieving and evaluating the licensing terms can be difficult and time-consuming.

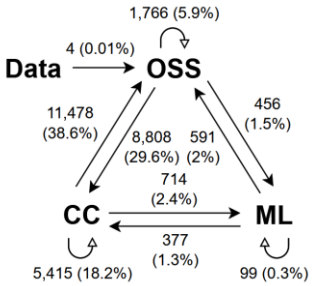

Figure 2: Parent/child license differences across classes

Model License	Dataset License	Count	Percentage
apache-2.0	other	8196	18.9%
apache-2.0	cc0-1.0	4621	10.6%
mit	mit	4479	10.3%
apache-2.0	apache-2.0	3916	9.0%
apache-2.0	unknown	2993	6.9%
apache-2.0	mit	2734	6.3%
mit	other	2290	5.3%
apache-2.0	cc-by-4.0	2076	4.8%
apache-2.0	cc-by-sa-4.0	1934	4.5%
mit	apache-2.0	1739	4.0%

Table 5: Model license and dataset license pairs

Other findings

Multi-licensing. A few models (188) and datasets (104) were released under more than one license. Multi-licensing is an existing phenomenon in the world of open-source software but is made more complex in the ML model context by the presence of novel license combinations that are not yet well understood. Examples include the 48 models we observed multi-licensed under apache-2.0 and cc-by-nc-4.0 as well as models under both OSS and ML licenses, such as MIT and OpenRAIL. We also observed datasets that were released under as many as six distinct licenses. The HF metadata provides no way of determining the relationship among these licenses, which could be either an AND or an OR. This distinction is critical to determining compliance. For example, if a user who uploaded a llama2 derivative model chose to make it available with an OR relationship between the apache-2.0 and llama2 licenses, this would contradict the exclusive llama2 licensing of its base model, resulting in a violation that could propagate to models further down the chain. While we observed only a few instances of multi-

licensing, such activity motivates the need to understand the interplay between the various license classes and for HF to supply a standardized way to specify the intended relationship between licenses.

Naming requirements. Some licenses, like Llama3, impose a generational limitation that specifies naming requirements for derivative works. According to the license, models that are built on a model under this license must have names prefixed with "llama3"[23]. In our dataset, there are 3,335 models licensed under llama3. However, of these, only 473 (14.2%) models correctly have names beginning with "llama3" as required by the license. Only 768 (23%) models even contain the term "llama3" in their name. Just 2,421 (72.6%) models contain the term "llama" at all, and 384 (11.5%) only contain the string "l3." The problem is likely even worse in practice, as these statistics only include models that documented that they were under the llama3 license.

Attribution. As was observed during the manual model card analysis, attribution appears to be important to model owners regardless of licensing status: rather than requesting or requiring attribution through a license, even models without a license often asked for some form of attribution (29/84), typically in the form of citations. In fact, 83,628 models (11.1%) mined using the HF API included a link to a paper on arxiv[3] in their tags. In our manual analysis of model cards, we observed that the model owner was typically also affiliated with (e.g., an author on) the linked research paper. This desire for recognition is reflected by the top licenses in the space, namely MIT and Apache-2.0, each of which specifies attribution requirements. Given the research community's adoption of HF, it may be worthwhile to add dedicated metadata fields to facilitate the standardization of citation information.

Discussion

Differences between ML and software supply chains

Our findings highlight a notable difference between the ML supply chain and the traditional software supply chain. In many software ecosystems with dependency management, dependency information for components in the software supply chain is organized in manifest files such as requirements.txt, package.json, or pom.xml. These files can then be used to generate other dependency tracking documents like SBOMs and have also been used by forges like GitHub to build dependency graphs. Notably, however, these manifest files have utility beyond dependency tracking. They are necessary for setting up a fresh installation of the software – that is, the list of dependencies must be downloaded in order for the software to function properly. This is not the case with the information supplied by developers in model cards. Because erroneous mappings to datasets and base models do not leave the model in question unusable as a practical matter, it is therefore easier for typos and other mistakes to go unnoticed, and thus uncorrected. This disconnect between correctness and functionality necessitates tooling that has similar functionality and richness to dependency management tools for traditional software, as it leaves ample opportunity for errors without comparable means of checking for and correcting them.

Potential difficulties in using model cards as MLBOM

Amidst the growing push for better transparency and security in software through Software Bills of Materials (SBOMs), calls have been made for similar BOMs for ML components[30]. These documents serve as inventories of all components within a piece of software and can document a variety of the software's traits, including dependencies, licensing, and security. Distinct ML/AI Bills of Materials (ML/AIBOMs) may be necessary to address such components' different inputs, security concerns (such as model poisoning), and ethical considerations. The SPDX Working Group, operating under the auspices of the Linux Foundation, is developing guidelines and a proposed standard for such an ML/AIBOM[2] that will complement the SPDX ISO Standard ISO/IEC 5962:2021, which describes the use of SBOMs to document the components used in creating a software system. CycloneDX is also working on support for ML/AIBOMs[5].

Prior work has also suggested that model and data cards could serve as an ML/AIBOM[30], or the information they provide can assist with their creation[29], but our work suggests that these tools are not yet robust enough to serve this purpose. As highlighted previously, the information provided by model cards is often missing key elements, including datasets that the models were trained on. This immediately obviates many of the benefits of ML/AIBOMs, including understanding licensing obligations that might be associated with that dataset, being aware of potential model poisoning attacks, and providing the ability to select models trained on ethically sourced data. Additionally, the cases where model cards were locked behind Terms of Service agreements or other restrictions could further limit their usefulness as ML/AIBOMs to consumers. Our work provides additional evidence that, in practice, model

cards contain little actionable information, making them difficult to use as ML/AIBOMs[3].

Bibliography

[1] Sina Ardabili, Amir Mosavi, and Annamária R Várkonyi-Kóczy. 2019. Advances in machine learning modeling reviewing hybrid and ensemble methods. In International conference on global research and education. Springer, 215–227.
[2] Karen Bennet, Gopi Krishnan Rajbahadur, Arthit Suriyawongkul, and Kate Stewart. 2024. A Comprehensive Guide to Creating AI and Dataset Bill of Materials. https://spdx.dev/implementing-an-ai-bom/
[3] Avinash Bhat, Austin Coursey, Grace Hu, Sixian Li, Nadia Nahar, Shurui Zhou, Christian Kästner, and Jin LC Guo. 2023. Aspirations and practice of ml model documentation: Moving the needle with nudging and traceability. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. 1–17.
[4] Joel Castaño, Silverio Martínez-Fernández, and Xavier Franch. 2024. Lessons Learned from Mining the Hugging Face Repository. arXiv preprint arXiv:2402.07323 (2024).
[5] cyclonedx 2025. Machine Learning Bill of Materials (ML-BOM). https://cyclonedx.org/capabilities/mlbom/
[6] Margaret Mitchell Ezi Ozoani, Marissa Gerchick. 2022. Model Cards. https://huggingface.co/blog/model-cards.
[7] Hugging Face. [n. d.]. User Studies. https://huggingface.co/docs/hub/en/model-cards-user-studies.
[8] Hugging Face. 2024. Licenses. https://huggingface.co/docs/hub/en/repositories-licenses.
[9] Hugging Face. 2024. Model Card Template. https://github.com/huggingface/huggingface_hub/blob/main/src/huggingface_hub/templates/modelcard_template.md.
[10] Hugging Face. [n.d.]. Model Cards Writing Tool. https://huggingface.co/spaces/huggingface/Model_Cards_Writing_Tool.
[11] Kai Gao, Runzhi He, Bing Xie, and Minghui Zhou. 2024. Characterizing Deep Learning Package Supply Chains in PyPI:
Domains, Clusters, and Disengagement. ACM Trans. Softw. Eng. Methodol. 33, 4 (2024), 97:1–97:27.
[12] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. 2022. A survey of quantization methods for efficient neural network inference. In Low-Power Computer Vision. Chapman and Hall/CRC, 291–326.
[13] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Xiaodong Song, Aleksander Madry, Bo Li, and Tom Goldstein. 2020. Dataset Security for Machine Learning: Data Poisoning, Back-door Attacks, and Defenses. IEEE Transactions on Pattern Analysis and Machine Intelligence 45 (2020), 1563–1580.
[14] Ahmed E Hassan, Dayi Lin, Gopi Krishnan Rajbahadur, Keheliya Gallaba, Filipe Roseiro Cogo, Boyuan Chen, Haoxiang Zhang, Kishanthan Thangarajah, Gustavo Oliva, Jiahuei Lin, et al . 2024. Rethinking software engineering in the era of foundation models: A curated catalogue of challenges in the development of trustworthy fmware. In Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering. 294–305.https://api.semanticscholar.org/CorpusID:229934464
[15] Geoffrey Hinton. 2015. Distilling the Knowledge in a Neural Network. arXiv preprint arXiv:1503.02531 (2015).
[16] Wenxin Jiang, Nicholas Synovic, Purvish Jajal, Taylor R Schorlemmer, Arav Tewari, Bhavesh Pareek, George K Thiruvathukal, and James C Davis. 2023. PTMTorrent: A Dataset for Mining Open-source Pre-trained Model Packages. In 2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR). 57–61.
[17] Wenxin Jiang, Nicholas Synovic, Rohan Sethi, Aryan Indarapu, Matt Hyatt, Taylor R Schorlemmer, George K Thiruvathukal, and James C Davis. 2022. An empirical study of artifacts and security risks in the pre-trained model supply chain. In Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses.
[18] Wenxin Jiang, Jerin Yasmin, Jason Jones, Nicholas Synovic, Jiashen Kuo, Nathaniel Bielanski, Yuan Tian, George K Thiruvathukal, and James C Davis. 2024. Peatmoss: A dataset and initial analysis of pre-trained models in open-source software. In 2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR). IEEE, 431–443.
[19] Keita Kurita, Paul Michel, and Graham Neubig. 2020. Weight Poisoning Attacks on Pretrained Models. ArXiv abs/2004.06660 (2020). https://api.semanticscholar.org/CorpusID:215754328
[20] Katherine Lee, A Feder Cooper, and James Grimmelmann. 2023. Talkin' 'Bout AI Generation: Copyright and the Generative-AI Supply Chain. arXiv preprint arXiv:2309.08133 (2023).
[21] Tshilidzi Marwala, Eleonore Fournier-Tombs, and Serge Stinckwich. 2023. The use of synthetic data to train ai models: Opportunities and risks for sustainable development. arXiv preprint arXiv:2309.00652 (2023).
[22] Saeed Masoudnia and Reza Ebrahimpour. 2014. Mixture of experts: a literature survey. Artificial Intelligence Review 42 (2014), 275–293.
[23] Meta. 2024. META LLAMA 3 COMMUNITY LICENSE AGREEMENT. https://llama.meta.com/llama3/license/.
[24] Microsoft. [n.d.]. trocr-base-handwritten. https://huggingface.co/microsoft/trocr-base-handwritten.
[25] Microsoft. [n.d.]. trocr-large-stage1. https://huggingface.co/microsoft/trocr-large-stage1.
[26] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer,
Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model cards for model reporting. In Proceedings of the conference on
fairness, accountability, and transparency. 220–229.
[27] OpenAI. [n. d.]. clip-vit-large-patch14-336. https://huggingface.co/openai/clip-vit-large-patch14-336.
[28] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud
Texier, and Jeff Dean. 2021. Carbon Emissions and Large Neural Network Training. http://arxiv.org/abs/2104.10350
[29] Federica Pepe, Vittoria Nardone, Antonio Mastropaolo, Gabriele Bavota, Gerardo Canfora, and Massimiliano Di Penta. 2024. How do Hugging Face Models Document Datasets, Bias, and Licenses? An Empirical Study. In Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension. 370–381.
[30] Trevor Stalnaker, Nathan Wintersgill, Oscar Chaparro, Massimiliano Di Penta, Daniel M German, and Denys Poshyvanyk. 2024. BOMs Away! Inside the Minds of Stakeholders: A Comprehensive Study of Bills of Materials for Software Systems. In Proceedings of the 46th IEEE/ACM International Conference on Software Engineering. 1–13.
[31] Xin Tan, Kai Gao, Minghui Zhou, and Li Zhang. 2022. An exploratory study of deep learning supply chain. In Proceedings of the 44th International Conference on Software Engineering. 86–98.
[32] Mistral AI team. 2023. Mixtral of experts. https://mistral.ai/news/mixtral-of-experts/.
[33] James Vincent. 2022. The lawsuit that could rewrite the rules of AI copyright. www.theverge.com/2022/11/8/23446821/microsoft-openai-github-copilot-class-action-lawsuit-ai-copyright-violation-training-data.
[34] Zhi Wang, Chaoge Liu, and Xiang Cui. 2021. EvilModel: Hiding Malware Inside of Neural Network Models. 2021 IEEE Symposium on Computers and Communications (ISCC) (2021), 1–7. https://api.semanticscholar.org/CorpusID:236087474
[35] Nathan Wintersgill, Trevor Stalnaker, Laura A Heymann, Oscar Chaparro, and Denys Poshyvanyk. 2024. "The Law Doesn't Work Like a Computer": Exploring Software Licensing Issues Faced by Legal Practitioners. Proceedings of the ACM on Software Engineering 1, FSE (2024), 882–905. (ICSE). IEEE, 2630–2642.
[36] Kenneth Ward Church, Zeyu Chen, and Yanjun Ma. 2021. Emerging trends: A gentle introduction to fine-tuning. Natural Language Engineering 27, 6 (2021), 763–778.
[37] Trevor Stalnaker, Nathan Wintersgill, Oscar Chaparro, Laura Heymann, Massimiliano Di Penta, Daniel M German, and Denys Poshyvanyk. 2025. Online replication package. https://archive.softwareheritage.org/browse/origin/directory/?origin_url=https://github.com/TStalnaker44/hugging_face_analysis_replication.
[38] Christopher Vendome. 2015. A large scale study of license usage on github. In 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Vol. 2. IEEE, 772–774.
[39] Stalnaker, T., Wintersgill, N., Chaparro, O., Heymann, L. A., Di Penta, M., German, D. M., & Poshyvanyk, D. (2025). The ML Supply Chain in the Era of Software 2.0: Lessons Learned from Hugging Face. *arXiv preprint arXiv:2502.04484*.