MACHINE LEARNING MODEL STORES: A CONSTRUCT OF SOFTWARE ENGINEERING FOR ARTIFICIAL INTELLIGENCE

by

ERNESTO LANG OREAMUNO

A thesis submitted to the School of Computing
in conformity with the requirements for the
Degree of Master of Science

Queen’s University
Kingston, Ontario, Canada
December 2022

Copyright © Ernesto Lang Oreamuno, 2022
Abstract

Machine learning as a discipline continues to evolve, today developers, business and other entities are seeking to build applications from machine learning models and datasets, giving rise to a new discipline of SE4AI. As a parallel development, the development of machine learning model stores has become a prominent trend.

Machine learning model stores like Hugging Face integrate a series of processes that allow for the reuse of datasets and models. First, Hugging Face provides standards based on academic research in order to facilitate the documentation of models and datasets. Second, they provide tools to train and deploy datasets and models more easily. Third, they provide infrastructure such as git, to maintain these artifacts. These elements are provided with the objective of facilitating the development and usage of machine learning artifacts so that they may be more easily integrated into production environments.

However, while these developments show promise, there is little research on how
they operate. In terms of documentation, there are standards which have been used by Hugging Face, but they have not been empirically validated yet. In the same way, there is research that talks about the common problems present in the machine learning development lifecycle, but empirical validation of such issues. Additionally, while issues relating to the evolution of software development problems have been researched before, these have largely remained unexplored within machine learning. In this research we address two problems related to the latter. First, we analyze the documentation practices present throughout the models and datasets in Hugging Face, to determine how closely these resemble the practices established by previous standards. We also look into how dependencies between datasets exist in Hugging Face and how these are documented by the model store. Second, we performed a thorough statistical analysis of the commit history data pertaining to model, dataset, and spaces in order to determine the effort necessary to maintain and run a model store. Our findings by provide useful information to developers on what they can expect to find in terms of documentation in models stores and how much effort is needed to maintain these artifacts.
First, I would like to thank Dr. Catherine Stinson, my supervisor. I am grateful for her guidance, help and for allowing me to take part in her lab. I have greatly enjoyed my time here, due in no small part to how great everyone at the lab is. Second, I would like to thank Dr. Bram Adams for all his help with ICSE and getting this project up and running, his support was invaluable to getting this thesis written. I would like to give special thanks to my parents, both of which have been supportive in every facet of my journey to achieve my goals. Finally, I would also like to acknowledge my friends, it would have not been possible to do all of this work without all those weekends we spent out and about around Kingston and elsewhere. As a final note, this has been an awesome experience, thank you all, and we’ll be seeing each other.
One of the chapters of this thesis is based on published research. The research idea, data collection, research method, analysis and writing of the manuscript for that paper is based on such research. In addition to that, it must be clarified that support was provided by various co-contributors in order to make that publication possible. This work is presented below:

1. Ernesto Lang Oreamuno, Rohan Faiyaz Khan, Catherine Stinson, Bram Adams
   On The Practices For Documenting ML Models and Datasets. ACM International Conference of Software Engineering (ICSE). Under Review. This work is described in Chapter 3.
# Contents

Abstract i

Acknowledgements iii

Co-authorship iv

List of Tables vii

List of Figures viii

1 Introduction 1

1.1 Thesis Statement .................................................. 3
1.2 Thesis Overview .................................................. 3
1.3 Thesis Contribution ............................................... 7

2 Background 8

2.1 Machine Learning, Datasets, Models and Model Stores ............ 9
2.2 The Dataset and Model Card Standards ........................... 10
2.3 Quality of Software Engineering Documentation ................... 11
2.4 Maintainability Metrics, Commit Histories and Software Evolution Over Time .................................................. 14
2.5 Software Engineering for Artificial Intelligence (SE4AI) .......... 15


3.1 Introduction ....................................................... 18
3.2 Methodology ....................................................... 20
3.3 Model Card Analysis Results (RQ1) ................................ 30
3.4 Dataset Card Analysis Results (RQ2) ................................ 35
3.5 Dataset Dependency Results (RQ3) ................................ 38
3.6 Implications ....................................................... 44
3.7 Threats to Validity ............................................... 46
List of Tables

<table>
<thead>
<tr>
<th></th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Statistics on Models with(out) cards</td>
<td>31</td>
</tr>
<tr>
<td>3.2</td>
<td>Statistics on Datasets with(out) cards</td>
<td>35</td>
</tr>
<tr>
<td>3.3</td>
<td>Similarity Score Statistical Measures</td>
<td>40</td>
</tr>
<tr>
<td>4.1</td>
<td>Statistical Measures for Datasets</td>
<td>66</td>
</tr>
<tr>
<td>4.2</td>
<td>Kolmogorov-Smirnov Distances for Dataset Distributions</td>
<td>68</td>
</tr>
<tr>
<td>4.3</td>
<td>Statistical Measures for Models</td>
<td>70</td>
</tr>
<tr>
<td>4.4</td>
<td>Kolmogorov-Smirnov Distances for Model Distributions</td>
<td>72</td>
</tr>
<tr>
<td>4.5</td>
<td>Mann-Whitney U Test Results</td>
<td>74</td>
</tr>
<tr>
<td>4.6</td>
<td>White Test Statistics for Spaces</td>
<td>79</td>
</tr>
<tr>
<td>4.7</td>
<td>Statistical Measures for Models</td>
<td>84</td>
</tr>
<tr>
<td>4.8</td>
<td>Kolmogorov-Smirnov Distances for Space Distributions</td>
<td>86</td>
</tr>
</tbody>
</table>
List of Figures

2.1 Model Card Example ........................................... 12
2.2 Dataset Card Example ......................................... 13
3.1 RQ1 Steps ....................................................... 21
3.2 RQ2 Steps ....................................................... 24
3.3 RQ3 Steps ....................................................... 26
3.4 Threshold Box Plots ........................................... 29
3.5 Model Card Section Title Prevalence ....................... 32
3.6 Model Card Hybrid Card Sorting ............................ 34
3.7 Dataset Card Section Completeness (all datasets with cards) 36
3.8 Dataset Card Deductive Coding (deductive coding) ........ 39
3.9 Similarity Score Distributions .............................. 41
3.10 Core-Periphery Structure of 214 merged dataset nodes ...... 42
4.1 RQ1 Steps ....................................................... 55
4.2 RQ2 Steps ....................................................... 57
4.3 RQ3 Steps ....................................................... 60
4.4 Dominant Topics for Datasets ............................... 62
4.5 Terms Of Use for Datasets .................................. 63
4.6 Dominant Topics for Models ................................. 65
4.7 Dataset Commit Data Histogram ............................ 67
4.8 Exponential Distribution Fitting for Models ................. 69
4.9 Model Commit Data Histograms ............................ 71
4.10 Exponential Distribution Fitting for Models ............... 73
4.11 Dataset Risk Distribution .................................. 75
4.12 Model Risk Distribution .................................... 76
4.13 Dominant Topics for Models ............................... 78
4.14 Time Series for Spaces (1) ................................. 80
4.15 Time Series for Spaces (2) ................................ 81
4.16 ACF Plot for Spaces (1) ................................... 82
4.17 ACF Plot for Spaces (2) ................................... 83
4.18 Spaces Commit Data Histograms ........................... 85
4.19 Exponential Distribution Fitting for Spaces
A machine learning (ML) model store [Xiu et al., 2020] is a venue that provides access to pre-trained models and preprocessed datasets that can be easily deployed in several contexts. Hugging Face 1, AWS Marketplace 2 and Wolfram Neural Net Repository 3 are notable examples of these stores.

Currently, these stores have begun to develop into strong ecosystems that are used for business purposes. For instance, since its launch in 2016, Hugging Face has grown significantly, featuring 70,870 models and 9,798 datasets as of September 17th, 2022. The store allows data scientists and software engineers to upload their models and datasets to make them publicly available, while also offering capabilities for versioning.

---

1https://huggingface.co/
2https://aws.amazon.com/marketplace
3https://resources.wolframcloud.com/NeuralNetRepository/
and updating these artifacts. Hugging Face itself was originally valued at $80 million USD \(^4\). Later on, it attracted the attention of companies such as Amazon \(^5\) and Google \(^6\). In essence, the store itself has become important as a hub for machine learning models and datasets.

Furthermore, these stores also play a role in the developing field of software engineering for artificial intelligence (SE4AI), as they often combine various paradigms of software engineering such as documentation standards \([\text{Mitchell et al., 2019, McMillan-Major et al., 2021}]\), deployment procedures \([\text{Paleyes et al., 2020}]\) and strategic practices \([\text{Nahar et al., 2022}]\) for the development of models and datasets with the various groups involved in those areas. In general the models and datasets provided by these stores make it easier for developers to integrate such artifacts into software solutions, given that these elements usually require expert knowledge and skills to properly use them.

However, many of these developments into ML model stores are currently unexplored. In terms of documentation, there is little to no verification of how models and datasets are documented. There is little information on the standards that are used to provide instructions for the use of these artifacts within production environments. Moreover, there is also a lack of research regarding the management of these stores as it concerns the development efforts to keep model and dataset repositories up to date as well as how these elements have evolved over time.

The purpose of this thesis is to provide an in-depth exploration of Hugging Face

\(^4\)https://www.forbes.com/sites/kenrickcai/2022/05/09/the-2-billion-emoji-hugging-face-wants-to-be-launchpad-for-a-machine-learning-revolution

\(^5\)https://huggingface.co/amazon

\(^6\)https://huggingface.co/google
as a model store with focus on two areas. First, this thesis talks about the documenta-
tion practices for models and datasets within Hugging Face with a special focus
on determining what sort of information is present with relation to the standards it
has adopted for documentation practices [Mitchell et al., 2019, McMillan-Major et al.,
2021]. Second, we pursue a statistical study of how the model and dataset reposi-
tories have evolved, in addition to another places of repository *spaces*, which serve as
applications that encapsulate both datasets and models with the goal of providing
visual previews of these elements working together. The goal of these two approaches
is to provide a notion of what is necessary to maintain a model store, something that
we wish to clarify in order to give insights to other businesses and developers on what
is necessary to maintain their own SE4AI ecosystems.

1.1 Thesis Statement

Machine learning model stores are new developments that introduce concepts
such as the reusability of models and datasets for production contexts. It is nec-
essary to understand how these stores have managed to document and maintain
their models, datasets and other artifacts. This is done specifically to provide
clarity on what is necessary to create a machine learning ecosystem in the context
of software engineering for artificial intelligence.

1.2 Thesis Overview

This section provides an overview of this thesis.
1.2.1 Chapter 2: Background

In this chapter, background information is provided with relation to machine learning and ML model stores, software engineering documentation practices, documentation standards for datasets and models, software engineering for artificial intelligence and mining software repositories.

1.2.2 Chapter 3: Documentation Practices of Third-Party Models and Datasets: A Mixed-Methods Empirical Study

Documentation practices for within the domain of ML model stores as observed through Hugging Face are largely dependent on previously defined standards such as the dataset card standard [McMillan-Major et al., 2021] and the model card standard [Mitchell et al., 2019]. Given that documentation has been described as one of ML’s biggest challenges [Fischer et al., 2021], it is therefore necessary to understand how these practices are followed.

In this chapter, a mixed methods empirical study is presented to describe the documentation practices that are present within Hugging Face’s models and datasets. First, this study presents a statistical overview of the number of documented models, we also delve into the topics that can be found within the text of the model cards by using techniques such as manual analysis using hybrid card sorting [Bacchelli and Bird, 2013, Beyer and Pinzger, 2014] which presents a categorization of the model cards with respect to the standards. Second, a statistical overview of the number of documented datasets is presented, as well as a manual analysis using deductive coding
that details the prevalence of matching information with relation to the standards for the dataset cards. Third, this research provides a look into how dependencies between datasets are documented through the use of digital forensics techniques and visualizations of complex graph structures.

The findings within this research show that most models (60.38%) and datasets (71.52%) are not documented. Moreover, we find that in terms of dependencies between datasets there is a high degree of similarity (50.6 out of 100), in accordance to what is specified by the procedures of the digital forensics tools that were used. These results validate much of the previous research that has addressed various problems that exist within machine learning ecosystems [Paleyes et al., 2020, Nahar et al., 2022, Sculley et al., 2015].

1.2.3 Chapter 4: The Evolution Of Machine Learning Model Stores: A Statistical Analysis

Machine learning models and datasets consist of code artifacts in the same way that software projects do. That is to say that, much in the same way as a software project uses version control history, so do the artifacts that are commonly used within machine learning. In the case of Hugging Face, this is readily apparent through their use of git versioning systems and infrastructure which is utilized for their projects involving machine learning models, datasets, and spaces (visual applications that combine models and datasets to provide previews of how these would work together). Perhaps, more importantly, is that the management of these artifacts throughout time has an extensive effort requirement in the way of code commits (additional code that

\footnote{http://roussev.net/sdhash/tutorial/03-quick.html}
has been, added, deleted or modified in a code file belonging to an artifact), which provides insights into how much effort was put into maintaining a model, dataset or space.

In this chapter, we explore the commit histories of models, datasets and spaces. We discover the topics present in the commit messages of 7,877 datasets, 37,887 models and 6,330 spaces (accounting for more than 300 GB of data) with the usage of topic modelling with LDA [Blei et al., 2003]. Furthermore, this chapter also delves into the applications of metrics such as the Delta Maintainability Metric [di Biase et al., 2019] to the commit histories of the models, datasets and spaces, as well as the analysis of the time series data that can be retrieved from the sum of all the artifacts extracted through this study’s massive data extraction procedures. Finally, an analysis with python code smell detectors is performed on the code files of spaces, specifically to understand how complex these small applications given that they offer a preview of how an SE4AI application might look like.

Some of the findings of this research show that when it comes to datasets the three most relevant topics are Training Files with 58.57% topic dominance, Initial Commits and Updates with 21.57% topic dominance and Licensing with 20.16% topic dominance. In terms of time series information, it has been determined that the amount of commits and the frequency with which they are made is heteroskedastic in nature, demonstrating how there is an element of uncertainty that comes with machine learning projects [Vogelsang and Borg, 2019].
1.3 Thesis Contribution

This thesis is focused on aiding entities that may seek to create their own ML ecosystems as well as entities that might be seeking to deploy models and datasets into production contexts. These contributions can be summarized as follows:

1. This work is the first to focus on providing statistical data to validate many of the common claims that exist with relation to machine learning issues. Furthermore, it is also focused in providing clarity on the documentation practices that exist in ML, a subject that has not been researched extensively.

2. This work analyzes the effort necessary to build an ML ecosystem by taking into account commit histories of the models, datasets and spaces (small visual applications that combine software engineering practices, models and datasets) present within Hugging Face. This is done to demonstrate the effort necessary to maintain an ML ecosystem, another subject that has not been researched extensively.
In this chapter we provide an overview of the different fields that this research touches upon. Section 2.1 refers to the basics of Machine Learning, datasets, models and model stores. Section 2.2 refers to the dataset and model card standards, which are the standards used to document models and datasets at Hugging Face. Section 2.3 refers to previous research that identify issues that exist within software engineering, in terms of documentation of software projects. Many of these issues are somewhat similar to those that can be observed in an SE4AI context. Section 2.4 refers to the usage of maintainability metrics and information related to commit histories, which are common approaches towards measuring the maintainability of a software project. Section 2.5 refers to software engineering for artificial intelligence (SE4AI), a new area within software and machine learning, which encapsulates ideas from both disciplines.
2.1 Machine Learning, Datasets, Models and Model Stores

Datasets and models represent the major pieces of the typical machine learning lifecycle. In essence, datasets are the inputs and models are the outputs [Amershi et al., 2019]. The first stages of this process involve the selection of a dataset, which itself often involves a complicated data collection and engineering process followed by data filtering and data cleaning. Whereas when it comes to models, data scientists and data engineers are tasked with creating training scripts and infrastructure able to use a dataset, train and experiment with multiple models until an appropriate level of performance is reached. Similar to dataset procedures, model training involves many steps, such as splitting the data into training, testing and validation splits as well as the application of metrics such as accuracy, precision, recall and F1 scores to evaluate model performance [Kelleher et al., 2020]

These steps require significant domain expertise and resources (especially in terms of hardware, people and capital). Recently within industry, the trend has been moving towards replacing in-house models and datasets with 3rd-party pretrained models and preprocessed datasets. This has enabled an ease-of-use that did not exist beforehand, it is now mostly about locating a model or dataset within a model store like Hugging Face, fine-tuning it, integrating and testing it within the context of an end user application. If this process is successful, the application is then deployed into production where the predictions that are made by an integrated model can then be made available to regular users.

Currently, there are two specific concerns with these elements. First, if a model
or dataset is to be used in this fashion, it is essential to know what exactly it is that these models or datasets provide. As a result, it is essential for organizations to clearly understand all necessary details about the datasets and models hosted by model stores. This information should be easily available from their documentation. Second, if these organizations have an interest in deploying these models and datasets, then it is also necessary for them to understand the steps that are needed to properly maintain them, as well as the effort that these processes may require.

### 2.2 The Dataset and Model Card Standards

The dataset and model card standards are recent specifications that Hugging Face has adopted for the purpose of model and dataset documentation. These standards are divided into a series of sections (and subsections for the dataset cards) that detail specific information such as *Model Details* and *Intended Use* in the case of model cards, *Dataset Description* and *Dataset Creation* in the case of dataset cards. While these standards are new, they are the result of published research [McMillan-Major et al., 2021, Mitchell et al., 2019] created specifically for the purposes of documenting the particularities that come with ML models and datasets.

In terms of models, Figure 2.1 shows how a model card is structures, with the model card section title and its section content being circled in blue and green respectively. To further elaborate, the model cards at Hugging Face consist of various sections such as "model description" with textual content that provides the actual documentation within each section. However, even though Hugging Face adopted the model card standard [Mitchell et al., 2019], it is only suggested as a way to document cards, and is not actively enforced. For instance, the model card standard mentions
a section title *Model Details*, yet the card in Figure 2.1 uses "model description" instead. In total, the model card standard contains 9 sections.

On the other hand, for datasets Figure 2.2 presents a screenshot of the dataset card. These cards are divided into sections, which are in turn divided into subsections. These subsections are circled in blue in Figure 2.2 with their content being circled in green. A section like *Dataset Description* is composed of multiple subsections (*Dataset Summary*, *Supported Tasks and Leaderboards* and *Languages*), each detailing a specific part of the *Dataset Description* section. In total, the dataset card specification [McMillan-Major et al., 2021] contains 5 sections and 19 subsections. Other example sections within the dataset cards include *Dataset Structure*, *Dataset Creation*, *Considerations For Using The Data* and *Additional Information*. In contrast to the model cards, Hugging Face actively enforces this standard, given that it provides a template to document this information.

### 2.3 Quality of Software Engineering Documentation

Before the AI era, various researchers have highlighted the issues that are present within the documentation of software engineering projects. A systematic mapping study [Zhi et al., 2015] found that most software documentation is centered around design, code, and generic aspects. Furthermore, recent research [Aghajani et al., 2019] focuses on the quality of software documentation, with most of the quality issues found being related (268 issues), up-to-dateness (190 issues) and usability (138 issues).
Regarding completeness, it has been found that it was difficult to automatically check the completeness of documentation, since this relies on human interpretation based on deep knowledge requirements of a project as well as the specific process and context of the development team [Briand, 2003]. In terms of up-to-dateness, it was discovered that most of the time software documentation is outdated and is for the most part written from the point of view of implementation rather than maintenance [Garousi et al., 2013]. As a result documentation was considered difficult to use whenever maintenance tasks were involved. Finally, in terms of usability there are different definitions for that subject. For instance in some cases it is believed that usability should be about the ability to easily navigate documentation while in others
it should be about how well it explains how to use a software application [Aghajani et al., 2020a]. Given that SE4AI (and by extension model stores) is a subfield of software engineering, it is likely that many of these previous issues identifies can presently be found within the documentation of models and datasets.

Currently, there is little research that deals with the quality of the documentation of machine learning models and datasets. For instance, there are claims (without empirical evidence) that datasets are sparsely documented [Nahar et al., 2022].
terms of the importance of documentation, it has been stated before that the doc-
umentation of datasets and models is one of the biggest challenges within machine
learning [Fischer et al., 2021]. Finally, as far as popular datasets are concerned, such
as BookCorpus, there is research that has found several inconsistencies with the way
this dataset is documented, with problems ranging from copyright issues, duplication,
unmarked biased to overrepresentation of particular elements [Bandy and Vincent,
2021].

2.4 Maintainability Metrics, Commit Histories and
Software Evolution Over Time

Software projects are constantly evolving and changing due to the nature of software
engineering. Software engineers must always adapt to the changing circumstances of
requirements, evolving technologies and stakeholder knowledge, which often dictate
how software projects evolve over time. As such a key part of software engineering
projects is maintainability, and the capacity to deal with it as time goes, after all it
has been said before that ”The total cost of maintaining a widely used program is
typically 40 percent or more of the cost of developing it.” [Brooks, 1987]. Furthermore,
the most successful projects in this regard, are those that are able to manage these
circumstances appropriately [Rajlich, 2014].

In order to deal with these ever-changing circumstances that are present within
software projects, a series of metrics have been established to gain insight into how
software evolves over time. These insights are then used to better assess how to add or
modify an existing software project in a way that such changes will fit properly. Some
of these metrics include cyclomatic complexity [McCabe, 1976] or more recently the delta maintainability model [di Biase et al., 2019], which quantify software projects through concepts such as lines of code per change, risk of a change to a software project and number of files affected by a change. More importantly, these metrics are can be used in conjunction with the commit histories of a software project to assess how its evolution has proceeded throughout time, and whether said process involved a high degree of volatility or not.

At this moment, there is little research regarding how models and datasets evolve over time, in particular within the scope of software engineering. Given that model stores such as Hugging Face utilize technologies such as git to keep a versioning history of their models and datasets and that these are often software projects that involve code, it is therefore necessary to explore how the store has managed to maintain such models, in particular because of the differing circumstances that exist between machine learning projects and regular software projects [Vogelsang and Borg, 2019].

2.5 Software Engineering for Artificial Intelligence (SE4AI)

The area of SE4AI is a relatively new discipline, both in terms of implementation and research. To further understand this area, this document encourages readers to refer elsewhere [Fernandez et al., 2021] to gain a comprehensive overview of this domain. In the context of this work 3 particular areas of research are discussed. First, there are detailed studies that talk about the collaboration that exists between data scientists, software engineers and data engineers regarding ML model development [Nahar et al.,
Second, there is research that deals with some of the more common issues that exist with models and datasets, in particular when it comes to dealing with implementing, documenting and maintaining them [Sculley et al., 2015]. Third, there is research that has looked into industry use cases that deal with the deployments of datasets and models, and the challenges that come with this [Paleyes et al., 2020].

Furthermore, there are also studies that have surveyed machine learning model stores in order to determine their capabilities, especially as they relate to pre-AI era mobile app stores [Xiu et al., 2020]. Most of this research was exploratory, and focused on assessing stores such as ModelDepot 1, AWS Market Place 2 and Wolfram Neural Net Repository 3. In the case of this thesis, the focus lies on identifying the key elements that are necessary to maintain a model store when it comes to documentation and maintenance effort of models and datasets. In addition to that, this thesis provides quantitative evidence to support the claims made by previous research regarding various issues surrounding models and datasets.

1 https://modeldepot.io/
2 https://aws.amazon.com/marketplace
3 https://resources.wolframcloud.com/NeuralNetRepository/
This chapter is under review for publication at the International Conferences for Software Engineering (ICSE).

Model stores provide access to third party ML models and datasets ready for deployment. Just as with mobile app and other software stores, entities seeking to deploy models and datasets should expect to find detailed specifications of these datasets and models in the documentation, leveraging newer documentation standards such as model and dataset cards. However, despite substantial research on documentation issues in other domains of software engineering, there are currently no insights on the quality of such documentation for machine learning. To address this problem, we use
statistical analysis, hybrid card sorting and deductive coding to assess the information present in the dataset and model cards of one of the largest model stores in use today (i.e., Hugging Face), to validate how well it follows the documentation standards. We also study the documentation of dependencies between data sets, based on digital forensics techniques and stochastic block modeling. We found that only 21,902 (39.62%) of models and 1,925 (28.48%) of datasets are documented. Finally, in terms of dependencies, only 20 (9%) of the studied datasets document their dependencies, despite evidence of higher similarity between datasets. Overall, our findings demonstrate an inconsistent documentation of datasets and models.

3.1 Introduction

The development and deployment of machine learning models and datasets is an arduous task that requires considerable expertise, specific domain knowledge and significant resources (in capital, manpower and infrastructure). These processes often involve cooperation between specialists such as software engineers, data scientists and data engineers [Nahar et al., 2022]. As such it is difficult for companies to develop machine learning solutions given the need for these complicated procedures. However, as an answer to these issues, the use of machine learning models stores for the purposes of selecting pretrained models and preprocessed datasets has become a trending development, due to the ease-of-use they provide. It is no longer a question of training models or gathering data, now the issue is becoming one of selecting the right dataset or model for a particular task.

In order for companies and developers to utilize these models and datasets they must have an accurate description of their functioning as well as their capabilities.
In an effort to provide that information, Hugging Face has adopted the dataset card [McMillan-Major et al., 2021] and the model card [Mitchell et al., 2019] standards. These provide guidelines to document critical information for models and datasets, for instance in the case of models the *Model Details* should provide an overview of a model with all the necessary information to understand its functioning. *Intended Use*, on the other hand, should detail what the model should be used for and where it does not apply. Moreover, on the dataset front, sections such as *Dataset Description* provide information about subjects such as the general overview of a dataset, the *Supported Tasks and Leaderboards* it is part of, and the *Languages* it provides support for.

The aim of this chapter (and the research that was elaborated here) is to provide clarity on the information that is present within the documentation of the datasets and models within Hugging Face. It is focused specifically on understanding how the dataset and model documentation matches that of the standard as well as determining how dataset dependencies are documented within Hugging Face. To address these issues, this research focuses on the following three research questions:

**RQ1:** What kind of information is documented for the models at Hugging Face?  
– In terms of deployment, machine learning models are the outcome of the machine learning pipeline [Paleyes et al., 2020]. It is therefore essential to understand the nuances of models from their documentation in order to provide an accurate assessment to entities seeking to deploy them in production. These assessments will in turn provide an accurate view of what is to be expected from Hugging Face when it comes to model documentation. As a preview of the results that were found, it was determined that a majority of 33,378 (60.38%) models are not documented.

**RQ2:** What kind of information is documented for the datasets at Hugging Face?
Datasets are the input of the machine learning pipeline, but more importantly, they dictate the success of a model [Paleyes et al., 2020]. Therefore, in similar fashion to models, understanding how datasets are documented in Hugging Face is crucial. Providing these assessments will give interested parties information about what they should expect to find when it comes to the available datasets in Hugging Face. Much as has been specified before (without empirical evidence however), publicly available datasets are poorly documented [Nahar et al., 2022]. In the case of Hugging Face, it was found that a majority of 4,833 (71.52%) have no documentation.

RQ3: How well are dependencies between datasets documented? – Similar to code clones, a ”bug” in a source dataset might require updates to any derived datasets (and consequently, all models trained on them), assuming that there are clear traceability links between the root and its derived datasets. This research question aims to understand the extent to which these derivation relations exist between Hugging Face datasets and how they are explicitly documented. This assessment provides much needed information on troublesome issues such as those of dataset dependencies, which have been pointed out as a serious issue by previous research [Sculley et al., 2015]. The initial findings of this research question show that there are only 20 (9%) datasets that make any reference towards other datasets within their documentation.

### 3.2 Methodology

This section provides the details related to the methodology that was used to carry out this research. Section 3.2.1 outlines the goal of this research and the way in which we pursued said goal while the rest of the sections present the approach taken towards each RQ.
3.2.1 Goal

We relied on the Goal/Question/Metric template [Basili and Rombach, 1988] to define our research goal as follows: we analyze documentation practices regarding datasets and models distributed by model stores, as well as the underlying relationships that exist between the datasets themselves. We do this for the purpose of understanding the kinds of information that are present in the documentation of the model cards and dataset cards, and to understand whether relations between datasets are being documented. Furthermore, we do this from the point of view of entities seeking to deploy models and datasets, and from the context of Hugging Face, which is our case study.

3.2.2 RQ1 Steps

![Diagram of RQ1 Steps]

Figure 3.1: RQ1 Steps
Figure 3.1 represents the steps that we took to carry out the model card analysis for RQ1.

**Step 1.1: Model Card Scraping** – First, we extracted all the model cards for every model present in Hugging Face. The total number of models at the time of analysis (July 4, 2022) was 55,280. This is our main source of data for our analyses. We had to rely on web scraping, because Hugging Face does not offer any APIs or data caches from which to extract this data.

**Step 1.2: Data Storage** – Second, we took the data for each model extracted from the model cards at Hugging Face and store it into a PostgresSQL database. The scripts divided the cards into sections and stored the sections, their titles, and the content associated to them.

**Step 1.3 Data Filtering** – Third, we filtered out the models that did not have any content, which left 21,902 models that we used to perform the next steps of our analysis.

**Step 1.4 Descriptive Statistical Analysis** – Fourth, we performed descriptive statistical analysis to determine the most common section titles of the model cards. We put special focus into comparing how closely the categories we identified through our manual analysis resemble the sections described in the model card standards that Hugging Face suggests as a guide to document models [Mitchell et al., 2019]. However, since the use of these standards is merely a suggestion by Hugging Face, our further analysis in Step 1.5 considers all titles regardless of whether they are mentioned in the standards.

**Step 1.5 Hybrid Card Sorting** – Sixth, we perform hybrid card sorting [Bacchelli and Bird, 2013, Beyer and Pinzger, 2014] to understand clearly what information is
being documented in the sections identified in the previous step. The process to perform this was as follows: for each analyzed model, its sections are analyzed and classified in the context of the standards [Mitchell et al., 2019] to see if the sections match with what is supposed to be documented. For instance, if a section title is called *Model Description*, step 1.4 would map this to *Model Details* in accordance to the standards, however if the content of the section does not accurately match the title or the correct category of the standards (for instance, the content is more appropriate for *Training Data*), then our hybrid card sorting would classify it as Training Data instead and label it as having mismatching content.

We chose a hybrid card sorting approach because as we mentioned earlier, Hugging Face does not enforce the standards, so while it may be possible to find content that matches those standards closely, it is also likely that there are other sections that do not belong to the standard, something that necessitated the creation of new categories.

This process was carried out by the first author and a second person on a random sample of 378 model cards with a confidence level of 95% and a confidence interval of 5%. The Cohen’s kappa agreement rate between the first author and the second author (an expert on image based ML models and ethical standards in ML) was 82% [McHugh, 2012]. Any disagreements that were found were thoroughly discussed until a consensus was reached to label a section.

### 3.2.3 RQ2 Steps

Figure 3.2 represents the steps we took to carry out the dataset card analysis for RQ2.

**Step 2.1 Dataset Card Scraping** – First, similar to what was done for model
cards, we used web scraping to extract all the dataset cards for every dataset present in Hugging Face. The total number of datasets retrieved at the time (July, 4, 2022) was 6,758 dataset cards.

**Step 2.2 Data Storage** When storing a dataset card, our scripts look at whether the dataset had a card; if it did, the card was broken down into sections, and then those sections were broken down into subsections.

**Step 2.3 Data Filtering** – Third, we filtered out the datasets that did not have any content, which left 1,925 cards with content.

**Step 2.4 Descriptive Statistical Analysis** – Fourth, we performed descriptive statistical analysis with reference to the section titles of the dataset cards, with reference to the standards established for documenting datasets in Hugging Face [McMillan-Major et al., 2021]. In contrast to the model cards, we focused explicitly on the
template provided by the standards, given that Hugging Face enforces it explicitly.

**Step 2.5 Deductive Coding** – Sixth, we performed deductive coding based on the sections and subsections proposed by the dataset card standards [McMillan-Major et al., 2021]. First, for a given subsection such as Dataset Summary, we check its content and verify that (1) it matches with what is supposed to be documented and that (2) the title and the content match properly. This analysis was carried out by two people on a random sample of 321 datasets with a 95% confidence level and a 5% confidence interval. The first author and the other person (an expert on image based neural networks and machine learning ethical standards) that carried out this analysis had a Cohen’s kappa agreement rate of 86% [McHugh, 2012] and when a disagreement was found both people discussed the differences and settled on a categorization that they both agreed on. This approach has been commonly used for qualitative analysis in software engineering [Bacchelli and Bird, 2013, Beyer and Pinzger, 2014].

### 3.2.4 RQ3 Steps

Figure 3.3 presents the sequence of steps taken to carry out our analysis for **RQ3**.

**Step 3.1 Dataset Downloading** – First, we took another random sample of 321 datasets (from the 1,925 datasets with cards) with a confidence level of 95% and a confidence interval of 5%. We then proceeded to download that sample of datasets along with all of their "configurations". "Configurations" of a dataset represent the different variants of the dataset uploaded by the dataset owner, for example subsets of the dataset with different sizes or characteristics. Since each of these variants contains one specific view of the dataset, this RQ requires taking a look at all variants. This provided us with 831 archives (1.8 TB in total).
Additionally, many of these datasets (and their configurations) consist of separate training, testing and validation splits, each representing a subset of the entire dataset. These files are provided by Hugging Face as arrow files. For our analysis, we opted to focus on training split files only for two reasons. First, according to the best practices of training machine learning models, the training split of a dataset should be statistically representative in order to avoid troublesome scenarios such as under-fitting and over-fitting [Kelleher et al., 2020]. Second, most models in Hugging Face provide a training split, but not all of them provide a testing and validation split. To obtain a more balanced comparison between datasets, we opted to only use training splits.

**Step 3.2 Similarity Calculation** – Second, we use a digital forensics algorithm called `sdhash` to analyze data similarity between the arrow files [Roussev, 2010].
Sdhash basically takes a file and generates a base-64 encoded string from its binary contents. Then, it compares this string with the strings generated from other files. After these comparisons are made, a score from 1-100 is outputted to denote the significance of the relationship between each pair of files.

We selected sdhash because of its robustness in handling large-scale terabyte data in digital forensics investigations, since in our analysis we had to deal with 1.8 TB of data. Furthermore, sdhash has been found to outperform its competitors [Rous-seev, 2010], and it works equally well on a range of different file types such as text documents, images, audio and video.

If we consider each configuration of every dataset downloaded in Step 3.1 as a node in a graph, then use the sdhash similarity score between every pair of nodes as a weight for the edge between those nodes, the output of Step 3.2 is a dataset similarity graph. However, given that this graph is complete (i.e., each pair of nodes is connected through an edge) and that a given dataset is represented by multiple nodes (i.e., all its configurations), we first need to filter the raw dataset similarity graph before doing any analysis. In particular, we need to (Step 3.3) filter out (configuration) nodes that have too low similarity with other nodes, then (Step 3.4) merge the remaining configuration nodes into one unified dataset node.

**Step 3.3 Filtering nodes based on similarity threshold** – In order to determine a threshold for the sdhash similarity values we first define $\mathcal{D} = \{d | d \in 321 \text{ studied data sets of Step 3.1}\}$, $\text{conf}(d)$ as the set of configurations of a given data set $d \in \mathcal{D}$, $\mathcal{N}_{\text{all}} = \{n | n \in \text{conf}(d) \land d \in \mathcal{D}\}$, $\mathcal{E}_{\text{all}} = \{\{m, n\} | m \in \mathcal{N}_{\text{all}} \land n \in \mathcal{N}_{\text{all}} \land m! = n\}$, $\text{edges}(n) = \{e | e \in \mathcal{E}_{\text{all}} \land n \in e\}$ and $\text{weight}(e)$ as the weight of an edge $e \in \mathcal{E}_{\text{all}}$. Note that the dataset similarity graph edges in $\mathcal{E}_{\text{all}}$ are
undirected, hence are represented as sets of 2 nodes. Then, \(\forall n \in \mathcal{N}_{\text{all}}\), we calculate the average similarity across all edges involving node \(n\):

\[
\text{sim}_{\text{ind}}(n) = \frac{\sum_{i=1}^{\text{edges}(n)}}{|\text{edges}(n)|} \text{weight}(\text{edges}(n)[i]) \tag{3.1}
\]

We then created a set of histograms to detect the underlying distributions of the sim_{ind} values to better understand how these scores are behaving. Moreover, we calculated the mean, median, standard deviation, kurtosis and skewness of these distributions to determine how outliers played a role within the datasets. In particular, kurtosis is a measure of whether a distribution is heavy-tailed or light-tailed concerning the normal distribution.

We obtained a mean \(\text{sim}_{\text{ind}}\) of 43.57 (median 40.30) with a standard deviation of 13.92, a skewness of 1.75 and kurtosis of 2.27. The positive skew of this data meant that most outlier nodes were found towards the right end of the distribution. Hence, based on the histogram we then decided to use the 75th percentile (or 3rd quartile) of 53.87 and set a limit to 60, as a similarity threshold to filter out configuration with overall similarity to other nodes.

To better elaborate on why 60 was chosen instead of 53, we selected groupings of nodes with different thresholds ranging from 50 to 70. Through these tests we see that the median jumps from 54.2 to 78.9. Since outliers are relevant towards this RQ we opted for a threshold of 60 which is the more conservative choice in this case. We also provide Figure 3.4, which shows the boxplots of \(\text{sim}_{\text{ind}}\) at these different thresholds.

**Step 3.4 Merging configuration with nodes** – Having filtered out dataset similarity graph nodes with weak \(\text{sim}_{\text{ind}}\) we now move on to merging all remaining configurations
Figure 3.4: Threshold Box Plots

of a given dataset into one dataset node. Basically, for each dataset \(d \in \mathcal{D}\), we first calculate the average \(\text{sim}_{\text{ind}}(n)\) across all \(n \in \text{conf}(d) \cap \mathcal{N}_{\text{filtered}}\) (i.e., the remaining configurations of \(d\)), then assign this value to \(\text{sim}_{\text{merge}}(d)\) value to a new graph node for \(d\). If no configuration node of \(d\) remains after step 3.3, no node is created for \(d\).

Then, \(\forall d' \in \mathcal{D}, d! = d'\) we add an edge \(\{d, d'\}\) with as weight the average weight across all possible remaining edges between configurations of \(d\) and \(d'\), i.e., \(\{n, m\} | n \in \text{conf}(d) \cap \mathcal{N}_{\text{filtered}} \land m \in \text{conf}(d') \cap \mathcal{N}_{\text{filtered}}\). Finally, we remove all individual configurations nodes \(n \in \text{conf}(d) \cap \mathcal{N}_{\text{filtered}}\), as well as all remaining edges \(\mathcal{E}_{\text{all}}\). We are left with a filtered dataset similarity graph with the datasets \(d \in \mathcal{D}\) as nodes, and a complete set of similarity edges between datasets. Since both the nodes and their edges have changed, we re-calculate \(\text{sim}_{\text{ind}}\) for each dataset node and update their \(\text{sim}_{\text{merge}}\) with these values.

**Step 3.5 Stochastic Block Model Visualization** – Fifth, we model and visualize
the filtered dataset similarity graph using stochastic block modelling to reveal the underlying structures [Holland et al., 1983]. This is needed, because the filtered graph is still a complete graph and just shows (average) similarity between each pair of datasets. This does not intuitively allow determining which dataset nodes are the “roots” from which other datasets are derived. Stochastic block modelling identifies communities within larger graphs based on edge (weight) density. We hypothesize that, if some datasets $R$ are more commonly roots from which a larger group of datasets $G$ is derived, $R$ and $G$ would show up as different communities in stochastic block modelling. We use graph-tool [Peixoto, 2014] to perform stochastic block modelling.

**Step 3.6 Dependency Discovery** – Sixth, we use the stochastic block modelling results to analyze how dataset dependencies obtained through derivation (or even cloning) are reflected within the documentation at Hugging Face. After the analysis of the stochastic block model visualization, we find that the datasets form a core-periphery network, which can be described as “a small number of hubs that gather a disproportionate amount of connections” [Hojman and Szeidl, 2008]. Armed with this new information we then proceed to check how these relationships might be documented in the corresponding dataset’s cards by looking for any mentions of the identified data dependencies.

### 3.3 Model Card Analysis Results (RQ1)

**60.38% of models in Hugging Face are not documented.** Table 3.1 displays the total number of models with cards and without cards showing that most models in Hugging Face are not documented at all. This is similar to what has been found
in previous research in "traditional" software engineering where the lack of documentation has been considered one of the major issues for software engineers [Aghajani et al., 2019]. While traditional SE documentation also tends to lack installation, deployment and release instructions [Aghajani et al., 2020b], their absence for ML models (which are more abstract than say, third party libraries) might even be a larger concern.

<table>
<thead>
<tr>
<th>Type</th>
<th>Number Of Models</th>
<th>Percentage(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Models With Cards</td>
<td>21,902</td>
<td>39.62</td>
</tr>
<tr>
<td>Models Without Cards</td>
<td>33,378</td>
<td>60.38</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>55,280</strong></td>
<td><strong>100</strong></td>
</tr>
</tbody>
</table>

Table 3.1: Statistics on Models with(out) cards

At the section title level, Hugging Face model cards mostly document the Training Data, Model Details and Intended Use sections. Figure 3.5 displays a bar chart with the most common occurrences of each section title throughout our sample of 378 out of the 21,902 documented models. This figure shows 13 categories, 6 of which are not part of the official model card standard but were identified by our manual analysis of the section titles. We highlighted in bold the section titles that are part of the standard.

This data suggests that the providers of models at Hugging Face are especially focused on the Model Details (654 occurrences; 32%), Training Data (482 occurrences; 23%) and Intended Use (398 occurrences; 19%) sections of the model card standards [Mitchell et al., 2019]. Additionally, in the next 4 most prevalent sections, three (Versioning, Domain Specific, and Citation Details) are not part of the official standard, but are very common in the analyzed model cards of Hugging Face, nonetheless.
Surprisingly, documentation involving categories such as *Metrics* appears to be less common with only 26 occurrences, representing a mere 1% of all section titles. Moreover, *Quantitative Analyses* and *Factors* do not appear to be documented at all in current model cards. Equally surprising is the lack of documentation concerning *Ethical Considerations* and *Caveats and Recommendations* (2 occurrences for both in total).

At the content level, *Model Details* contains the most content matches and the most content mismatches, with *Training Data* second in content matches, whereas *Intended Use*, *Training Data* and *Evaluation Data* have the most empty sections. Finally, *Ethical Considerations* and *Caveats and Recommendations*...
And Recommendations have almost no documentation whatsoever. Figure 3.6 presents a bar chart with the results of our hybrid card sorting on the 2,503 sections of the 378 analyzed model cards. Each bar represents a different category, with orange meaning the number of sections with no content, blue meaning the number of sections with content but having a mismatch with what the section should document according to the model card specification, and green meaning the number of correct content matches.

Apart from being the most documented (see Figure 3.5), Model Details, Training Data and Intended Use are also documented the best with 437 (21.2%) of content matches, Training Data having 337 (16.4%), and Intended Use having 162 (7.8%) of content matches. However, even though these sections are the most well documented, they also show inconsistencies, for instance, Model Details has the highest mismatched content percentage with 117 (5.7%) occurrences, Intended Use has the highest number of empty sections with 235 (11.4%) occurrences while Training Data lies somewhere in the middle with 116 (5.65%) empty sections and 19 (1%) mismatches. Surprisingly, Evaluation Data while being the 4th most common section title in Figure 3.5 has 115 (5.6%) empty sections. Across all the model card sections combined, 563 (23%) were empty, while 215 (9%) did not match.

In general, it seems that the providers of models and datasets at Hugging Face tend to give more importance to information pertaining to Model Details and to the Training Data while Evaluation Data, Metrics, Ethical Considerations and Caveats and Recommendations are practically nonexistent. The latter is problematic, especially in cases such as that of huggingtweets/mike_pence, a model created to generate tweets on the basis of the tweets of the former Vice President of the United States. By

1https://huggingface.co/huggingtweets/mike_pence
leaving out Ethical Considerations and Caveats and Recommendations, entities that might seek to use this model in a production context might be opening themselves to potential legal repercussions similar to those that Google DeepMind had to deal with when it deployed a model that did not properly address these concerns [Paleyes et al.,
Our results show that only 0.0009% of the analyzed sections of the models cards correspond to Ethical Considerations and Caveats And Recommendations.

Summary of RQ1: The majority of model cards are not documented (60.38%). In the case of the model cards that do contain documentation, 23% of the sections do not have content while 9% of the content does not match the information that should be specified according to the model card standards. Model cards are focused mostly on Model Details, Training Data, and Intended Use, with topics like Ethical Considerations and Caveats and Recommendations being sparsely (or even rarely) documented.

3.4 Dataset Card Analysis Results (RQ2)

The majority of dataset cards (71.52%) are not documented. Table 3.2 shows how documentation for datasets is even sparser than for models. Case in point, previous research that examines collaboration between software engineers, data scientists and data engineering teams mentions how publicly available datasets are difficult to work with because of their lack of documentation [Nahar et al., 2022]. Our results provide empirical evidence of the extent of this issue.

<table>
<thead>
<tr>
<th>Type</th>
<th>Number Of Datasets</th>
<th>Percentage(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datasets With Cards</td>
<td>1,925</td>
<td>28.48</td>
</tr>
<tr>
<td>Datasets Without Cards</td>
<td>4,833</td>
<td>71.52</td>
</tr>
<tr>
<td>Total</td>
<td>6,758</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 3.2: Statistics on Datasets with(out) cards
Figure 3.7: Dataset Card Section Completeness (all datasets with cards)

At the section title level, the standards used to document dataset cards are followed inconsistently. Figure 3.7 provides a bar chart with the section title-level percentages of fully documented content, partially documented content and lack of content for all 9,622 sections of the 1,925 documented dataset cards with respect to the dataset card standards [McMillan-Major et al., 2021].

Even though Hugging Face enforces the use of the dataset card standard, Dataset Description and the Dataset Structure sections are slightly more frequent than the other 3, presenting 1,933 (20.0%) and 1,763 (18.3%) respectively of all sections analyzed, compared to 1,640 (17.04%) for Dataset Creation, 1,611 (16.7%) for Considerations For Using The Data and 1,639 (17.03%) for Additional Information. This is
because Dataset Description and Dataset Structure are almost always included into the dataset cards, while the other sections are sometimes omitted entirely (despite being part of Hugging Face’s template), such as in the case of wongnai_reviews \(^2\) where the Considerations For Using The Data section was removed.

Among the five section titles, we noticed 3 different trends. Dataset Description and Additional Information both have a large percentage of partial content instances (13.3% and 10.7%) compared to full content (6.5% and 6.2%). The opposite trend is shown with Dataset Structure with 13.6% full content compared to 1.3% partial content. Finally, Dataset Creation and Considerations For Using The Data have most of their instances empty (9.9% and 11.4%).

The latter is surprising, since it is clear that Dataset Creation and Considerations For Using The Data are sections that are key to understanding the limitations of a dataset. The Dataset Creation section is used to understand who created and annotated the data, while the Considerations section specifies the potential ethical considerations involved with the data [McMillan-Major et al., 2021]. Similar to what was said before for model cards, the lack of this information could potentially lead to legal repercussions that could halt deployments of machine learning software systems.

At the subsection level, our deductive coding of the content of the 321 dataset cards shows that the more than half (11/19) of subsections have at least 45% empty content while (7/19) have at least 44% full content across all 321 cards.

Figure 3.8 presents a bar chart with the results of our deductive coding of 321 cards for the 19 subsections of the dataset card standard. In contrast to earlier plots, Similar to the section level results of Figure 3.7, especially the 6 Dataset Creation

\(^2\)https://huggingface.co/datasets/wongnai_reviews
subsections and the 3 Considerations, these have empty content in at least 55% of the cases. More unexpectedly, Supported Tasks (Dataset Description), Dataset Curators and Licensing (Additional Information) still have at least 43.6% percentage of empty content.

On the other hand, as expected, all subsections with more than 57.3% of content matches belong to the Dataset Structure, Dataset Description and Additional Information sections. Content mismatches are relatively rare, with a maximum percentage of 14 (4.3%) for the Initial Data Collection subsection.

(8/19) of subsections have at least 17% omissions. Contributions contains the highest level of omission with 98 (30.5%) subsections being omitted. This subsection followed by most of the subsections of the Dataset Creation, Considerations and Additional Information is notable, given the presence of many content matches as well. Related to the latter, even the Dataset Structure (Data Instances subsection) and, to some extent Dataset Description section (Supported Tasks) have a subsection that is often omitted.

Summary of RQ2: The majority of datasets (71.52%) are not documented. At the subsection level, 11/19 subsections are empty in almost half of the instances, while 7/19 have a complete match in almost half of the instances.

3.5 Dataset Dependency Results (RQ3)

The mean level of similarity between datasets is approximately 50.6 (out of 100), with 17 (8%) of the datasets having an average similarity of at least 75
with other datasets. Table 3.3 provides statistical measures such as mean, median, skewness, kurtosis and standard deviation based on the $\text{sim}_{\text{ind}}$ scores. Moreover, in Figure 3.9, a histogram displays the distributions of the similarity scores for the 214 datasets remaining after the merging of the nodes in Step 3.4 in Section 3.2.4.

It is readily apparent that the level of similarity is high according to what is specified by sdhash’s authors who consider anything above 21 (out of 100) high \(^3\). This is hinted at by the mean of 50.58 (compared to the common threshold of 21), with the

\(^3\)http://roussev.net/sdhash/tutorial/03-quick.html
data spread out substantially according to a standard deviation of 14.28. This distribution is affected by outliers meaning that there are datasets that contain extreme levels of similarity that are influencing the distribution as a whole. This is shown by the kurtosis of 2.27, indicating that the distribution is nearly leptokurtic, i.e., this distribution has a significant number of outliers causing it to be skewed [Aggarwal, 2016]. Since the skew is 1.14, then this makes the distribution positively skewed, meaning that the outliers present within this distribution lie rightwards of the mean.

In other words, there is a clear set of datasets with a disproportionate similarity score and a significant amount of connections within the graph, which is what is referred to as a core-periphery structure [Hojman and Szeidl, 2008].

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Mean</th>
<th>Median</th>
<th>Std</th>
<th>Skew</th>
<th>Kurt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>43.57</td>
<td>40.3</td>
<td>13.92</td>
<td>1.75</td>
<td>3.63</td>
</tr>
<tr>
<td>Merged</td>
<td>50.58</td>
<td>49.95</td>
<td>14.28</td>
<td>1.14</td>
<td>2.27</td>
</tr>
</tbody>
</table>

Table 3.3: Similarity Score Statistical Measures

28 datasets represent the roots (pink, gold and blue) of the 214 merged dataset nodes. Since similarity measures denote correlation, not causation [An et al., 2017], we use stochastic block modelling to obtain an approximate idea of which datasets form the roots from which other datasets are derived (copied, customized, etc.). Figure 3.10 displays the visualization of what the core-periphery structure that we identified beforehand looks like with respect to dataset similarity. Due to the complexity of the similarity graphs, even after merging the datasets with their configurations, we show the underlying core-periphery structure in Figure 3.10 without edges.

The visualization shows six different groups of nodes. Each color represents a
community, of nodes whose edge have comparable distributions of edge weights (i.e., similarity values). For example, the pink, gold, and blue communities in the center are the roots (or cores) of all the other datasets. These cores are the most influential datasets, i.e., the similarity scores of the edges that connect them to other nodes are higher with reference to other communities of nodes. On the other hand, teal, purple and orange nodes are peripheries, meaning that their level of similarity with respect
Figure 3.10: Core-Periphery Structure of 214 merged dataset nodes

to the edges that connect them to other nodes, is smaller, which is why they surround
the cores (edges were elided on the figure).

Out of the purple, orange, and teal peripheries 11 (out of 98) purple, 5
(out of 49) orange, and 2 (out of 40) teal datasets document dependencies on core datasets. In addition, 2 blue (out of 6) and 1 gold (out of 12) datasets also refer to another dataset.

While the exact number of datasets that missed documenting their root datasets is unknown, and would require tedious manual analysis of terabytes of data, the core periphery model seems to suggest that a non-trivial number of datasets on our dataset sample are related (highly similar). The lack of such documentation in most of the datasets studied in this RQ makes it impossible to be aware of and to propagate fixes and updates made to the root datasets. This confirms claims made by previous research, which described these data dependencies as "hidden technical debt" that are also more difficult to detect than regular software dependencies [Sculley et al., 2015].

As a first example, the GEM dataset \(^4\), is an important gold core dataset that is used as a benchmark for another 37 datasets \(^5\), it explicitly documents that e2e_nlg \(^6\) is one of the datasets that uses GEM. A pink core dataset in our sample uses it, and the edge between these two datasets in our filtered dataset similarity graph indeed has a similarity score of 85 marking them as strongly related, demonstrating a dependency between them.

A counter-example is isixhosa_ner_corpus \(^7\), a blue core dataset. Its documentation states that it depends on the conll benchmark \(^8\), a popular benchmark for language datasets. In the datasets that were collected, conll2000 \(^9\) and conll2002 \(^10\) are also

---

\(^4\)https://huggingface.co/datasets/gem
\(^5\)https://gem-benchmark.com/
\(^6\)https://huggingface.co/datasets/e2e_nlg
\(^7\)https://huggingface.co/datasets/isixhosa_ner_corpus
\(^8\)https://www.conll.org/2021
\(^9\)https://huggingface.co/datasets/conll2000
\(^10\)https://huggingface.co/datasets/conll2002
present within the pink core nodes. The edges of these two datasets have scores of 83 and 81.5 with ixishosa, showing that there is a dependency between them and that dataset. However, it is difficult to know just what is the nature of this relationship, and which variant was derived from which other dataset. More research is needed to understand how these elements come into play with relation to deployments of datasets.

Summary of RQ3: The high average $sdhash$ similarity of 50.58 in our filtered dataset similarity graph indicates an overall high level of dependencies between datasets, with a limited number of root datasets. Yet those dependencies are not well documented with only 11 (out of 98) purple, 5 (out of 49) orange, and 2 (out of 40) teal nodes of the three peripheries making references to such relationships (in addition to 3 core datasets).

3.6 Implications

3.6.1 Implications for Researchers

The main implication as far as research is concerned is related to how (lack of) documentation remains a major issue in the domain SE4AI. In fact, given that SE4AI involves collaboration between multiple additional roles (data engineer, data scientist, software engineer, etc.) the lack of documentation could be argued to have an even larger impact. As such, we believe that our research will open new avenues. Furthermore, our results suggest that the state-of-the-practice [Mitchell et al., 2019, McMillan-Major et al., 2021] used in Hugging Face does not seem to meet the
needs of users, are misunderstood or are simply not enforced enough. It is our hope that these findings will lead to more effective practices and tool support, such as recommender systems that suggest documentation practices [Aghajani et al., 2020a].

3.6.2 Implications for Deployment

In terms of deployment, the lack of incomplete/low-quality nature of documentation represents a substantial risk for models obtained from third-party model stores. First, the lack of documentation makes it difficult to truly understand what sort of problems will occur, when the models or datasets are put into production, given that as we saw before, the models and datasets are not consistent in their documentation, even with critical information such as dataset dependencies.

Second, in terms of legal repercussions, deploying third-party models in a business context might cause issues, especially since in many cases there is little to no information on the ethical issues that might arise. For instance, in cases like that of huggingtweets/mike_pence ¹¹ discussed in RQ1, there are recommendations that have been made directly to the United States government that have stated ”The USG should not use a bot to conduct actions that would be legally or ethically prohibited if done without the bot. Any promotion of false information will raise serious legal and ethical flags.” [Marcellino et al., 2020]. For models that do not outline any ethical considerations in their documentation, it is entirely possible that the legality of this model (and any product it is integrated in) would be questioned.

¹¹https://huggingface.co/huggingtweets/mike_pence
3.6.3 Implications for Tool Builders

Our results suggest that tools are essential to help software engineers, data engineers, and data scientists to enforce documentation standards, as well as to validate the quality of documentation. Apart from recommender systems or tools that might suggest or automate documentation for models and datasets (similar to what exists in software engineering). There is a strong need to create tools that trace the roots of datasets. This would in turn enable notifications to users of a given dataset about any major changes that might occur to the root dataset(s) from where it was derived from.

3.7 Threats to Validity

In this section we present the threats to the validity [Wohlin et al., 2012] of this empirical study.

Construct Validity: RQ1 uses hybrid card sorting, while RQ2 uses deductive coding. This is because Hugging Faces strongly suggests, but does not enforce, the model card standard [Mitchell et al., 2019], while it does enforce the dataset card standard and template. In terms of RQ3, we evaluated a range of digital forensics tools for data similarity, but decided on using sdhash because of its reliability and more recent versions in comparison to the other tools. We opted to use only the train split arrow files for RQ3 as opposed to using the train, validation and testing files, because not every dataset contains testing and validation files to be used for the sdhash analysis of RQ3. Nonetheless, given that within machine learning training splits should represent datasets in a statistically significant manner, we assume that
this is the case for the datasets at Hugging Face.

**Internal Validity:** In terms of internal validity, we performed manual analysis procedures on RQ1 and RQ2. In order to avoid issues with bias, two people were involved in the coding process. Both performed the manual classifications, then discussed the disagreements until a consensus was reached. This usually meant that both people either agreed on an existing category or decided to create a new one altogether when it came to a data point that could not be categorized. The agreement rates were of 79% and 91% for the models and datasets respectively. The agreement rate for the models is lower because of the uncertainty that comes with their documentation. Since Hugging Face does not enforce a standard for the models, often some section titles could not be classified as part of the model card standards.

Furthermore, while we detected a series of dependencies between datasets through the procedures detailed in Section 3.5, there is no explicit understanding of what these relationships mean. We cannot unambiguously say that one dataset is derived from another or vice-versa, we can only say that there is a relationship. For this reason we adopted the notion of the core-periphery structure because the statistical data that we found placed a great deal of focus on outliers. Moreover, the stochastic block modelling results were confirmed by our exploration of their datasets found to document their dependencies.

Additionally, while the documentation of models and datasets within Hugging Face is of poor quality, it is possible that many of those models and datasets are hosted elsewhere. This could mean that their lack of documentation at Hugging Face might not be reflected in their original sites.

**External Validity:** The analyses that we performed were focused on Hugging
Face. While we validated the existence of documentation issues within Hugging Face, there is no guarantee that the same can be said of other stores such as AWS Marketplace or Wolfram Neural Repository. However, given that Hugging Face has been assessed to be valued at over $80 million USD, and as the ”GitHub of machine learning” \(^\text{12}\) we believed it to be the best choice of model store for our study. Finally, the data dependency links and their (lack of documentation) should be studied on larger subsets of model store datasets.

\(^{12}\)https://www.forbes.com/sites/kenrickcai/2022/05/09/the-2-billion-emoji-hugging-face-wants-to-be-launchpad-for-a-machine-learning-revolution
Software engineering projects usually involve some sort of version controlling systems such as git. These elements are used in order to maintain a record of how the artifacts and code files of a software engineering project have evolved over time. More importantly, however, is that there is a wealth of information that can be extracted from these version control repositories. For instance, the number of changes throughout time, which files were modified, how they were modified and how such changes may be risky or not are all pieces of data that can be obtained from these repositories. In the context of machine learning projects, they use these version control systems similarly. For instance, one of Hugging Face’s features is to provide version controlling for datasets and models through git. Currently, there is little information on what
can be found on these repositories, how they have evolved over time and what sort of concerns exist throughout the repositories available at Hugging Face. In order to address this, we applied topic modelling to with LDA, statistical analysis and hypothesis testing to 7,877 dataset repositories and 37,887 model repositories. In addition to that we do the same with an additional 6,330 space app repositories (small UI applications that integrate models and datasets), and do static code analysis with python tools to determine the quality of the code present in these apps. To present a some of our findings, the 3 most relevant topics in dataset commit messages are \textit{Initial Uploads/Training}, \textit{updates}, and \textit{Versioning/Licensing}. Whereas with models the main topics are \textit{Initial Commits and Updates}, \textit{Training Specifications}, and \textit{Hyper-parameter Tuning}. Finally, on the subject of spaces the topics are focused on \textit{Initial Commits}, \textit{Application Updates} and \textit{New File Uploads and Requirements}. More than that, this research has found that over time the commit frequency of datasets and models is heteroskedastic, meaning that it is they are mostly randomly updated and modified.

\section{Introduction}

Machine learning products and solutions that are actively deployed into production require a rigorous process of development in order to be functional. This necessitates the use of writing code in programming languages such as python in order to deal with the many tasks that are present in the machine learning development cycle. This requires considerable effort in different facets of programming, mainly those that involve datasets, models and the applications that may utilize these two machine learning artifacts. To begin, datasets are considered to be the first integral
part of a machine learning solution, they are the key to generating a robust model that will predict or classify correctly [Paleyes et al., 2020]. In order to ensure that datasets work properly, a series of tasks such as data collection, imputation, feature scaling and continuous updating are necessary, something which requires the use of programming and versioning control systems. The next part is focused on models, which are the end result of obtaining and managing a dataset properly, in addition to being the second integral part of a machine learning solution [Paleyes et al., 2020]. Similarly to datasets, models require a series of coding tasks in order to function properly. These include training, validation, using pipelines and algorithms like grid search to find the model with the best performance. As a result this also leads to the use of version control systems and software engineering practices to develop these models. Furthermore, at some point when the right datasets and models are found, these are often integrated into a software solution, to make them readily available to the end users [Vogelsang and Borg, 2019]. This additional applications, are also handled like other software projects, in that they also use version controlling. Furthermore, Hugging Face provides an avenue into understanding these applications through spaces, small UI applications that use models and datasets, and that are also version controlled with software like git.

Hugging Face provides tools, infrastructure and version controlling systems, so that individuals and organizations can manage their datasets, models and applications. This is done due to the heavy reliance that exists on software artifacts. In order for companies to understand how they can use these tools to their advantage or possibly develop their own machine learning ecosystems they must understand the effort that is necessary to do this. The key to understanding this is to examine the commit
histories that are present in Hugging Face’s dataset, model and space repositories. For instance, the commit histories of a software project provide valuable information such as the number of lines of code inserted, the number of lines deleted, the number of files changed and which sections of code were changed. Additionally, previous research has defined a series of metrics such as cyclomatic complexity [McCabe, 1976] to measure the complexity of a program, as well as others such as the Delta Maintainability Metric [di Biase et al., 2019] which serves to calculate the overall risk associated to the commit changes made to a software project.

The aim of this chapter (and this research) is to provide a series of statistical measurements on the model, dataset and space repositories present at Hugging Face. This is focused on determining the overall effort that was required by the machine learning model store to maintain and modify their artifacts throughout time, specifically focusing on datasets, models and spaces. To address these problems, this research has focused on three research questions:

**RQ1:** What are the common topics that are present in the commit messages of datasets and models? – In terms of version controlling systems, the commit messages that practitioners make reveal the sorts of challenges that exist within the development of these projects. It is therefore essential to discover the most common topics present within the commit histories, in order to better understand which issues are being faced more frequently. As a preview of the results that were found, 3 topics were found for datasets, with these being *Training Files*(58.27%), *Initial Commits and Updates*(21.57%), and *Licensing*(20.16%), whereas with models it was found that the most common topics are *Initial Commits and Updates*(23.39%), *Training Specifications*(52.48%), and *Hyperparameter Tuning*(29.29%).
RQ2: What is the effort involved in maintaining and modifying datasets and models? – In addition to providing a wealth of information through the use of commit messages, commit histories also provide useful time series data to observe how a project has been changing over time. In addition to that, the information provided per change also gives valuable insight into the risk associated with a change to a dataset or model. To provide some information on the results, a series of statistical tests reveal that, in general the changes made to models are riskier than those of datasets. On average, with respect to DMM, dataset changes have an average \textit{dmm\_unit\_size} of 0.588, a \textit{dmm\_unit\_complexity} of 0.878 and a \textit{dmm\_unit\_interfacing} of 0.854, whereas models have a \textit{dmm\_unit\_size} of 0.548, a \textit{dmm\_unit\_complexity} of 0.718 and a \textit{dmm\_unit\_size} of 0.792. Further statistical tests reveal that changes to models are riskier.

RQ3: What are the particularities of SE4AI applications, such as Hugging Face’s spaces? – Another facet of the machine learning development cycle is the creation of software applications that integrate models and datasets. This research explores Hugging Face’s space applications similarly to how the models and datasets were explored, especially since they use much of the same infrastructure. An analysis of the most common topics was done to determine the issues present in these applications. Additionally, a statistical analysis using time series, hypothesis testing and descriptive statistics was elaborated to find out the effort necessary to maintain these applications. In contrast, to models and datasets, while the time series in for spaces are heteroskedastic, they also have strong auto correlations within at least 10 time lags with relation to insertions, deletions, lines of code modified and files modified, indicating that volatility within spaces is autoregressive.
4.2 Methodology

This section provides the details related to the methodology that was used to carry out this research. Section 4.2.1 outlines the goal of this research and the way in which this goal was pursued while the rest of the sections present the approach to address each RQ.

4.2.1 Goal

We relied on the Goal/Question/Metric template [Basili and Rombach, 1988] to define our research goal as follows: we analyze the commit histories of models, datasets and spaces present in model stores as well as the text based information found per commit message. We do this for the purpose of identifying the common issues within model, dataset and space development. Furthermore, we do this from the point of view of entities seeking to build their own machine learning applications and from the context of Hugging Face, which is our case study.

4.2.2 RQ1 Steps

Figure 4.1 represents the steps that were taken to carry out the topic modelling analysis for the datasets and models in RQ1.

Step 1.1 Extract Commits – First, the commits for both models and datasets are extracted from Hugging Face with the use of a tool called Pydriller [Spadini et al., 2018]. This tool will take a repository and extract its individual commits along with the messages present for every commit created by the people who worked with these models and datasets. The commit messages within these modifications are essential
for the next parts of this research question, as they are the main source of information for our topic modelling procedures. The commit messages of 7,877 datasets and 37,887 models were retrieved to carry out the analyses of this research question.

Step 1.2 Store commit messages – Second after obtaining the commit messages through pydriller, they are stored in a PostgreSQL database. The entirety of the messages present within the commits are stored, file by file into the database. This was done so that everything that was present within these messages was kept intact so that it could be used easily with LDA analysis [Blei et al., 2003]. In the later steps these stored messages are used to discover the topics present within the commit histories of these projects. 81,707 commit messages related to datasets and 274,186 commit messages related to models were stored for this analysis.

Step 1.3 Clean commit messages – Third the commit messages are cleaned by
removing stop words and punctuation. Every message is also broken down into a series of words which are then to be used by the LDA procedures to analyze the common reoccurring topics within the datasets and the models. These must be cleaned in order to ensure that the LDA procedures carefully capture the correct information from these messages.

**Step 1.4 Perform LDA Analysis** – Fourth, an LDA analysis is performed separately for both the dataset and model messages. In each of these the model was run multiple times with multiple parameters. For instance, several numbers of topics were used until optimal coherence and perplexity scores were obtained. In the case of this research these LDA models were finalized for both datasets and models with a total of 3 topics for each. The perplexity and UMass coherence scores for the datasets were 18.032 and -10.48 respectively. Perplexity and UMass coherence 10.916612 and -16.094738 respectively for the models.

To go further into detail about perplexity and UMass coherence, these two metrics are popular with LDA. Perplexity can be defined as *the measure of surprise for a model when seeing new data that has not been seen before*. In the case of LDA, this applies to words and topics, perplexity measures how surprised the model is to find new topics [Blei et al., 2003]. On the other hand the coherence score is a measure of the likelihood of the words of a topic being present together within a document. In the case of our experiments, we utilized UMass coherence [Stevens et al., 2012], which is the recommended method and most efficient metric available to measure LDA model performance.

Finally, the LDA analysis was elaborated with gensim [Rehurek and Sojka, 2011], a machine learning library created specifically for solving problems that require LDA.
Step 1.5 Perform Descriptive Statistics – Finally, this research presents a series of charts and tables that provide additional information about the topics found and how dominant these are per commit message. This provides valuable insight into how each topic plays a role in determining the concerns that exist within the commit messages of datasets and models.

4.2.3 RQ2 Steps

Figure 4.2: RQ2 Steps

Figure 4.2 represents the steps that were taken to carry out the topic modelling analysis for the datasets and models in RQ2.

Step 2.1 Extract Commits – First, the commits for the models and datasets were extracted from Hugging Face. This was done with pydriller [Spadini et al., 2018], but more importantly it was done to extract important information relevant to commits.
such as insertions, deletions, lines of code modified, and files modified. Additionally, with the use of DMM metrics [di Biase et al., 2019] information pertaining to the risk of a particular commit change was also obtained.

**Step 2.2 Store commit time data** – Second, the commit history data is stored into the postgresSQL data. This data holds a timestamp in addition to a series of information to mark what happened at a said point in time. For instance, the number of insertions per day is stored by taking the commit timestamp and the number of insertions at that particular point in time.

**Step 2.3 Calculate DMM Risk** – Third, as mentioned before, pydriller [Spadini et al., 2018] comes with several advantages that allow it to retrieve additional metric information in the way of DMM [di Biase et al., 2019]. To do this, the overall risk of each change must be calculated. Pydriller assigns a number from 0.0 to 1.0 to provide a risk score in accordance to what is stipulated by the DMM metrics. The closer to 0.0 a number is, the more likely it is to be a change that brings considerable risk, otherwise if a change has a score of 1.0 then it has very low risk.

To be more specific, DMM risk is calculated with three particular values. These are $dmm_{\text{unit size}}$, a measure of the number of lines that a code change brings to a function or method, with low risk thresholds being below 15 lines. $dmm_{\text{unit complexity}}$ which is the measure of cyclomatic complexity of a method or function [McCabe, 1976] (a measure of the number of linearly independent paths in a program module) of a code change, in which case low risk thresholds are below 5. The final measure is $dmm_{\text{unit interfacing}}$ which is the number of parameters that a method or function might have. For further information refer to Pydriller’s documentation \(^1\)

\(^{1}\)https://pydriller.readthedocs.io/en/latest/deltamaintainability.html
Step 2.4 Perform Descriptive Statistics – Fourth, a series of histograms are carried out on this data, in order to understand the frequency of many of the metrics that exist, as well as to understand how the risk associated data is distributed, given its continuous nature. This is done to provide a picture as to how changes are distributed in the models and datasets of Hugging Face, as well as to view how DMM risk is distributed.

Step 2.5 Perform Hypothesis Testing – Fifth, we test for powerlaw distributions as specified by previously researched methods [Klaus et al., 2011; Clauset et al., 2009] with the python powerlaw package [Alstott et al., 2014]. In addition to this, this research also utilizes a mann-whitney-u test to determine whether there is a significant difference between the data present for datasets and models in terms of the effort necessary to maintain these artifacts.

4.2.4 RQ3 Steps

Figure 4.3 represents the steps that were taken to carry out the topic modelling analysis for the datasets and models in RQ3.

Step 3.1 Extract commits – First, similarly to the previous steps commits are extracted from Hugging Face’s space repositories. However, because these elements are handled differently by the store, their extraction relied on interaction with more complicated APIs. In essence, it was required to work directly with http servers through the use of http requests to extract the list of repositories. These were then used in conjunction with pydriller to extract the data corresponding to each repository. The data for 6,330 spaces was retrieved for these experiments.

Step 3.2 Store commit data – Second, we take the data retrieved in Step 3.1 and
then store it into a PostgresSQL database. However, in contrast to the other RQs, this step also stores the app.py file of every repository. This file is the main file for spaces, most of the critical operational commands exist within this file, as such we stored these files to do further analysis on the next steps. Additionally, we also store the time series data and commit messages associated with spaces similarly to that of the datasets and models.

**Step 3.3 Perform LDA Analysis** – Third, we use the commit messages from the spaces that were stored in Step 3.2 to perform an LDA analysis similar to that of the datasets and models. Again, we utilize perplexity [Blei et al., 2003] and UMass [Stevens et al., 2012] to measure the performance of our topic modelling. In the case of spaces a perplexity score of 19.89 was found along with a UMass coherence of -15.16. Three topics were used after trying different numbers of topics until
an optimal value was reached. This was done on 66,409 commit messages.

**Step 3.4 Calculate DMM Risk** – Fourth, we take use the time series data initially retrieved in Step 3.1 and calculate the DMM risk measures similarly to how they were calculated before with the datasets and models. Once again we do this to measure the impact of the changes in Hugging Face’s space apps.

For more information on the DMM risk measures used (dmm_unit_size, dmm_unit_complexity, and dmm_unit_interfacing), refer to Step 2.3.

**Step 3.5 Hypothesis Testing** – Fifth, we apply White’s test to the time series of the space commit histories, in order to determine if heteroskedasticity is present. Additionally, we also perform an auto correlation analysis to determine if the time series in spaces are autoregressive, in order to determine whether past time steps influence the actions of future time steps.

4.3 Topic Modeling Analysis Results (RQ1)

*Training Files* dominates most documents with 58.27% of all topics, *Initial Commits and Updates* are the second most dominant topic with 21.57% of all topics and *Licensing* having 20.16% of all topics. Figure 4.4 presents a bar chart with the percentages corresponding to the dominant topics found for the datasets. We also present Figure 4.5 which shows the terms of use for a particular dataset in Hugging Face.

To put this into perspective, *Training Files* are the most dominant topic because of the key part that datasets play in the machine learning lifecycle, in this case it would appear that most of the commits are focused on the training files of the datasets. As has been mentioned before, much of the machine learning development lifecycle
depends on the quality of the dataset [Paleyes et al., 2020], furthermore other research has identified that one of the key difficulties related to machine learning solutions lies on the data quality aspects [Nahar et al., 2022]. In this case it appears that the version control history of the datasets at Hugging Face is largely concerned with the issues of training for datasets.

On the other hand Initial Commits and Updates and Licensing have similar percentages. While it is not unsurprising to see these 2 categories have less importance than the latter, it is surprising to see that licensing is the least important. As stated
before, licensing is a key issue when it comes to meeting legal thresholds set by governmental organizations. In the case of the version control history of datasets at Hugging Face it appears to be the least pressing issue when referring to how datasets change over time. This is a troubling concern, given that in the past the lack of interest in such issues has resulted in the indefinite halting of deployments of machine learning models [Paleyes et al., 2020]. Furthermore, there are a variety of privacy issues that have been identified over time, with these playing a key role in how licenses are developed and updated for datasets [Paullada et al., 2021].

Moreover, it seems that the main issue is not with ethical concerns, but more with the legal aspects of how a dataset might be used or which licenses intersect...
with the data inside. For instance, bigcode/the-stack\(^2\), a dataset made up of several programming languages requires an acceptance of terms of use. Figure 4.5 shows an example of this. This also gives weight to the claims previous research has made regarding requirements in machine learning having different issues (such as legal and ethical concerns) from those of present in software [Vogelsang and Borg, 2019].

*Training Specifications* is the dominant topic with 52.48% of all topics, *Hyperparameter Tuning* is second with 29.29% of all topics and *Initial Commits and Updates* represents 23.39% of all topics. Figure 4.6 presents a bar chart with the three topics along with the percentages that each one of these topics has obtained with regard to the total number of documents analyzed.

In terms of these results, similarly to the datasets, it is clear that a large part of the documents are focused on the training specifications of models. Just as with the datasets, models are the second key part of the machine learning development lifecycle [Paleyes et al., 2020]. As such it is expected that most of the efforts are focused on making sure that model training is the best possible that it can be. Furthermore, it can also be observed that *Hyperparameter Tuning* is the second topic, which also demonstrates that training and optimizing models for the future is a relevant concern for matters concerning machine learning solutions.

Moreover, an important part of ensuring model quality is done through testing. This is important because data is always changing, and as such if it is not handled then a model risks becoming obsolete in the future. In such cases, past research has created testing steps for models in order to account for these issues. Two key elements were hyperparameter tuning and model staleness. In the case of hyperparameter tuning tests were established to ensure that the correct hyperparameters are found [Breck

\(^2\)https://huggingface.co/datasets/bigcode/the-stack
et al., 2017]. More importantly, however is that these hyperparameters must also be updated as data changes. As such, it makes sense that concerns of updating and hyperparameter tuning are visible in the commit messages.

**Summary of RQ1:** There were three topics found by the LDA analysis of the datasets. *Training Files* (58.27% of documents), *Initial Commits and Updates* (21.57% of documents) and *Licensing* (20.16% of all documents). Similarly, three topics were found for models, and these were *Training Specifications* (52.48% of documents), *Hyperparameter Tuning* (29.29% of documents) and *Initial Commits and Updates* (23.39% of documents).
4.4 Statistical Analysis Results (RQ2)

The metrics related to insertions, deletions, number of lines modified and number of files modified are skewed positively and are leptokurtotic in the case of datasets which indicates that outliers dominate in terms of updates to these artifacts. Table 4.1 presents the mean, standard deviation, skewness and kurtosis of these measurements. In here we can see that these distributions are heavily influenced by outliers. The standard deviations are significantly large, with significant differences with the mean. More importantly all measures of skewness and kurtosis are larger than 40 and 2,000, marking these distributions as positively skewed and leptokurtotic, clear signs of outlier influence [Aggarwal, 2016]. This means, that in terms of updating and modifying datasets, they are generally not updated as frequently. To provide further insight into this Figure 4.7, four histograms with the different categories are presented. Through this figure, these issues of skewness and kurtosis can be observed.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Mean</th>
<th>Std</th>
<th>Skew</th>
<th>Kurt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertions</td>
<td>835.60</td>
<td>14,481.79</td>
<td>45.21</td>
<td>2,840.84</td>
</tr>
<tr>
<td>Deletions</td>
<td>246.39</td>
<td>7,913.06</td>
<td>94.92</td>
<td>11,557.95</td>
</tr>
<tr>
<td>Total Lines Modified</td>
<td>1,081.99</td>
<td>16,876.97</td>
<td>41.38</td>
<td>2,219.18</td>
</tr>
<tr>
<td>Total Files Modified</td>
<td>3.74</td>
<td>83.24</td>
<td>75.73</td>
<td>7,705.10</td>
</tr>
</tbody>
</table>

Table 4.1: Statistical Measures for Datasets

To further elaborate on this point, these figures show that for all metrics, it is usually lower values that have the highest number of occurrences. This means that in terms of these metrics commits, for the most part do not contain a lot of changes for the models and datasets Nonetheless, the heavy skewness values and kurtosis seem
to indicate that there are important outliers. Furthermore, the means of insertions, deletions, lines and files modified for datasets are 835, 246, 1,081, and 3.7 respectively, but their standard deviations are 14.81, 7,913, 16,876 and 83.24, which are large
numbers that are shifting the averages significantly.

In essence, if a developer is interested in using the datasets at Hugging Face they must keep in mind that the datasets are not as updated as they appear to be. This is important to consider if such datasets are to be used to train models or to build SE4AI applications, given that one of the main concerns when it comes to data is ensuring that it is properly updated to reflect changes that new generated may bring [Breck et al., 2017].

The metrics retrieved from the dataset commit histories are best fit by the power law distribution. Table 4.2 displays the results of a kolmogorov-smirnov test carried out to compare the distances between the Power Law, Exponential and Log Normal distributions. This test reveals that all four metrics are closest to the power law distribution, thereby showing that when it comes to maintaining datasets, the updates that are made to such artifact are power law distributed. Figure 4.8 shows a visualization of the power law fitting performed to identify these distributions.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Power Law</th>
<th>Log Normal</th>
<th>Exponential</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertions</td>
<td>0.04</td>
<td>0.03</td>
<td>0.25</td>
</tr>
<tr>
<td>Deletions</td>
<td>0.034</td>
<td>0.039</td>
<td>0.31</td>
</tr>
<tr>
<td>Total Lines Modified</td>
<td>0.03</td>
<td>0.05</td>
<td>0.29</td>
</tr>
<tr>
<td>Total Files Modified</td>
<td>0.06</td>
<td>0.09</td>
<td>0.40</td>
</tr>
</tbody>
</table>

Table 4.2: Kolmogorov-Smirnov Distances for Dataset Distributions

In keeping with what was said before, we can see that many of these distributions fit into the power law distribution. In software, previous research has described this as "the rich get richer" [Louridas et al., 2008] in terms of modules, meaning that the maintenance effort of certain projects is focused specifically on a few key modules of a software project. More importantly, previous research has found that code in software
Figure 4.8: Exponential Distribution Fitting for Models

projects, is prone to power laws, especially when it comes to code changes in commit histories [Lin and Whitehead, 2015]. It would seem that this phenomenon stretches beyond software projects, however, as it is also present within datasets. Figure 4.8 serves to further illustrate this point.

In the case of the commit histories of the datasets, developers interested in using
these datasets for model training and applications should keep in mind that, the types of changes that occur within Hugging Face are usually not significant when it comes to the datasets. They mostly remain the same with little to no changes occurring within these metrics.

**In terms, the metrics extracted for models are also positively skewed and leptokurtotic, demonstrating that outliers also play a large role in the updates that are made to models.** In contrast to datasets, this appears to be slightly less relevant, however. Table 4.3 shows that the differences between means and standard deviations are significantly high, whereas skewness for all measures is larger than 20 and kurtosis is larger than 1,000 showing that the data is positively skewed and the leptokurtotic. This is different from datasets, in which updates appeared to carry less information, whereas with models it appears that the skewness and kurtosis make it so that they are being updated more frequently. We also present Figure 4.9 which provides a visualization of how these data points are distributed.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Mean</th>
<th>Std</th>
<th>Skew</th>
<th>Kurt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertions</td>
<td>4,952.15</td>
<td>27,767.60</td>
<td>20.48</td>
<td>1306.61</td>
</tr>
<tr>
<td>Deletions</td>
<td>163.53</td>
<td>5,158.37</td>
<td>98.07</td>
<td>15,848.48</td>
</tr>
<tr>
<td>Total Lines Modified</td>
<td>5,115.68</td>
<td>28,499.81</td>
<td>20.97</td>
<td>1,341.79</td>
</tr>
<tr>
<td>Total Files Modified</td>
<td>3.06</td>
<td>26.57</td>
<td>98.50</td>
<td>13,713.39</td>
</tr>
</tbody>
</table>

Table 4.3: Statistical Measures for Models

Similarly to datasets, models are also not as frequently updated as they might seem. Once again, the frequencies of the changes with respect to insertions, deletions, lines modified and files modified are on the lower side. In this case, the means are 4,952, 163, 5115, and 3.05 respectively, with standard deviations of 27,767, 5,158,
28,499 and 26.57. Once again, there are key outliers in which a disproportionate amount of changes were made through commits. Previous research indicates that models that are deployed successfully into production require continuous retraining
in order for them to adapt to the changing circumstances of new data [Breck et al., 2017, Vogelsang and Borg, 2019]. As such, developers should take into account that the models they may download from Hugging Face are provided ”as is”, meaning that it might be necessary for them to modify said models themselves to make them work appropriately for custom use cases.

Similarly to datasets, the metrics related to models can also be best described as following a power law distribution, however the distance between these elements is even closer with regard to the models. Table 4.4 presents the distances calculated between the model metrics and the Power Law, Exponential and Log Normal distributions. The closest distances are with the Power Law distribution. Figure 4.10 visualizes the distances between the distributions.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Power Law</th>
<th>Log Normal</th>
<th>Exponential</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertions</td>
<td>0.08</td>
<td>0.10</td>
<td>0.13</td>
</tr>
<tr>
<td>Deletions</td>
<td>0.10</td>
<td>0.21</td>
<td>0.89</td>
</tr>
<tr>
<td>Total Lines Modified</td>
<td>0.08</td>
<td>0.21</td>
<td>0.87</td>
</tr>
<tr>
<td>Total Files Modified</td>
<td>0.09</td>
<td>0.14</td>
<td>0.45</td>
</tr>
</tbody>
</table>

Table 4.4: Kolmogorov-Smirnov Distances for Model Distributions

To put these results into perspective, the first observation to keep in mind is that power laws are common in git commit changes [Lin and Whitehead, 2015]. These are present in several repositories for software projects. However, in the case of machine learning models these are also a phenomenon, which implies that some of the same issues that exist in software are also present within models. Further to that, the idea that ”commits usually involve a large amount of small monthly changes and a small amount of large changes” is present within the model commit histories [Lin and Whitehead, 2015].
In addition to that, power laws have also been observed to occur within machine learning contexts through the use of Jupyter notebooks [Koenzen et al., 2020]. In that case, the power laws were present because of code reuse throughout Jupyter notebooks. In essence, this means that power law distributions are also common throughout the machine learning development cycle.
Developers using models from Hugging Face should keep in mind then, that the way in which models are updated follows a power law distribution, meaning that small changes are what is expected when a model is updated in Hugging Face. As such, they should not rely upon Hugging Face to continuously update the models in a fashion that changes the models significantly.

The effort and risk necessary to maintain models is larger than the effort and risk that is needed to maintain datasets. Table 4.5 shows the results of a Mann-Whitney-U test that was performed to measure the variances of the metrics retrieved from the commit histories of models and datasets. The p-values in these comparisons are below 0.05, showing that it is most likely that when it comes to models and datasets, the maintenance and updating of these artifacts is significantly riskier and more effort intensive when it comes to models. In addition to that, we also present Figure 4.11 and Figure 4.12 which present the distributions of risks per commit for models and datasets.

<table>
<thead>
<tr>
<th>Time Series</th>
<th>MannWhitney T-Statistic</th>
<th>MannWhitney P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertions</td>
<td>77,194.5</td>
<td>1.79e-01</td>
</tr>
<tr>
<td>Deletions</td>
<td>91,968</td>
<td>5.27e-12</td>
</tr>
<tr>
<td>Total Lines Modified</td>
<td>77,750</td>
<td>1.29e-01</td>
</tr>
<tr>
<td>Total Files Modified</td>
<td>95,309.5</td>
<td>2.52e-20</td>
</tr>
<tr>
<td>DMM Unit Size</td>
<td>95,309.5</td>
<td>1.66e-01</td>
</tr>
<tr>
<td>DMM Unit Complexity</td>
<td>34,853.5</td>
<td>9.19e-12</td>
</tr>
<tr>
<td>DMM Unit Interfacing</td>
<td>42,342.5</td>
<td>5.64e-03</td>
</tr>
</tbody>
</table>

Table 4.5: Mann-Whitney U Test Results
On average the risks with dataset changes have means of 0.588 for \textit{dmm\_unit\_size}, 0.878 for \textit{dmm\_unit\_complexity} and 0.854 for \textit{dmm\_unit\_interfacing}, whereas models \textit{dmm\_unit\_size} has a mean of 0.548, \textit{dmm\_unit\_complexity} has a mean of 0.718 and \textit{dmm\_unit\_size} has a mean of 0.792. In terms of unit sizes, both the datasets, and models have changes that can be considered moderate (0.5) in terms of risk, whereas with cyclomatic complexity and unit interfacing the changes contain a slight risk (0.7 and 0.8). However, it is notable that the changes for models are riskier, which is validated both by the values present, and the hypothesis testing elaborated with mann-whitney-u. However, models have to deal with other issues such as concept drift, in which the joint distributions of inputs and outputs drift into different values.
Figure 4.12: Model Risk Distribution

because of events that alter data over time [Paleyes et al., 2020]. Furthermore, updating models also comes with the risk of having data become noisy, and thus require denoising techniques to eliminate stochasticity and to ensure that models maintain backwards compatibility with old tasks [Paleyes et al., 2020].

In this case, developers should be aware of two particularities. First, is that if they use Hugging Face’s models for their own purposes, they will have to deal with how code changes to said models are in overall riskier. While it has been observed that Hugging Face does not update models significantly, it still does updates, however small they may be. These might have issues for the models being used. Furthermore, it was also mentioned that since Hugging Face provides models “as is”, then developers
are likely to have to modify them for their own purposes. In that light, they should consider that their changes could also carry considerable risk on the functioning of the models themselves.

**Summary of RQ2:** The commit histories of both models and datasets are heteroskedastic, meaning that they are the results of stochastic processes. Furthermore, in terms of insertions, deletions, lines modified, and files modified, the data shows that these data points best fit a power law distribution. Finally, in terms of effort and risk, models are significantly more difficult to maintain and manage.

### 4.5 Spaces Analysis Results (RQ3)

In terms of topic dominance, *New File Uploads and Requirements* is the most dominant topic with it being the most relevant in 52.10% of all documents. *Application Updates* is the second with 27.89% of all documents and *Initial Uploads* is the last with 20.04% of all documents. Figure 4.13 presents a bar chart that denotes the percentage share of documents that each topic dominates.

The key finding here is that requirements play a large role in the development of these applications, and thus it is the most relevant topic. In contrast to the findings for models and datasets, requirements were an indirect concern, whereas with spaces it is a direct one. Spaces are end user applications, meaning that regular users, as opposed to developers and data scientists, directly interact with these applications. One such example is the DALL-E Mini space[^1] in which end users can interact with

[^1]: https://huggingface.co/spaces/dalle-mini/dalle-mini
a "mini" version of DALL-E to generate images based on the text that is inputted into the model. Furthermore, spaces are UI applications, which in turn require a particular design strategy to make them appealing and easily usable to end users. This particularity makes it essential that requirements be understood correctly in order to deliver an appropriate product. The commit messages are reflecting these issues as a result.

**Heteroskedasticity is present in the commit histories of space apps.** Table 4.6 shows the result of White’s test of heteroskedasticity on the time series data of the space apps. Whereas Figure 4.14 and Figure 4.15 display the volatility
of the changes for the metrics extracted with respect to the commit histories of the spaces.

<table>
<thead>
<tr>
<th>Time Series</th>
<th>White Statistic</th>
<th>White P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertions</td>
<td>33.42</td>
<td>5.52e-08</td>
</tr>
<tr>
<td>Deletions</td>
<td>18.11</td>
<td>1.16e-04</td>
</tr>
<tr>
<td>Total Lines Modified</td>
<td>30.67</td>
<td>2.19e-07</td>
</tr>
<tr>
<td>Total Files Modified</td>
<td>47.77</td>
<td>4.24e-11</td>
</tr>
</tbody>
</table>

Table 4.6: White Test Statistics for Spaces

The development efforts of SE4AI applications differ greatly from that of software projects. To start, they require the collaboration of different counterparts with a wide arrange of skills, such as that of data scientists, data engineers and software engineers [Nahar et al., 2022]. In that regard, the development process can lend itself to particular situations that might make the development somewhat uncertain. However, it must be established that the development process of software is not a random statistical process but the careful decision-making of developers [Lin and Whitehead, 2015]. Even so, the development process of spaces appears to be completely heteroskedastic, meaning that the changes made, and how much they affect the spaces are stochastic in nature. This is, perhaps a reflection of the uncertainty that comes with creating SE4AI applications, a subject which is exacerbated by the fact that ML practitioners lack specific ways of qualifying and expressing requirements, along with their intended targets and trade-offs, which are often heavily influenced by their domain context [Horkoff, 2019].

In this regard, developers seeking to create their own applications with datasets and models should keep in mind that there are certain uncertainties that come with such a task. In this case Hugging Face, can be observed as a preview of the issues
that might come with developing an SE4AI application.

The time steps associated with the metrics (insertions, deletions, lines modified and files modified) within the spaces are autocorrelated. Figure 4.16 and Figure 4.17 present the results of auto correlation functions used to
identify whether time steps of a time series are correlated.

To put this into perspective, and to contrast it with the previous discovery of heteroskedasticity, the results here demonstrate that the development process of these applications and their metrics (insertions, deletions, lines modified, and files modified)
is not entirely random, even if they are heteroskedastic. This means, that the previous commits and their changes affect the volatility of future commits and changes. This would mean that as far as the development process of these applications is concerned, there is likely an explicit process to follow.
The metrics related to commit histories for spaces are influenced by outliers. Table 4.7 shows that the mean and standard deviation have a large difference between each other, showing that the data itself varied. Furthermore, the measures of skewness are greater than 50 and while kurtosis is greater than 4,700,
showing positively skewed and leptokurtic distributions, which are often subject to movements from outliers [Aggarwal, 2016]. Additionally, we further illustrate this point by providing Figure 4.18 which visualizes what these elements look like with a series of histograms denoting the different metrics obtained.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Mean</th>
<th>Std</th>
<th>Skew</th>
<th>Kurt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertions</td>
<td>357.35</td>
<td>8825.75</td>
<td>62.62</td>
<td>5642.75</td>
</tr>
<tr>
<td>Deletions</td>
<td>39.83</td>
<td>2021.427</td>
<td>113.81</td>
<td>16,260.69</td>
</tr>
<tr>
<td>Total Lines Modified</td>
<td>397.18</td>
<td>9,083.69</td>
<td>58.88</td>
<td>5,073.59</td>
</tr>
<tr>
<td>Total Files Modified</td>
<td>2.37</td>
<td>32.96</td>
<td>61.31</td>
<td>4,799.53</td>
</tr>
</tbody>
</table>

Table 4.7: Statistical Measures for Models

Similarly, to models and datasets, and perhaps unsurprisingly, the changes in spaces are also small and on the low side when it comes to modifications of these applications. This is perhaps explained by the fact that spaces are not full-fledged applications intended to be used in more professional settings, but instead they are a way for the authors at Hugging Face to provide a small example of what an application using their datasets and models might look like.

In that regard, developers should definitely consider these applications as an example of what is necessary to make a successful SE4AI application, but they should also keep in mind that they are not as complex as other applications in other domains might be. They should not be used as a template to design or create other applications that use machine learning models and datasets, since they are not representative of the overall complexities of larger systems.

The distributions for the metrics can be best described as power law distributions. Table 4.8 shows the Kolmogorov-Smirnov distance between the power law distribution and the distributions found in this data. Figure 4.19 shows the
Figure 4.18: Spaces Commit Data Histograms

Visualization of the power law fitting tests that were created in order to determine if the metric distributions are indeed approximate of the power law distribution.
Similarly, to models and datasets, the distances of these metrics are closest to the power law. However, in contrast to models and datasets, the distance between these distributions and the power law distribution is smaller. This is not surprising, given that these applications fall within the domain of software engineering, and thus are more likely to follow these patterns than the datasets and models. However, given this discovery between the similarities of SE4AI applications and regular software engineering applications, this might open up clues for other similarities that might exist between these domains, in addition to potentially providing similar solutions to the problems that exist for SE4AI as a whole.
Summary of RQ3: *New File Uploads and Requirements* is the most relevant topic with 52.10% of all documents. *Application Updates* is the second with 27.89% of all documents and *Initial Uploads* is the last with 20.04% of all documents. Heteroskedasticity is present in the
commit histories of spaces, which means that the development of these applications is volatile, but unlike models and datasets, the volatility of their development is strongly dependent on the past. Outliers also play a large role in the changes that are made to spaces, given how closely the distributions resemble the power law distribution.

4.6 Implications

4.6.1 Implications for Researchers

The main implication as far as this research is concerned is that the current state of the development of machine learning solutions is one of randomness and uncertainty. Thus far a series of problems have been pointed out by previous research, for instance collaboration between data scientists, data engineers and software engineers is a key issue [Nahar et al., 2022]. It would appear that this issue is somewhat reflected in the information retrieved. Moreover, the development of spaces as evidenced by the data retrieved from Hugging Face, is volatile, something that has been pointed out by previous research [Vogelsang and Borg, 2019]. Additionally, it is also readily apparent that many of the phenomenons, such as power laws, present in traditional software engineering are also visible in models, datasets but more importantly in SE4AI applications. This research provides an avenue into many of these issues by providing numerical and statistical evidence of how they manifest themselves into the development process. Hopefully, this will lead to a better understanding of how these issues can be approached and dealt with in future research.
4.6.2 Implications for Developers

Armed with the information provided by this research, developers wanting to create their own SE4AI applications have been provided with a series of issues to consider. First, in terms of datasets, while Hugging Face facilitates the procurement of these artifacts, the store does not necessarily facilitate the maintenance of such items. Additionally, the store presents restrictions of how the datasets can be used through Terms of Use that developers need to acknowledge. Even so, developers should be wary of utilizing datasets without considering the legal implications of such items. The store may provide legal agreements such as Terms of Use, but these terms cover only certain things such as specific licenses, in addition to being different on a dataset to dataset basis.

Second, in terms of models, developers should be aware that they are not updated frequently, and that in the same fashion as datasets, they cannot count for support in terms of maintenance from the store. More importantly, they should consider that they will have to put their own efforts to ensure that models do not become stale and do not suffer from issues such as concept drift [Sculley et al., 2015, Breck et al., 2017]. More importantly, developers should carefully consider that changes to models are, in fact, riskier than changes to datasets. This implication is important for both Hugging Face, and for any development of models that may be elaborated later on.

Finally, in terms of spaces, developers have been presented with a picture of how SE4AI applications can be built, and what they entail in terms of development efforts over time. In this case the process demonstrated at Hugging Face shows that the process itself is uncertain. Even if Hugging Face provides a series of benefits for SE4AI application development, it is not immune to the common issues that
exist in machine learning in general [Paleyes et al., 2020, Sculley et al., 2015, Horkoff, 2019, Nahar et al., 2022, Sculley et al., 2015, Fischer et al., 2021, Paullada et al., 2021]

4.7 Threats to Validity

In this section we present the threats to the validity [Wohlin et al., 2012] of this research.

**Construct Validity** This research was carried out with the use of several statistical tests such as White’s test of Heteroskedasticity, the Mann-Whitney-U test to measure variances and means between populations and the distribution fitting and distance measuring for power law distributions. These statistical tests are all backed by scientific research. However, their results are meant to suggest the most likely outcome, they do not give a strict result that reflects one outcome or another. In that manner, different tests may reveal other information within the data. Even so, these tests were performed according to their need. White’s test was used because of its previous usage in financial time series, which are in general the results of stochastic processes much like the time series of the commit histories. The power law fitting mechanisms were selected because of how often these distributions can be found in software projects. Finally, Mann-Whitney-U was selected because of the lack of normality in the data as well as its non-parametric nature, something to which other tests like ANOVA are not suited for.

**Internal Validity** The time series data found for the models, datasets and spaces was not periodical meaning that there was not always data to be found at a particular moment in time. As such, it was necessary to account for this missing data by interpolating the time series to be able to properly assess them. However, the missing
data was in the case of spaces was 39%. Nonetheless, the data that we use, has been used for empirical purposes only, and not for prediction or for application purposes. We only provide information about the observations were found here. There are no pronouncements about whether this data is indicative of future behavior within those commit histories.

**External Validity** The results found for this research are mostly centered on models and datasets found in Hugging Face. These models may appear in other repositories outside Hugging Face and may diverge in commit histories and updates elsewhere. However, given that Hugging Face is a popular platform with models of every type, including language, image, reinforcement learning and audio, these results encompass a wide variety of models and datasets that should make it sufficient to consider what has been found here as accurate to what is present elsewhere. Nonetheless, when it comes to spaces, these applications are a construct designed within Hugging Face, while they may be applications that fit into the SE4AI paradigm, these applications have been developed solely with Hugging Face in mind. Other SE4AI applications may have different complexities and concerns compared to those that were found in here.
Conclusions and Future Work

5.1 Chapter 3 Conclusions

The main goal of Chapter 3 was to explore the documentation practices within model stores with reference to standards established by the machine learning community [McMillan-Major et al., 2021, Mitchell et al., 2019], empirically validating observations and issues on the machine learning development cycle [Nahar et al., 2022, Sculley et al., 2015, Paleyes et al., 2020]. We did so by examining 55,280 model cards and 6,758 dataset cards at Hugging Face in a quantitative fashion, as well as by exploring the hidden dependencies that exist between datasets, and determining how well these were documented.

In terms of general conclusions we can say that at this moment, datasets and
models in Hugging Face are poorly documented. The data shows that the majority of models (60.38%) and datasets (71.52%) have no documentation. Even if a dataset or model is documented, there is no guarantee that it will be fully and consistently documented. This suggests that the standards that are currently being used are insufficient or that there is a significant disconnect between what users think should be documented and what the standards think should be documented. Moreover, little knowledge seems to be available in terms of dependencies between datasets and how these are documented. Changes in practices and tool support are necessary to reduce the risk of reusing third party datasets and models through more complete, higher-quality documentation.

5.2 Chapter 4 Conclusions

The main goal of Chapter 4 was to measure the effort necessary to maintain machine learning artifacts and SE4AI applications with reference to problems that have been known to occur in their development [Nahar et al., 2022, Paleyes et al., 2020, Vogelsang and Borg, 2019]. It was done by examining 7,877 dataset repositories, 37,887 models and 6,330 spaces which accounted for more than 300 GB of data.

In terms of general conclusions it is clear that the topics of requirements, while not dominant in all cases, is clearly of importance in models, datasets and spaces. Through datasets, we see it through the Licensing topic, in which 20.16% of documents show it as the dominant topic. Whereas with models this can be observed through Hyperparameter Tuning which dominates 29.29% of all documents. Meanwhile, in spaces requirements are the most relevant, with 52.10% topic dominance among all documents. Furthermore, even with these considerations, it can be said
that the development of spaces is volatile given the presence of heteroskedasticity in the commit histories of these spaces, which validates previous claims in which it has been said that machine learning solutions introduce uncertainty into software projects [Vogelsang and Borg, 2019, Horkoff, 2019]. Finally, in terms of effort, it is clear that datasets require less than models, and that commits that modify dataset code are less risky than those that modify models. Generally speaking, the machine learning development process comes with similar issues to that of software, but also has its own problems that must be addressed differently.

5.3 Future Work

To conclude this thesis, we provide three new avenues in which future work could be developed in terms of SE4AI.

- **Tools for automating and suggesting documentation based on standards and common practices.** While Hugging Face has provided its own set of standards and templates that are readily available for users and developers to work on their own documentation, our results have shown that this has been ineffective. In contrast, there are accounts of recommender systems and other automation tools that have accomplished better results with documentation practices [Aghajani et al., 2020a]. These systems are tailored to software projects, but they can probably be built and adapted to function properly with ML concerns.

- **Tools for dependency resolution and detection for datasets.** It is clear that dependencies between datasets are a common issue, it is present in several cases,
and they have the potential to cause serious issues that are more difficult to manage and to detect than in software projects [Sculley et al., 2015]. Nonetheless, the development of such tools, has for the most part been absent. Throughout our analysis, we found that many datasets depend on other datasets, but that these dependencies are rarely listed or accounted for. This would seem to be a problem caused by lack of tooling. In the case of Hugging Face there are no systems that address these issues, and thus when using one of their datasets, it is a risk that must be considered when utilizing them for deployment purposes.

- **Development of better practices for managing and developing SE4AI applications, datasets and models.** Given the nature of machine learning projects and what is involved in them, it is apparent that an effort into determining how to best manage these projects is necessary. Unlike, software projects, machine learning projects require the collaboration of different disciplines [Nahar et al., 2022] and they bring with them an element of uncertainty that is not present in software [Vogelsang and Borg, 2019]. Our results have found that the way in which commits are done and managed within Hugging Face as a store is dependent on outliers and power laws, in addition to being somewhat stochastic in the case of spaces. As a result, developers interested in using this store should keep this in mind before selecting these models. Furthermore, they should also consider the implications of these developments for the maintenance of their models, datasets and applications.


