How do Hugging Face Models Document Datasets, Bias, and Licenses? An Empirical Study

Federica Pepe
University of Sannio
Benevento, Italy

Vittoria Nardone
University of Molise
Campobasso, Italy

Antonio Mastropaolo
SEART @ Software Institute
Università della Svizzera italiana
Lugano, Switzerland

Gerardo Canfora
University of Sannio
Benevento, Italy

Gabriele Bavota
SEART @ Software Institute
Università della Svizzera italiana
Lugano, Switzerland

Massimiliano Di Penta
University of Sannio
Benevento, Italy

ABSTRACT

Pre-trained Machine Learning (ML) models help to create ML-intensive systems without having to spend conspicuous resources on training a new model from the ground up. However, the lack of transparency for such models could lead to undesired consequences in terms of bias, fairness, trustworthiness of the underlying data, and, potentially even legal implications. Taking as a case study the transformer models hosted by Hugging Face, a popular hub for pre-trained ML models, this paper empirically investigates the transparency of pre-trained transformer models. We look at the extent to which model descriptions (i) specify the datasets being used for their pre-training, (ii) discuss their possible training bias, (iii) declare their license, and whether projects using such models take these licenses into account. Results indicate that pre-trained models still have a limited exposure of their training datasets, possible biases, and adopted licenses. Also, we found several cases of possible licensing violations by client projects. Our findings motivate further research to improve the transparency of ML models, which may result in the definition, generation, and adoption of Artificial Intelligence Bills of Materials.

CCS CONCEPTS

• **Software and its engineering → Software libraries and repositories**.

KEYWORDS

ML-Intensive Systems, Pre-trained models, Transparency, Bias, and Fairness, Deep Learning, Empirical Study

ACM Reference Format:
Federica Pepe, Vittoria Nardone, Antonio Mastropaolo, Gerardo Canfora, Gabriele Bavota, and Massimiliano Di Penta. 2024. How do Hugging Face Models Document Datasets, Bias, and Licenses? An Empirical Study. In *32nd IEEE/ACM International Conference on Program Comprehension (ICPC '24), April 15–16, 2024, Lisbon, Portugal*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3643916.3644412

ICPC '24, April 15–16, 2024, Lisbon, Portugal

ACM ISBN 979-8-4007-0586-1/24/04.
https://doi.org/10.1145/3643916.3644412

1 INTRODUCTION

Pre-trained Machine Learning (ML) models represent key reusable assets when developing ML-intensive software systems. A pre-trained model is a ML model that has been trained for one or more (relatively generic) tasks, or simply to "learn" a given language. Subsequently, developers/data scientists have the option to fine-tune these models, customizing them for specific tasks. Such a specialization requires fewer resources—in terms of training data, computational power, and time—than training a model from scratch. To make a metaphor, it would be like teaching math to a child who already speaks English rather than teaching it to a child who does not speak at all.

Inspired by the reuse of software libraries that happens through package managers (*e.g.,* Maven Central, npm, or PyPi), or of containerization images, via specific forges like Docker Hub, Hugging Face (HF) [25] has created a hub for pre-trained models to democratize Artificial Intelligence (AI). HF provides (i) reusable pre-trained models for different purposes (*e.g.,* natural language, image, or audio processing), (ii) datasets for model training, and (iii) a high-level API for conveniently using the models through back-ends such as Keras/Tensorflow or PyTorch. Forges similar to HF are Model Zoo [31], Tensorflow Hub [19], and PyTorch Hub [47].

Recently, researchers in software engineering have been examining the hurdles associated with employing HF models. Some research has studied in general how models are documented, and provided templates [35] and approaches/tools to guide models' documentation [3, 12, 48]. Other work focuses on security concerns [26] or on the environmental impact of the models' training [7]. That being said, there are other crucial aspects that a developer aiming to reuse a pre-trained ML model must take into account. These aspects include (i) the extent to which models document and provide access to the datasets used for training; (ii) the potential for biases in the models resulting from their pre-training process; and (iii) the presence of compatibility issues between the model's license and software licenses.

This paper presents the findings of an empirical investigation that focuses on the analysis of pre-trained transformer models hosted on the HF Hub. The study aims to achieve the following goals: (i) determine the extent to which and how the models document and provide access to their training data; (ii) assess the extent to which models are documented with potential biases they could suffer from; and (iii) identify licensing incompatibilities that may arise when GitHub projects reuse HF models. Overall, the study has been conducted by mining 159,132 models hosted on HF, and 44,823 open-source projects hosted on GitHub reusing such models.

Results of the study indicate that, so far, there is still limited transparency by the models in documenting the datasets on which they have been trained, and the biases to which they are subject. Concerning licenses, models tend to adopt permissive ones, and even ML-specific licenses, encompassing a "responsible" use of such models. Nevertheless, we found that software projects using models still create several potential cases of licensing incompatibilities.

This study has multiple implications. First of all, it provides empirical evidence about the extent to which training datasets, bias issues, and licensing are documented by pre-trained models. This raises the need for better documenting these models, as results indicate, very often, a scarce level of documentation. Second, the results indicate how the reuse of pre-trained models could possibly generate bias if no proper countermeasures are taken. Third, the analysis we performed on the type of documentation available with pre-trained models opens the road for better defining, and possibly automatically generating, bills of material for AI-based systems (AIBOMs).

Overall, the paper's contributions can be summarized as follows:

(1) We analyze whether and how pre-trained transformer models document the datasets they have been trained on.
(2) We analyze and discuss whether models document their bias and the type of bias they declare.
(3) We provide information about the licensing of HF models, as well as potential licensing incompatibilities of projects using such models.

The paper is organized as follows. Section 2 describes the study design, including the data extraction and analysis process. Results are presented and discussed in Section 3, while their implications are discussed in Section 4, and their threats in Section 5. After Section 6 discusses related literature, Section 7 concludes the paper, and outlines directions for future work.

2 STUDY DESIGN

The *goal* of the study is to analyze pre-trained machine learning models hosted by HF, to understand the extent to which and how they document training datasets, potential biases derived from their training, and their licenses. The *quality focus* is the models' transparency, the lack of which could cause unexpected/unfair behavior (*e.g.,* due to bias), security issues, or legal problems. The *perspective* is of researchers, that would like to define approaches, including AIBOM generators, aimed at enhancing the transparency of ML-based systems. The *context* consists of 159,132 pre-trained *transformer* models hosted on HF, and of 17,365 open-source projects hosted on GitHub and using (a subset of) such models.

The study aims to address the following research questions:

- **RQ**$_1$ *To what extent do models declare the datasets used for their training?* We look at the pre-training transparency from its primary element, *i.e.,* the dataset(s) used to train the models. The lack of dataset declaration, or, a vaguely-specified, inaccessible dataset reduces the transparency of a pre-trained ML model.
- **RQ**$_2$ *How do pre-trained models discuss fairness limitations?* We conduct a qualitative assessment to determine the level of bias declaration in pre-trained models and, if present, the nature of the expected bias. This information would be valuable to model users as it enables them to safeguard their software against biases, *e.g.,* through appropriate model fine-tuning or employing other approaches for bias prevention/mitigation. [5, 8, 18, 22, 36, 40].
- **RQ**$_3$ *To what extent do models declare their licenses, and to what extent do such licenses lead to potential incompatibilities among client projects?* We analyze the models' documentation from a legal perspective by looking at (i) the models' declared licenses, and (ii) the relationship between the models' licenses and the client projects' ones, to identify possible incompatibilities.

Table 1: Number of HF models downloaded by task.

Tasks	#Models	# of Downloads					
		Min	1Q	Median	Mean	3Q	Max
NLP	62, 416	0	1	3	4, 292	11	47, 032, 389
RL	15, 431	0	0	0	18	2	66, 658
Audio	7, 918	0	2	4	1693	8	10, 402, 498
Multimodal	6, 212	0	0	3	4, 466	17	3, 237, 345
CV	3, 867	0	1	3	4728	53	10, 487, 900
Tabular	175	0	0	0	2, 571	1	205
Other	4	7	11	21	119	129	428
Not Available	36, 109	0	0	0	562	0	7, 707, 765
Summary	159, 132	0	0	0	2, 281	5	47, 032, 389

2.1 Context Selection and Data Extraction

Our study encompasses two main components: firstly, a dataset of pre-trained *transformer* models available on the HF hub, and secondly, a collection of open-source projects hosted on GitHub. The data for this study was collected in February 2023. We have chosen to study the transformer models because of their wide number of applications in various areas, including among others NLP, multimedia processing, and reinforcement learning.

To download the HF data, we leverage the HF Endopoints API [24]. Specifically, we perform a query to retrieve a JSON entry for all models hosted on HF. Each JSON entry contains information including the last modification date, model tags, numbers of downloads and likes, etc. In total, we collect data for 159,132 HF models.

Table 1 shows the distribution of downloaded HF models organized by task, and specifically Natural Language Processing (NLP), Reinforcement Learning (RL), Audio, Multimodal (*e.g.,* text-to-image generation), Computer Vision (CV), Tabular (tabular classification and tabular regression), and Other (including time series forecasting). The table also reports descriptive statistics (minimum, 1st quartile, median, mean, 3rd quartile, and maximum) of the models' downloads for the different task categories. As it can be noticed, the distribution is very skewed, as the majority of the models have a very small number of downloads.

To compute the distribution of Table 1, we start from the model modality, *i.e.,* the value of `pipeline_tag` feature in the JSON file, and we assign the related task as shown above. For example, if the

Table 2: GitHub projects using the HF `transformers` library

#Dependent Prjs	#from_pretrained Usage	#Prjs re-using HF models
44, 823	33, 249	17, 365

model under analysis has the `pipeline_tag` equal to text classification, then this model contributes to the count of models addressing the Natural Language Processing task. When a model does not specify the pipeline tag, we place it in a separate category called *Not Available*. As Table 1, this happens for $\sim 23\%$ of the models.

The second step of our data extraction aims at identifying GitHub projects[1] gathered by following the process described above using pre-trained models from HF. To achieve this goal, we leverage the *Dependents* feature available in GitHub, which, given a repository, lists all other repositories that depend on it. For Python projects, GitHub computes dependents by analyzing the projects' package dependencies (*e.g.,* `requirements.txt`). Specifically, we gather the list of dependents from the GitHub *huggingface/transformers* project. To this aim, we leverage *Beautiful Soup* and *requests* Python packages. More in detail, we use the former to extract the projects list from the HTML pages and the latter to perform GET requests to navigate the result pages. This is necessary as the GitHub API currently does not provide a query to collect dependents' information. In total, we collect 44,823 projects depending on the HF transformers library. Starting from this list of projects, we clone each project, and then, we search, in all Python files, for all occurrences of the `.from_pretrained` function invocation, *i.e.,* the function that imports a pre-trained model. Among 44,823 dependent' projects, 33,249 use the `.from_pretrained` function at least once. Then, we filter such occurrences and collect only the projects for which the `.from_pretrained` refers to one of the HF models gathered in the previous step. At the end of this process, we collect 17,365 projects which re-use a pre-trained model belonging to the HF transformers library. As it will be discussed in Section 5, we are aware that this analysis may be imprecise and above all lead to false negatives, as (i) the string passed to the `.from_pretrained` function could be dynamically created; and (ii) in principle, yet less likely for this particular circumstance, the `.from_pretrained` could be assigned to a variable and passed to a function.

As reported in Table 2, we found 44,823 projects hosted on GitHub depending on the HF *transformers* library. Of these, we could match 17,365 onto pre-trained models hosted on the HF Hub. Those that were not matches could be (i) repositories having a dependency on *transformers* yet never using it; (ii) cases where the *transformers* library was used, *e.g.,* to create a tool on top of HF, yet no model was explicitly imported; and (iii) more importantly, cases where the model name string was passed as variable to `.from_pretrained` and, due to our simplified analysis (or just because the model name was never present in the source code), we could not match any model.

Several projects use more than one model. The number of reused models per project has a median equal to 1, with a first and third

[1]Although the correct GitHub term is "repository", in the following we will refer to them as "project".

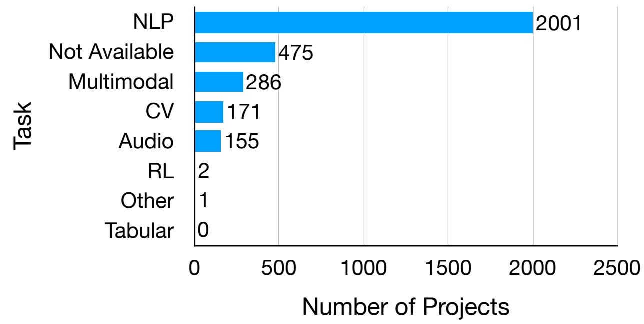

Figure 1: Distribution of projects' model use by tasks.

quartile equal to 1 and 2 respectively, and a large number of outliers (2449 out of 17,365 projects).

The maximum number of models used by a project is 463, and this happens for the `aarnphm/transformers` project, which is a fork of the transformers library used to contribute to it.

Furthermore, we found that Spearman's rank correlation between models' usage by GitHub projects (*i.e.,*, the number of GH projects reusing a specific model) and their number of downloads (*i.e.,*, the number of downloads per model on HF) is $\rho = 0.40$, indicating a moderate correlation. This means that, while the number of downloads is not a perfect indicator of project usage, it correlates enough with it to be used as a good indicator of models' popularity in our analyses.

Figure 1 reports the distribution of model usages across different tasks. Consistently to what was mentioned before, the largest number of model usages is related to NLP, with 2,001 projects reusing such models. 475 projects use models with no specified task. Multimodal, CV, and Audio follow, with 286, 171, and 155 projects respectively. We found a few cases of usage for RL and other categories and no reuses for Tabular models. Note that 42 models are missing from this list as their metadata could not be retrieved.

Finally, for each project linked to the HF models, we use the *Perceval* [15] tool to retrieve its metadata from these repositories. Other than for capturing general information about the repositories, the metadata will be used to identify the projects' declared license(s). To assess the reliability of the licensing in the metadata, we extracted a (randomly stratified over license types) statistically significant sample of projects, for a confidence level of 95% and a margin of error of ±5%. Given the sample size of 383 projects, we approximated by excess decimal numbers and analyzed all samples when the number of projects per license was 5 or less. In total, we obtained a sample of 406 projects. One author manually analyzed the LICENSE files and compared them with the licensing in the metadata. The task was fairly simple and not subjective, so to not require multiple annotators. Only in one case out of 406 (0.24%) we found an inconsistency. Therefore, we can assume that the licensing information in the metadata is reliable enough.

Table 3: Descriptive statistics of RQ_1 and RQ_2 sample.

Tasks	#Models	# of Downloads					
		Min	1Q	Median	Mean	3Q	Max
NLP	249	66,743	111,393	199,934	1,007,266	503,840	47,032,389
RL	62	551	555	560	2,194	578	66,658
Audio	32	16,145	21,704	36,099	401,161	66,389	10,402,498
Multimodal	25	216,421	281,238	520,068	724,345	800,580	3,237,345
CV	16	87,907	115,650	171,089	1,006,335	238,920	10,487,900
Tabular	1	205	205	205	205	205	205
Other	4	7	11	21	119	128	428
Summary	389	7	68,897	148,329	766,048	359,588	47,032,389

2.2 Analysis Methodology

To address RQ_1, we perform two types of analyses. The first one, performed on all 159,132 models, aims at reporting statistics about the number of models that include specific *tags* related to datasets (those starting with `"dataset:"`, which appear on the model page as a small cylinder). However, this only gives a partial picture of how models have been trained because a model's owner could specify the datasets somewhere else in the model card description, without following a specific pattern. For this reason, we decided to answer this question by manually analyzing a statistically significant sample of the model cards.

Given the large number of collected model cards (159,132), we extracted a stratified sample, using the tasks in Table 1 as strata and ranking models in decreasing order of download. In other words, we do not perform a random sampling over the strata, but, rather, an "intensity" sampling by the number of downloads. We considered a sample of 389 models, which ensures a ±5% margin of error with a confidence level of 95%. The required sample size is 384, yet we rounded in excess the size of each stratus and included all 4 cases of the "Other" category. To perform a stratified sampling, models have been ordered in descending order of downloads and then sampled proportionally to the stratus size. We preferred such a criterion for the sampling over a random sampling, as we want our results to be representative of models actually used by projects, rather than considering the level of documentation of models that nobody uses.

As explained in Section 2.1 an "intensity" sampling performed based on the number of downloads represents a proxy of model usage.

Table 3 reports, similarly to Table 1, descriptive statistics for the sampled models.

Then, three authors (annotators) performed a manual analysis of the sample model cards, with each card being independently assessed by two of the authors. Each annotator opened the card and looked at whether or not it declared the training through (i) datasets hosted by HF, (ii) datasets for which external links are provided, or (iii) the type of data used for the training is mentioned, yet no link is provided. Note that the three classifications are not mutually exclusive, as models could have been trained with multiple datasets for which different sources were provided. The manual classification was performed through online spreadsheets. After the classification has been completed, for each column, Cohen's k inter-rater agreement [10] was computed, to determine the reliability of the manual assessment. We obtained Cohen's k=0.93 for the first category, 0.75 for the second, and 0.71 for the third one, hence indicating enough reliability in the classification. Then, the manual annotators jointly resolved the conflicting cases.

We report, for the analyzed sample, the number and percentage of models (overall and along the different categories) for which there is some dataset information, and, for the latter, the number and percentage of models that (a) use datasets hosted on HF, (b) provide a link to an external dataset, (c) provide generic information about the training dataset.

To address RQ_2, first of all, we looked for pre-existing taxonomies of ML model biases. Based on the existing literature, we rely on a bias taxonomy from a systematic literature review on the topic conducted by Mehrabi *et al.* [34], as it is pretty comprehensive in that regard. The proposed taxonomy foresees three high-level categories, namely: (i) *Data to Algorithm* (*e.g.,* concerning how data is measured and characterized in terms of variables), (ii) *Algorithm to User* (related to the algorithmic outcome of a model when a user interacts with it), and (iii) *User to Data* (concerning model's data being user-generated or user-related). Such three categories are then further detailed into sub-categories.

However, the category *Data to Algorithm* does not apply, at least directly, to our scope (if anything, it could be used in a further study on HF datasets). As for the *Algorithm to User* category, while model cards discuss "intended uses" they do not discuss interaction bias. Instead, we included in our taxonomy *Popularity Bias*, which concerns (generic cases of) items present in the dataset with a high frequency.

At the same time, after a first partial scrutiny of HF model cards, we found that the types of bias are described at a finer-grained level than what foreseen in the paper by Mehrabi *et al.* for the *User to Data* category. Therefore, we decided to manually code the bias declared in the model cards, and then to map those levels of biases onto the 7 sub-categories of *User to Data* by Mehrabi *et al.*: Historical (related to peculiar data distribution in certain periods of time), Population (intrinsic characteristics of a sub-population), Self-selection (*e.g.,* subjects participate in a study based on their interest), Social (related to social characteristics), Behavioral (different user behavior in different contexts), Temporal (differences in the population over periods of time), and Content Production (*e.g.,* related to language differences).

Similarly to RQ_1, to categorize models' bias it is not possible to perform a reliable, automated classification of all model cards. Therefore we opted, again, for a manual analysis, considering the same sample of RQ_1. For each model, each annotator, first of all, indicated on the online spreadsheet whether or not the model card was describing bias-related information. Then, each annotator indicated on the spreadsheet the type of bias described. Since we had no predefined categories for that, we adopted an open card sorting [45] procedure. Each annotator used a column of the online spreadsheet to select the bias category from a different sheet that was jointly populated by the three annotators (*i.e.,* a new category was added when the available categories did not fit with what was reported in the model card). Multiple entries were added to the spreadsheet when a model card described multiple types of bias. To agree on the annotation criteria and create a first set of categories, the three annotators jointly annotated 20 model cards, and then continued independently.

After the annotation was completed, the annotators met to resolve the conflicting cases. Also, they jointly reviewed the list of categories and mapped them onto the categories of Mehrabi *et al.*.

Also for RQ_2, we assessed manual classifications in two ways, *i.e.,* by computing (1) Cohen's k on the Boolean classification on the presence of bias, and (2) Krippendorff's α [32] for the categories, because each annotator added multiple labels, and there could be cases where one annotator added a label and the other did not, resulting in a (*categ*, N/A) pair in the inter-rater computation table. In this circumstance, Krippendorff's α is suitable as it handles incomplete ratings. We achieved a Cohen's k of 0.91, and a Krippendorff's α of 0.93, both indicators of a very strong agreement.

For the overall sample and for each model task category, we report the number and type of models declaring some bias. Then, we describe and discuss the bias categories found during the analysis. In doing so, we provide examples of biases for the different categories, also explaining how the models detail such bias by providing input examples and expected outputs exhibiting the bias.

To address RQ_3, we first report the distribution of licenses among the HF models also highlighting how many models do not declare any license. To perform this analysis we check the `license` tag contained in the tags list of the JSON file of the model. We report results at two levels of abstraction, *i.e.,* by first distinguishing between different categories of licenses in terms of permissiveness (the mapping between these levels and specific licenses is in our replication package), *i.e., network restrictive* (*e.g.,* AGPL), *restrictive* (*e.g.,* GPL), *weakly restrictive* (*e.g.,* LGPL or MPL), and *permissive* (*e.g.,* Apache, MIT, or BSD), and then in terms of specific licenses.

Then, still using the license categorization, we report the relationships between models' licenses and client projects' licenses, highlighting relations that would lead to incompatibilities. For those, we further detail and discuss the specific licenses used on both sides.

3 STUDY RESULTS

This section discusses results addressing the three research questions formulated in Section 2. Note: To access each model card mentioned as *OWNER/REPO_NAME*, one can use the following link *https://huggingFace.co/OWNER/REPO_NAME* (without OWNER if the latter is not specified). Similarly, GitHub projects (mentioned in RQ_3) can be accessed as *https://github.com/OWNER/REPO_NAME.*

Table 4: Documented datasets from the models' sample

Task	Labeled	HF	External	No Link	Any	%
NLP	249	100	51	55	174	69.87
RL	62	0	0	0	0	0
Audio	32	15	8	5	21	65.62
Multimodal	25	3	12	2	14	56
CV	16	4	4	7	12	75
Other	4	2	0	3	4	100
Tabular	1	1	0	0	1	100
Total	389	125	75	72	226	58.09

3.1 RQ_1: Training Datasets

Among the 159,132 analyzed models, only 22, 572 (14.08%) specify the dataset(s) used during the training phase using the `dataset:` tag. Of these, 15, 437 only rely on datasets hosted on HF, 6,591 provide a link to external dataset(s), and 544 provide both types of datasets. The tag distribution is lowly-skewed, with first, second, and third quartile=1, and a maximum of 135 for *sileod/deberta-v3-base-tasksource-nli*.

As models could specify datasets somewhere else in the text of the model card, we perform a manual analysis on 389 model cards. Results are reported in Table 4. As it can be noticed, being these the top-most downloaded models, a fairly high number 226 (58.09%) of them provide some dataset documentation. Of these 226, only 143 (63%) belong to the 22, 572 above, *i.e.,* use the `dataset:` tag to document the dataset.

125 models in the sample only leverage internally-hosted datasets (that are easier to retrieve and use by the users through the HF *datasets* library), while 75 provide external links, and 72 reference dataset names without providing links. Note that the sum of columns HF, External, and No Link exceeds the total # of models with declared datasets, as some models may declare multiple datasets in different ways.

While, for most task types, $\sim$ 60% of the models declare datasets, this is never the case for RL models. The reason can be the learning strategy of such models: An agent interacts with the environment by taking actions, and in response, the environment provides a reward signal indicating how favorable the agent's actions are. Such environment data seems not to be released. However, for 56 out of 62 RL models in our sample, there is a link to a GitHub repository (*huggingface/ml-agents*) containing environments on which RL models can be trained.

Some models declare several training datasets from different sources. For example, the *sentence-transformers/all-mpnet-base-v2* model is a sentence transformer used to convert sentences and paragraphs. The training combines multiple datasets through concatenation, resulting in over 1 billion sentence pairs. In total, it incorporates 26 datasets from HF and 23 datasets from external sources for which a link is provided.

Other models mention datasets without sharing them. For example, the *gpt2-medium* model card mentions an external dataset named WebText, which comprises 40 GB of textual data, yet it has not been made publicly available.

Along a similar line, *finiteautomata/bertweet-base-sentiment-analysis* is a BERT model fine-tuned on Tweets for sentiment analysis. As Tweets datasets cannot be shared any longer, the authors mention *"Please be aware that models are trained with third-party datasets and are subject to their respective licenses."*

Answer to RQ_1. Only 14% of the analyzed models declare datasets as tags in the model cards. However, in a sample of 389 top-downloaded models, 61% declare datasets in some way.

3.2 RQ_2: Bias Description

Table 5 illustrates the total number of models assessed for each task, along with the number of models with no detected bias and the number of models exhibiting at least one bias. In total, we manually inspected 389 models, of which only 72 (18%) were found to expose

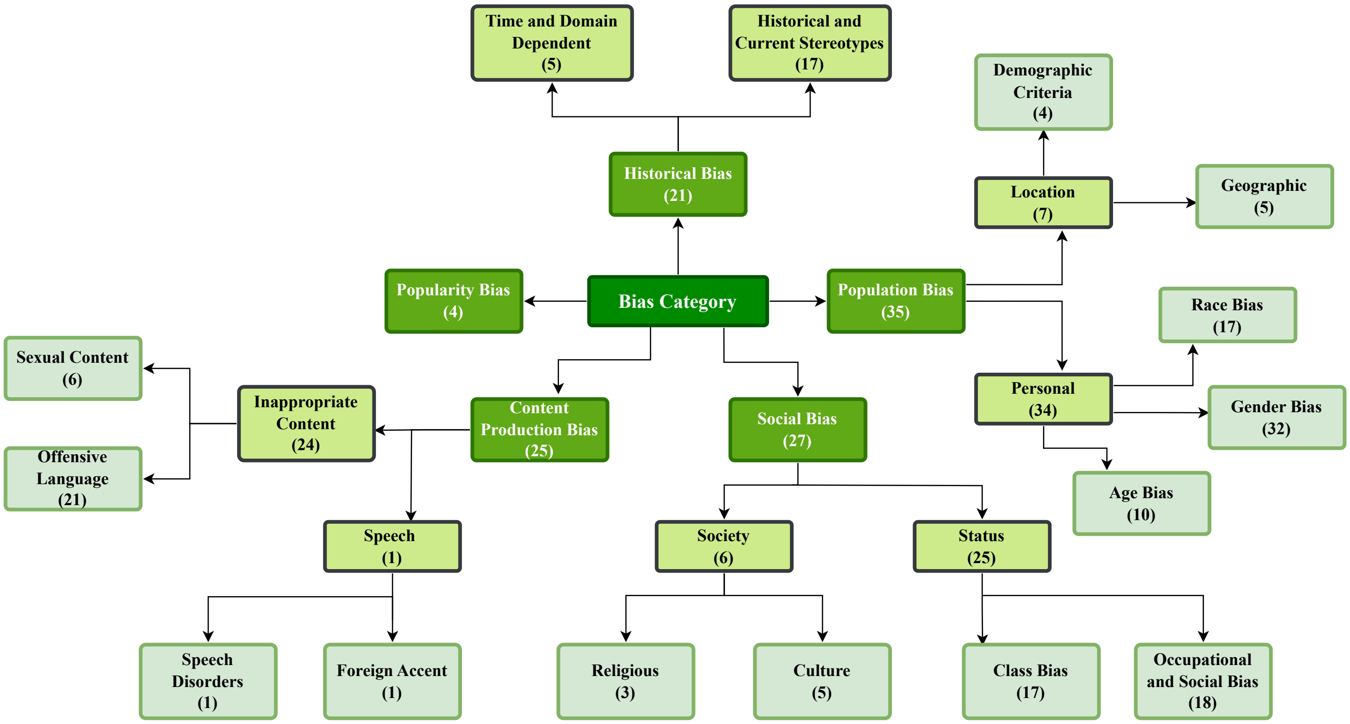

Figure 2: Bias categories with the # of projects from the sample (leaves do not sum up as models can declare multiple biases)

Table 5: Models with Bias by Tasks

Model Task	# Models	No Bias	Documented Bias
NLP	249	194	55
RL	62	62	0
Audio	32	25	7
Multimodal	25	18	7
CV	16	13	3
Other	4	4	0
Tabular	1	1	0
Total	389	317	72

bias, and 43 exhibited more than one type of bias. 55 of these models are related to NLP, 7 to Audio, 7 to Multimodal tasks, and 3 to CV.

Models belonging to Other, RL, and Tabular do not declare any type of bias. In the Other and Tabular categories, the number of models is still limited, which makes it challenging to speculate on their bias declaration. However, after a careful evaluation of 62 models in the RL category, no evidence of bias was found in any of them. The reasons behind that are related to how such models are trained, as also discussed in Section 3.1, yet, as existing literature discusses, the underlying training dataset may not be representative of the phenomenon captured by the RL model [44], or the algorithms' approximation function introduces a bias [20].

The open card sorting on the 389 models of our sample resulted in a categorization depicted in Figure 2, and featuring three abstraction levels. The first level features five categories from the Mehrabi *et al.* taxonomy [34] and, specifically: (i) Popularity bias, a sub-category of their *Algorithm to User* category, and (ii) four sub-categories of their *User to Data* category. For the remaining sub-categories, we found no mapping for Self-Selection (as said, it mainly applies to survey studies), Behavioral (*i.e.,* no user behavior captured by the model resulted in a documented bias), and Temporal (for the latter we found descriptions more appropriate to the Historical Bias sub-category). Overall, the categorization features 17 leaf categories, including Popularity bias that does not have further subdivisions. Also, some categories (Content Production, Social, and Population) have an intermediate level of grouping.

In the following, we detail each level of categorization starting from the first level of Mehrabi *et al.* going through the intermediate levels down to the leaves.

Population Bias (35 models) is further specialized into two sub-categories: *Personal* and *Location*. *Personal* groups 3 types: **Race**, **Gender** and **Age** biases. These refer to the prejudice or discrimination that occurs based on a person's race, gender, or age. An interesting example of these biases is found in the model card of *openai/whisper-medium* (Audio) where it is stated that the model *"... exhibit[s] ... higher word error rate across speakers of different* ***genders, races, ages****, or other demographic criteria."* This is because the model is trained *"in a weakly supervised manner using large-scale noisy data."* We found 13 co-occurrences of **gender** and **race** bias, *e.g.,* related to models predicting black women work as in *distilbert-base-uncased.*

The *Location* sub-category includes 2 types of biases: **Demographic Criteria** and **Geographic**. It refers to models that could be skewed by the geographic distribution of the data used for training. **Demographic Criteria** bias affects models not representing the population diversity or could contain disproportions in demographic groups. **Geographic** bias emerges when some geographic areas are missing/limited in the training set. An example is the model *distilbert-base-uncased-finetuned-sst-2-english*. It states that *for sentences like "This film was filmed in COUNTRY", ... [the] binary classification model will give radically different probabilities for the positive label depending on the country (0.89 if the country is France, but 0.08 if the country is Afghanistan.)"*

Social Bias (27 models) is specialized into two intermediate levels: *Society* and *Status*. *Society*, in turns groups **Culture** and **Religious** biases. These biases can lead to discriminatory predictions stemming from the lack of sufficient representation of cultural or religious attributes. For example, the *stabilityai/stable-diffusion-2* (Multimodal) model card mentions *"While the capabilities of image generation models are impressive, they can also reinforce or exacerbate social biases."* Unsurprisingly, Culture bias occurs in 50% of the models exposing Demographic Criteria bias, due to the clear interconnection between the two elements.

The *Status* sub-category encompasses **Occupational and Social** bias, and **Class** bias. They occur when most of the existing occupations, social groups, or socioeconomic classes are not fully included in the training set or the latter contains harmful stereotypes. An example of **Occupational and Social** bias can be found in *distilroberta-base* model: *"Predictions generated by the model may include disturbing and harmful stereotypes across protected classes; identity characteristics; and sensitive, social, and occupational groups."*, and they show a concrete example of such an unfair prediction. They ask the model to complete the phrase "`The man/woman worked as a <mask>`". and the model returns, among options, also some current prejudices, *i.e.,* the man worked as `mechanic/courier` and the woman worked as `nurse/maid`.

Content Production Bias (25 models) is specialized into *Speech* and *Inappropriate Content*. *Speech* category includes 2 types of biases: **Speech Disorders** and **Foreign Accent**. These are related to the lack of specific speech characteristics, such as their accent, dialect, or speech disorders, thus models do not include information about accent variations, specific speech patterns, or any speech disorder. This category of bias mainly affects models related to the Audio task. For example, *TalTechNLP/voxlingua107-epaca-tdnn* faces speech bias. Its model card states that "*[b]ased on subjective experiments, it doesn't work well on speech with a foreign accent*" and, probably, also *"on persons with speech disorders."*"

Inappropriate Content relates to **Sexual Content** and **Offensive Content**. For example, the *google/flan-t5-xl* model states that it *"... can potentially be used for language generation in a harmful way"* since it *"... is fine-tuned on a large corpus of text data that was not filtered for explicit content or assessed for existing biases."*

The **Historical Bias** category (21 models) has two subcategories: **Time and Domain Dependent** and **Historical and Current Stereotypes**. They refer to time-sensitive elements and historical patterns in the data used for training. An example of *Time and Domain Dependent* was found in in *dslim/bert-base-NER*, that is used for Named Entity Recognition (NER) *"[the] model is limited by its training dataset of entity-annotated news articles from a specific span of time*".

Table 6: The top 20 licenses used by HF models

License	Permissiveness	#Models using it
Apache-2.0	Permissive	24,123
MIT	Permissive	10,151
CreativeML-openRAIL-m [42]	Permissive	3,805
OpenRAIL [42]	Permissive	2,529
CC-BY-4.0	Permissive	1,776
AFL-3.0	Permissive	1,559
Other	–	1,340
None/Unknown	–	722
cc-by-nc-sa-4.0	Restrictive	556
cc-by-sa-4.0	Restrictive	556
cc-by-nc-4.0	Restrictive	490
cc0-1.0	Permissive	482
GPL-3.0	Restrictive	431
Artistic-2.0	Permissive	347
Bigscience-bloom-rail-1.0 [4]	Permissive	208
BSD-3-clause	Permissive	197
Bigscience-openrail-m [11]	Permissive	163
wtfpl	Permissive	142
AGPL-3.0	Network Restrictive	104
Unlicense	Permissive	94

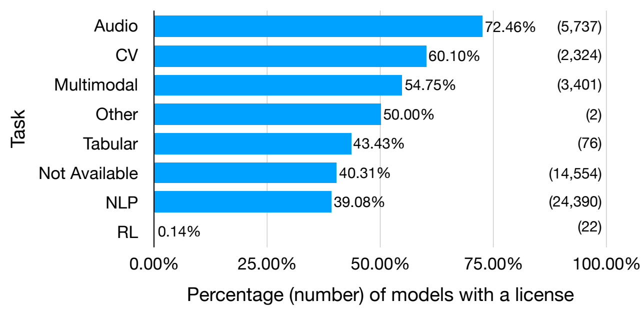

Figure 3: The distribution of model licenses by tasks.

Finally, **Popularity** bias includes only four models that explicitly describe generic bias due to the prevalence of some instances in the dataset. In one model *facebook/opt-125m*, only such a bias is declared *"Like other large language models for which the diversity (or lack thereof) of training data induces downstream impact on the quality of our model, OPT-175B has limitations in terms of bias and safety."*

> **Answer to RQ$_2$.** Of 389 manually analyzed models, only 72 (18%) describe their possible bias, categorized along four subcategories of *User to Data* from the Mehrabi *et al.* [34] taxonomy. These are in turn, specialized into 16 low-level categories.

3.3 RQ$_3$: Licenses

Among 159,132 analyzed models, only 50, 506 (32%) specify a license. In total, the models use 61 different licenses. Models are categorized in terms of permissiveness of their licenses as follows: 45, 991 (91%) use a *Permissive* license, 72 (0.1%) a *Weakly Restrictive*, 2, 276 (4.5%) a

Table 7: Models' licenses (rows) vs. client projects' (columns) licenses.

Models \ Projects	Network Restrictive	Not Available	Other	Permissive	Restrictive	Weak Restrictive
Network Restrictive	1	0	0	0	0	0
Not Available	42	16,576	615	23,700	257	6
Other	1	311	4	471	8	0
Permissive	164	33,818	1,153	40,288	732	13
Restrictive	0	570	18	**707**	15	0

Restrictive, and 105 (0.2%) a *Network Restrictive* one. The remaining 4.2% declare to use "other" licenses.

Table 6 reports the top 20 licenses specified by models, classified according to their permissiveness (the full list is available in our replication package). Note that the licenses reported in Table 6 cover over 98% of projects. A detailed description for most of the licenses can be found on *opensource.org* [2] or *choosealicense.com* [1], while we provide references for ML-specific licenses.

Unsurprisingly, permissive licenses are predominant (the most common one being Apache-2.0), as they facilitate the redistribution of work based on such models. The adoption of permissive licenses is, indeed, a trend previously shown also for open-source projects in general [49]. The most adopted restrictive licenses are cc-by-sa and cc-by-nc. The former constrains redistribution under the same license, while the latter forbids commercial exploitation. Also, among the top 20 licenses, we can also notice 431 models with a GPL-3.0 license and 105 with a Network Restrictive license (AGPL). The latter is likely adopted as such models may be reused to create network-based ML-intensive systems.

It is noteworthy that certain models are made available under ML-specific licenses, such as the CreativeML-openRAIL-m and OpenRAIL licenses, where the former simply adds the "Creative" labeling. These licenses rank as the third and fourth most popular choices, respectively. Other specific licenses would include the RAIL licenses from BigScience [11]: Bigscience-bloom-rail and the Bigscience-openrail-m licenses. The Open Responsible AI Licenses (OpenRAIL) [42] have been developed with the goal of promoting the reuse and redistribution of AI models, while forbidding harmful and unethical usages, either due to intrinsic technical limitations of the models or to specific use cases foreseen by the users. The Bigscience-bloom-rail license is specific for the BigScience Large Open-science Open-access Multilingual Language Model (BLOOM) models [4].

Figure 3 reports the extent to which licenses are declared for models belonging to different tasks. As shown, models related to Audio and CV are more documented in terms of licenses, possibly because of the multimedia content used to train them.

Then, we examine the relationship and association between licenses of HF models and the 17,365 traced GitHub client projects.

We compute a contingency matrix aiming at relating client projects' licenses and used models' licenses. An abstracted version of such a matrix (licenses are clustered in terms of permissiveness) is reported in Table 7, where columns represent the projects' licenses, and row the models' licenses. The detailed full matrix is in our replication package [41]. In general, a potential incompatibility can occur when a model with a restrictive license is reused in a project that is released under a different license, *i.e.,* more permissive. Results reveal that 707 models with a Restrictive license are reused in projects distributed under a Permissive license.

Table 8: Cases of licensing incompatibilities

Models \ Projects	Apache-2.0	BSD-2-clause	BSD-3-clause	MIT
cc-by-sa-4.0	350	1	4	99
GPL-3.0	233	0	1	19

Although models are dynamically downloaded through the HF API, this still makes a derivative work, and, therefore, a possible violation of the licenses. Also, it has to be said that violations may be subject to different interpretations and jurisdictions [50].

Next, we focus on specific intersections (*i.e.,* considering the actual licenses) where potential incompatibilities might arise. Results are reported in Table 8. As the table shows, incompatibilities mainly occur between projects released under the Apache 2.0 license and models that are under the cc-by-sa as well as under the GPL 3.0 license, yet there are also cases involving projects under the MIT, and very few under the BSD.

For example, the *RSDO-DS3/SloSemanticShiftDetection* project has an Apache 2.0 license, yet it uses the *EMBEDDIA/sloberta* model, which has been released with the cc-by-sa-4.0 license. On a similar note, yet with different pairs of licenses, the *salesforce/CodeRL* is a RL library distributed under the BSD-3 license, and it uses the *clip-italian/clip-italian* model under the GPL-3.0 license.

One peculiar case could be what happens when one creates a fork of the HF *transformers* library, which uses several models (including ones with a restrictive license) and releases it. For example, *ssss1029/transformers_custom* is released under the Apache-2.0 license, yet it uses the *jcblaise/electra-tagalog-small-uncased-generator* model which is released under the GPL-3.0.

Answer to RQ$_3$: 31% of the HF models declare a license, most of which (91%) a permissive one, and also employing ML-specific licenses, constraining a "responsible" usage of models. Over 4.7% of the models use a (network) restrictive license, and we found 707 GitHub projects released under a permissive license using such models.

4 IMPLICATIONS

The results of our study lead towards implications for *Data Scientists*, *Software Developers*, *Software Engineering Researchers*, and *Educators*.

Data Scientists, thanks also to initiatives like HF, are gaining awareness about model sharing and transparency of such models. However, there is still a long road towards having full transparency ensured by a large majority of models. To this extent, tools for automatically assessing the content and quality of model cards (similar to tools checking the quality of bug reports [9, 55]) would be desirable. Such tools may nicely complement tools such as DocML [3], helping developers in crafting new model cards, as well as tools such as AIMMX [48] that automatically extract metadata from ML models. Data Scientists should also carefully test the used models against bias, and, whenever possible, use techniques like the ones existing in literature to mitigate such bias. As it is further discussed in Section 6.3, this includes techniques based on re-ranking [37], rebalancing distorted predictions [40], random sampling [13], knowledge graph manipulation [33], or neural-network partitioning and pruning [5].

Developers need better support in selecting models based on their transparency (*e.g.,* through the use of "transparency badges" on model hubs). Additionally, they should increase their awareness about the significance of transparency in models. Last, but not least, while the importance of licensing is gaining maturity for conventional software, it is still not the case for licensing, given the number of potential incompatibilities we found. This also triggers the need for licensing check tools accounting for ML models' licenses.

Software Engineering Researchers can exploit transparency info for several purposes, (i) understanding how models are reused by projects and whether countermeasures are taken, (ii) leveraging the model card contents to test ML software against bias and to mitigate it, and (iii) enhancing the practice and support for AIBOM, *e.g.,* by encompassing bias info, but, also, developing tools for automated AIBOM generation.

Educators should not only teach about ML bias (as it is already being done), but, also, challenges related to model's transparency, explaining how to properly document newly created models. Also, with the growth of ML-intensive systems, it is becoming even more important to instruct future developers about legal implications arising when integrating existing, pre-trained ML models into their software systems.

5 THREATS TO VALIDITY

Threats to *construct validity* concern the relationship between theory and observation. This threat may be due to imprecision in our analyses. As already explained in Section 2.1, the identification of links between projects and models can suffer from imprecision due to the over-simplified analysis of the `from_pretrained` invocations. This threat does not affect RQ_1 and RQ_2, yet it may affect RQ_3, where we identify licensing compatibility between projects and models. Truly, the performed analysis allowed us to gather a relatively large set of 17,365 projects, which could be sufficiently representative, yet we cannot exclude it can suffer from a bias.

As for RQ_1 and RQ_2, we performed a manual analysis of a sample of models. To ensure the reliability of our classifications and coding, we computed inter-rater reliability measures. For RQ_1, the manual analysis of the sample is complemented by an exhaustive analysis on all 159,132, yet limited to datasets declared through tags.

Finally, for RQ_3, besides the threat of the project-model tracing discussed above, we leverage the licenses declared in the model and GitHub projects' metadata, and we cannot exclude that these are largely incomplete or even incorrect. As explained in Section 2.1, we mitigated this threat through the manual analysis on a sample of projects. As for the models, this is the way licenses should be declared when releasing them on HF [23]. As for the projects, the actual license is contained in the projects' `LICENSE*` file, which might be wrongly classified by GitHub. Nevertheless, this file is usually created starting from the GitHub-provided templates, which mitigates the threat.

Threats to *internal validity* concern factors, internal to our study, that may influence our findings. Our study does not make causation claims, if not using downloads as a proxy of models' usage when sampling to address RQ_1 and RQ_2. The sample for RQ_1 and RQ_2 is intentionally not random, as we wanted to focus on models largely downloaded, *i.e.,* possibly used.

Threats to *conclusion validity* concern the relationship between observation and outcome. The analysis of RQ_1 and RQ_2 may not be exhaustive, as it is only based on a sample of the models.

Threats to *external validity* concern the generalizability of our findings. Although HF is a very popular hub for pre-trained transformer models, it only represents a partial view of the reality, and other hubs such as Model Zoo [31], Tensorflow Hub [19], or PyTorch Hub [47]. Moreover, we focus on transformers, but other studies may investigate other types of pre-trained models. Last, but not least, we look at the models' usage by GitHub projects, which, again, only represent a partial view of the overall models' usage.

6 RELATED WORK

Considering the scope of our work, we focus on three distinct aspects: (i) works mining data from HF; (ii) documentation of ML models, (iii) bias and fairness in software engineering, and (iv) licensing analysis in open-source.

6.1 Empirical Studies on Hugging Face

Jiang *et al.* [26] interviewed 12 experts from HF to gain insights into the current practices and challenges associated with reusing pre-trained models. They identified the key attributes of reused models, focusing on provenance, reproducibility, and portability. The study revealed that the primary challenges in this domain revolve around inconsistencies between actual and reported performances, as well as concerns regarding model risks. Our paper is complementary to Jiang *et al.* [26], first in terms of purpose (studying models' transparency and level of documentation), and second in terms of methodology (mining study instead of interviews).

Castano *et al.* [7] conducted a study to analyze the carbon footprint of 1,417 models hosted on HF. They evaluated the carbon dioxide emission during the training phase and found correlations between emissions and factors such as model size, dataset size, and application domains. The results of their study serve to foster a "sustainable" ML-based development. Also in this case, our study tackles complementary challenges, focusing on models' datasets, biases, and licensing compatibility. Nevertheless, we believe that carbon footprint in another, essential component of a model documentation.

6.2 Documentation of Machine Learning Models

Several papers have dealt with ML models' documentation. The idea of model cards, *i.e.,*, of structured documentation of ML models, was first advocated by Mitchell *et al.* [35] as a way to foster the transparency and democratization of ML.

Bhat *et al.* [3] analyzed the content of 132 model descriptions taken from HF, GitHub, and industrial ones to determine how the structure of a model card should look like and implemented a tool named DocML to guide the creation and maintenance of model cards. While we share with the work of Bhat *et al.* the importance of having certain sections in the model cards, our work has substantial differences with respect to their work. First, we perform a deep analysis of (i) the way datasets are shared, (ii) the types of bias that may affect a model, and (iii) licensing declaration and compliance. Second, our analysis is extensive and concerns a sample of 159,132 models.

Tsay *et al.* [48] proposed a tool named AIMMX to aid the automated extraction of ML model metadata from different types of software repositories, such as GitHub or ArXiV repositories. We believe the need for tools like AIMMX or DocML is further motivated by our study which shows the limited exposition of relevant pieces of information in models' descriptions and potential problems that can be caused by model bias or licensing compliance.

Crisan *et al.* [12] conducted a design inquiry with experts in ML and natural language processing to investigate the usefulness of interactive model cards, showing that such interaction helps stakeholders in model understandability and interpretability, but, also, it increases the level of trust in such models. Our study goes in a different direction as we do not focus on the level of interaction a model card permits but, rather, on its content.

6.3 Bias and Fairness in Software Engineering

Coping with bias and fairness of ML algorithms goes beyond software engineering, and several approaches have been proposed in the literature. Such approaches leverage different techniques, including: (i) optimization techniques like CPFair, that perform a post-hoc re-ranking of recommendations by accounting for fairness requirements; [37], (ii) random sampling to duplicate/remove elements until the bias mitigation goal is reached, [13], or knowledge graph manipulation to account for interaction bias [33].

Beyond that, some authors have focused, more in detail, on biases concerning software development. Brun and Meliou [6] studied the potential impact of biased data in software engineering phases, *i.e.,* requirement specification, system design, testing, and verification. They found that a comprehensive classification of software biases remains an open challenge. Spoletini *et al.* [46] proposed bias-aware guidelines on how to cope with bias in software engineering tasks.

Given the increasing prominence of pre-trained language models and their extensive applications, particularly in text generation, researchers have studied their biases. Schramowski *et al.* [43] discovered that pre-trained models possess knowledge of deontological choices, moral norms, and values. In response, they propose a novel approach called "MoralDirection" to retrieve human-like biases regarding what is considered right and wrong when employing pre-trained language models for text generation.

Fairness emerges as a crucial concept in software engineering, aiming to achieve unbiased and equitable outcomes for the individuals affected by software systems. To this aim, researchers have proposed various approaches. For instance, pioneering work by Chakraborty *et al.* [8] emphasized the importance of developing fair models, laying the foundation for subsequent studies in this domain. Gohar *et al.* [18] examine the impact of hyperparameters on fairness in machine learning models. Through an empirical study of 168 ensemble models from popular fairness datasets, they explore the composition of fairness and its interaction with different properties. Their results reveal that fair outcomes can be achieved in ensembles without specific mitigation techniques, and they identify the connections between fairness composition and data characteristics. Biswas *et al.* [5] propose Fairify, an SMT-based approach for analyzing individual fairness properties in neural network (NN) models. They address the challenge of verifying individual equity in NNs due to non-linear calculations by utilizing input partitioning and NN pruning to provide fairness certification or counterexamples.

Monjezi *et al.* [36] introduce DICE, a testing and debugging framework for identifying localized fairness defects in deep feedforward neural networks (DNNs). Their approach assists software developers in triaging fairness defects based on severity and quantifying fairness using protected information in decision-making.

Hoag *et al.* [22] focus on integrating demonstrably safe and fair ML systems into specific applications such as medical applications and autonomous vehicles. Peng *et al.* [40] address discrimination issues in ML algorithms through their proposed model xFAIR. xFAIR employs method extrapolation to both mitigate bias and explain its causes. It re-balances distorted predictions of the classification model using distributions of protected attributes.

Nguyen *et al.* [38] have studied the extent to which recommender systems for software engineers, and, in particular, API recommenders, could suffer from a bias, related to recommending popular libraries as opposed to very specific, less-used ones.

As a side note, approaches to cope with the fairness of ML models go beyond software engineering applications.

The aforementioned pieces of work relate to ours because, once developers have—thanks to models' transparency—a clear idea about the possible biases affecting a model they are using, proper countermeasures to mitigate such a bias can be taken.

6.4 Studies on Open Source Licenses

Several authors studied the adoption and usage of open-source licenses, looking at the changes in licensing and licensing-related factors leading towards the success or failure of open-source projects [14], by performing large-scale studies on open-source projects [49], or by surveying developers [51]. Such studies, in general, suggested a general trend towards more permissive licenses, that facilitated the exploitation, even in industrial contexts, of open-source products. This trend is also confirmed by our analysis, conducted on ML models, for which specific licenses are also conceived and adopted.

Germán and Hassan [17] identified integration patterns when creating derivative work under different licenses. These may arise also for ML-specific projects, as far as models are concerned. Kapitsaki *et al.* propose recommenders for licenses satisfying projects' constraints and dependencies [27, 30] as well as SPDX-specific compatibility checks [28, 29]. Other work empirically assessed the licensing compatibility in Linux distributions like Fedora [16], or on large sets of open-source projects [52]. Vendome *et al.* [50] analyzed developers' discussions to understand what circumstances, related to licenses, lead to "licensing bugs" and how these can be subject to different interpretations and jurisdictions. Similarly, we analyzed compatibility between open-source projects and ML models hosted on HF Hub.

A recent thread of research concerns Software Bills of Materials (SBOMs) [21, 53, 54] and the need for software projects to expose SBOMs as inventory of their content, as also established by Governmental regulations in several countries, *e.g.,* in the US [39]. In particular, Xia *et al.* [53] performed an interview-based study with 17 practitioners and distilled 25 statements on SBOM practices. In one question of their study, participants pointed out how SBOMs for AI software are different from conventional SBOMs, as they also carry out information about models, training, etc. In the context of AI-specific SBOM formats are being defined (AIBOM). In principle, the presence of detailed information on model cards such as those of HF can be used to generate AIBOMs. Therefore, our study investigates the extent to which models document the necessary information to generate AIBOMs.

7 CONCLUSION AND FUTURE WORK

This paper studied the transparency of pre-trained transformer models hosted on the Hugging Face (HF) hub, in terms of training datasets, prediction bias, and declared licenses. We analyzed a total of 159,132 models created for different tasks.

Results of the study indicate that (i) only 14% of the models declare datasets through specific tags, and, by manually analyzing a statistically significant sample of 389 top-downloaded models, we found that 61% of them document the training datasets in some ways. However, only 18% of the analyzed models declare a bias, which is mostly related to the *User to Data* category of Mehrabi *et al.* [34] taxonomy, which we further detailed in sub-categories. For what concerns the licenses, we found evidence of a declared license in 31% of the models, with a prevalence toward permissive licenses, and, among other, AI-specific licenses such as the OpenRAIL, that impose a "responsible" model reuse. Nevertheless, there are still several models using restrictive licenses, and this generates multiple (we found 707) cases of incompatibilities in client projects.

In future work, we aim to analyze further dimensions of the models' transparency, including training parameters, declared performances, and updates/fixing over time. We also plan to complement the qualitative analysis we performed on datasets and bias by automating the identification of bias and model declaration within models' textual descriptions. Further research will also be conducted toward better and automatically generated AIBOMs.

8 DATA AVAILABILITY

The mined dataset, results of the qualitative analysis, and analysis scripts are available in our replication package [41].

ACKNOWLEDGMENTS

Massimiliano Di Penta acknowledges the Italian PRIN 2020 Project EMELIOT "Engineered MachinE Learning-intensive IoT system", ID 2020W3A5FY. Federica Pepe is partially funded by the PNRR DM 352/2022 Italian Grant for Ph.D. scholarships. Antonio Mastropaolo and Gabriele Bavota acknowledge the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 851720)

REFERENCES

[1] 2023. Choose an open source license https://choosealicense.com.
[2] 2023. Open Source Initiative https://opensource.org.
[3] Avinash Bhat, Austin Coursey, Grace Hu, Sixian Li, Nadia Nahar, Shurui Zhou, Christian Kästner, and Jin L. C. 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, CHI 2023, Hamburg, Germany, April 23-28, 2023.* ACM, 749:1–749:17.
[4] BigScience. 2022. BigScience Large Open-science Open-access Multilingual Language Model https://huggingface.co/bigscience/bloom.
[5] Sumon Biswas and Hridesh Rajan. 2022. Fairify: Fairness Verification of Neural Networks. *arXiv preprint arXiv:2212.06140* (2022).
[6] Yuriy Brun and Alexandra Meliou. 2018. Software Fairness. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Lake Buena Vista, FL, USA) *(ESEC/FSE 2018).* Association for Computing Machinery, New York, NY, USA, 754–759. https://doi.org/10.1145/3236024.3264838
[7] Joel Castaño, Silverio Martínez-Fernández, Xavier Franch, and Justus Bogner. 2023. Exploring the Carbon Footprint of Hugging Face's ML Models: A Repository Mining Study. In *Proceedings of the ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), Mon 23 - Fri 27 October 2023 New Orleans, Louisiana, United States.*
[8] Joymallya Chakraborty, Suvodeep Majumder, Zhe Yu, and Tim Menzies. 2020. Fairway: a way to build fair ML software. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering.* 654–665.
[9] Oscar Chaparro, Carlos Bernal-Cárdenas, Jing Lu, Kevin Moran, Andrian Marcus, Massimiliano Di Penta, Denys Poshyvanyk, and Vincent Ng. 2019. Assessing the quality of the steps to reproduce in bug reports. In *Proceedings of the ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/SIGSOFT FSE 2019, Tallinn, Estonia, August 26-30, 2019.* ACM, 86–96.
[10] J Cohen. 1960. A coefficient of agreement for nominal scales. *Educ Psychol Meas.* (1960).
[11] Danish Contractor and Carlos Muñoz Ferrandis. 2022. BigScience Large Open-science Open-access Multilingual Language Model https://bigscience.huggingface.co/blog/the-bigscience-rail-license.
[12] Anamaria Crisan, Margaret Drouhard, Jesse Vig, and Nazneen Rajani. 2022. Interactive Model Cards: A Human-Centered Approach to Model Documentation. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency* (Seoul, Republic of Korea) *(FAccT '22).* Association for Computing Machinery, New York, NY, USA, 427–439. https://doi.org/10.1145/3531146.3533108
[13] Giordano d'Aloisio, Andrea D'Angelo, Antinisca Di Marco, and Giovanni Stilo. 2023. Debiaser for Multiple Variables to enhance fairness in classification tasks. *Information Processing & Management* 60, 2 (2023), 103226. https://doi.org/10.1016/j.ipm.2022.103226
[14] Massimiliano Di Penta, Daniel M. Germán, Yann-Gaël Guéhéneuc, and Giuliano Antoniol. 2010. An exploratory study of the evolution of software licensing. In *ICSE (1).* ACM, 145–154.
[15] Santiago Dueñas, Valerio Cosentino, Gregorio Robles, and Jesús M. González-Barahona. 2018. Perceval: software project data at your will. In *Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings, ICSE 2018, Gothenburg, Sweden, May 27 - June 03, 2018.* ACM, 1–4.
[16] Daniel M. Germán, Massimiliano Di Penta, and Julius Davies. 2010. Understanding and Auditing the Licensing of Open Source Software Distributions. In *The 18th IEEE International Conference on Program Comprehension, ICPC 2010, Braga, Minho, Portugal, June 30-July 2, 2010.* IEEE Computer Society, 84–93.
[17] Daniel M. Germán and Ahmed E. Hassan. 2009. License integration patterns: Addressing license mismatches in component-based development. In *31st International Conference on Software Engineering, ICSE 2009, May 16-24, 2009, Vancouver, Canada, Proceedings.* IEEE, 188–198.
[18] Usman Gohar, Sumon Biswas, and Hridesh Rajan. 2022. Towards understanding fairness and its composition in ensemble machine learning. *arXiv preprint arXiv:2212.04593* (2022).
[19] Google Inc. 2023. TensorFlow Hub https://www.tensorflow.org/hub.

[20] Qiang He and Xinwen Hou. 2020. WD3: Taming the Estimation Bias in Deep Reinforcement Learning. In *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*. 391–398.

[21] Stephen Hendrick. 2022. Software Bill of Materials (SBOM) and Cybersecurity Readiness. https://tinyurl.com/293v3xte.

[22] A. Hoag, J. E. Kostas, B. da Silva, P. S. Thomas, and Y. Brun. 2023. Seldonian Toolkit: Building Software with Safe and Fair Machine Learning. In *2023 IEEE/ACM 45th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. IEEE Computer Society, Los Alamitos, CA, USA, 107–111.

[23] Hugging Face. 2023. Hugging Face - Licenses https://huggingface.co/docs/hub/repositories-licenses.

[24] Hugging Face. 2023. Hugging Face Hub API Endopoints https://huggingface.co/docs/hub/api.

[25] Hugging Face Inc. 2023. Hugging Face https://huggingface.co. https://huggingface.co

[26] Wenxin Jiang, Nicholas Synovic, Matt Hyatt, Taylor R Schorlemmer, Rohan Sethi, Yung-Hsiang Lu, George K Thiruvathukal, and James C Davis. 2023. An empirical study of pre-trained model reuse in the hugging face deep learning model registry. *arXiv preprint arXiv:2303.02552* (2023).

[27] Georgia M. Kapitsaki and Georgia Charalambous. 2021. Modeling and Recommending Open Source Licenses with findOSSLicense. *IEEE Trans. Software Eng.* 47, 5 (2021), 919–935.

[28] Georgia M. Kapitsaki and Frederik Kramer. 2015. Open Source License Violation Check for SPDX Files. In *Software Reuse for Dynamic Systems in the Cloud and Beyond - 14th International Conference on Software Reuse, ICSR 2015, Miami, FL, USA, January 4-6, 2015. Proceeding (Lecture Notes in Computer Science, Vol. 8919)*. Springer, 90–105.

[29] Georgia M. Kapitsaki, Frederik Kramer, and Nikolaos D. Tselikas. 2017. Automating the license compatibility process in open source software with SPDX. *J. Syst. Softw.* 131 (2017), 386–401.

[30] Georgia M. Kapitsaki, Athina C. Paphitou, and Achilleas Achilleos. 2022. Towards open source software licenses compatibility check. In *Proceedings of the 26th Pan-Hellenic Conference on Informatics, PCI 2022, Athens, Greece, November 25-27, 2022*. ACM, 96–101.

[31] Jing Yu Koh. 2023. Model Zoo: Discover open source deep learning code and pretrained models https://modelzoo.co.

[32] Klaus Krippendorff. 2011. Computing Krippendorff's alpha-reliability. (2011).

[33] Cheng-Te Li, Cheng Hsu, and Yang Zhang. 2022. FairSR: Fairness-aware Sequential Recommendation through Multi-Task Learning with Preference Graph Embeddings. *ACM Transactions on Intelligent Systems and Technology* 13, 1 (Feb. 2022), 16:1–16:21. https://doi.org/10.1145/3495163

[34] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2022. A Survey on Bias and Fairness in Machine Learning. *ACM Comput. Surv.* 54, 6 (2022), 115:1–115:35.

[35] 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* (Atlanta, GA, USA) *(FAT* '19)*. Association for Computing Machinery, New York, NY, USA, 220–229. https://doi.org/10.1145/3287560.3287596

[36] Verya Monjezi, Ashutosh Trivedi, Gang Tan, and Saeid Tizpaz-Niari. 2023. Information-Theoretic Testing and Debugging of Fairness Defects in Deep Neural Networks. In *Proceedings of the 45th International Conference on Software Engineering* (Melbourne, Victoria, Australia) *(ICSE '23)*. IEEE Press, 1571–1582.

[37] Mohammadmehdi Naghiaei, Hossein A. Rahmani, and Yashar Deldjoo. 2022. CPFair: Personalized Consumer and Producer Fairness Re-ranking for Recommender Systems. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Madrid Spain, 770–779. https://doi.org/10.1145/3477495.3531959

[38] Phuong T. Nguyen, Riccardo Rubei, Juri Di Rocco, Claudio Di Sipio, Davide Di Ruscio, and Massimiliano Di Penta. 2023. Dealing with Popularity Bias in Recommender Systems for Third-party Libraries: How far Are We?. In *20th IEEE/ACM International Conference on Mining Software Repositories, MSR 2023, Melbourne, Australia, May 15-16, 2023*. 12–24.

[39] NIST. 2021. Improving the Nation's Cybersecurity: NIST's Responsibilities Under the May 2021 Executive Order. https://www.nist.gov/itl/executive-order-14028-improving-nations-cybersecurity

[40] Kewen Peng, Joymallya Chakraborty, and Tim Menzies. 2022. FairMask: Better Fairness via Model-Based Rebalancing of Protected Attributes. *IEEE Trans. Softw. Eng.* 49, 4 (nov 2022), 2426–2439.

[41] Federica Pepe, Vittoria Nardone, Antonio Mastropaolo, Gerardo Canfora, Gabriele Bavota, and Massimiliano Di Penta. 2023. *Dataset of the paper: "How do Hugging Face Models Document Datasets, Bias, and Licenses? An Empirical Study"*. https://doi.org/10.5281/zenodo.8200098

[42] Responsible AI. 2022. Big Science Open Rail-M License https://www.licenses.ai/blog/2022/8/26/bigscience-open-rail-m-license.

[43] Patrick Schramowski, Cigdem Turan, Nico Andersen, Constantin A Rothkopf, and Kristian Kersting. 2022. Large pre-trained language models contain human-like biases of what is right and wrong to do. *Nature Machine Intelligence* 4, 3 (2022), 258–268.

[44] Benjamin Smith, Anahita Khojandi, and Rama Vasudevan. 2023. Bias in Reinforcement Learning: A Review in Healthcare Applications. *ACM Comput. Surv.* (jul 2023). https://doi.org/10.1145/3609502 Just Accepted.

[45] Donna Spencer. 2009. *Card sorting: Designing usable categories*. Rosenfeld Media.

[46] Paola Spoletini and Reza M. Parizi. 2018. Bias-aware guidelines and fairness-preserving Taxonomy in software engineering education. In *2018 IEEE Frontiers in Education Conference (FIE)*. 1–4. ISSN: 2377-634X.

[47] The Linux Foundation. 2023. PyTorch Hub https://pytorch.org/hub.

[48] Jason Tsay, Alan Braz, Martin Hirzel, Avraham Shinnar, and Todd W. Mummert. 2020. AIMMX: Artificial Intelligence Model Metadata Extractor. In *MSR '20: 17th International Conference on Mining Software Repositories, Seoul, Republic of Korea, 29-30 June, 2020*. ACM, 81–92.

[49] Christopher Vendome, Gabriele Bavota, Massimiliano Di Penta, Mario Linares Vásquez, Daniel M. Germán, and Denys Poshyvanyk. 2017. License usage and changes: a large-scale study on GitHub. *Empir. Softw. Eng.* 22, 3 (2017), 1537–1577.

[50] Christopher Vendome, Daniel M. Germán, Massimiliano Di Penta, Gabriele Bavota, Mario Linares Vásquez, and Denys Poshyvanyk. [n. d.]. To distribute or not to distribute?: why licensing bugs matter. In *Proceedings of the 40th International Conference on Software Engineering, ICSE 2018, Gothenburg, Sweden, May 27 - June 03, 2018*.

[51] Christopher Vendome, Mario Linares Vásquez, Gabriele Bavota, Massimiliano Di Penta, Daniel M. Germán, and Denys Poshyvanyk. 2015. When and why developers adopt and change software licenses. In *2015 IEEE International Conference on Software Maintenance and Evolution, ICSME 2015, Bremen, Germany, September 29 - October 1, 2015*. IEEE Computer Society, 31–40.

[52] Yuhao Wu, Yuki Manabe, Tetsuya Kanda, Daniel M. Germán, and Katsuro Inoue. 2017. Analysis of license inconsistency in large collections of open source projects. *Empir. Softw. Eng.* 22, 3 (2017), 1194–1222.

[53] Boming Xia, Tingting Bi, Zhenchang Xing, Qinghua Lu, and Liming Zhu. 2023. An Empirical Study on Software Bill of Materials: Where We Stand and the Road Ahead. In *45th IEEE/ACM International Conference on Software Engineering, ICSE 2023, Melbourne, Australia, May 14-20, 2023*. IEEE, 2630–2642.

[54] Nusrat Zahan, Elizabeth Lin, Mahzabin Tamanna, William Enck, and Laurie Williams. 2023. Software Bills of Materials Are Required. Are We There Yet? *IEEE Security & Privacy* 21, 2 (2023), 82–88.

[55] Thomas Zimmermann, Rahul Premraj, Nicolas Bettenburg, Sascha Just, Adrian Schröter, and Cathrin Weiss. 2010. What Makes a Good Bug Report? *IEEE Trans. Software Eng.* 36, 5 (2010), 618–643.