A First Look at Model Supply Chain: From the Risk Perspective

Ziqian Chen*
Fudan University
Shanghai, China
25213050138@m.fudan.edu.cn

Zekai Chen*
Fudan University
Shanghai, China
25213050135@m.fudan.edu.cn

Susheng Wu*†
Fudan University
Shanghai, China
scwu24@m.fudan.edu.cn

Bihuan Chen*†
Fudan University
Shanghai, China
bhchen@fudan.edu.cn

Wenyan Song
Carnegie Mellon University
Pittsburgh, China
wenyans@andrew.cmu.edu

Yiheng Huang*
Fudan University
Shanghai, China
yihenghuang23@m.fudan.edu.cn

Zhuotong Zhou*
Fudan University
Shanghai, China
zhouzt23@m.fudan.edu.cn

Yiheng Cao*
Fudan University
Shanghai, China
caoyh23@m.fudan.edu.cn

Xin Peng*
Fudan University
Shanghai, China
pengxin@fudan.edu.cn

Abstract

The rapid proliferation of publicly available artificial intelligence models on platforms such Hugging Face has formed a complex *model supply chain*, where base models are continually transformed, e.g., by fine-tuning, quantization, and merging, into new derived models. Despite its critical importance for reuse, provenance, and risk management, this supply chain remains poorly characterized at scale.

We construct the first comprehensive model supply chain based on models on Hugging Face as of June 25, 2025, which consists of 1.82 million models as well as 0.54 million dependency relations. Then, we conduct the first systematic study to characterize the usage, evolution, quality, and risk of this model supply chain. Finally, we provide actionable implications for model maintainers, consumers, platform designers, and researchers to foster this new direction in the era of AI.

CCS Concepts

• **Security and privacy → Vulnerability management**; • **General and reference → Empirical studies**; • **Computing methodologies → Artificial intelligence**.

ACM Reference Format:
Ziqian Chen, Zekai Chen, Susheng Wu, Bihuan Chen, Wenyan Song, Yiheng Huang, Zhuotong Zhou, Yiheng Cao, and Xin Peng. 2026. A First Look at Model Supply Chain: From the Risk Perspective. In *2026 IEEE/ACM 48th International Conference on Software Engineering (ICSE '26), April 12–18, 2026, Rio de Janeiro, Brazil*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3744916.3773171

*Z. Chen, Z. Chen, S. Wu, B. Chen, W. Song, Y. Huang, Z. Zhou, Y. Cao, and X. Peng are with the College of Computer Science and Artificial Intelligence, and Shanghai Key Laboratory of Data Science, Fudan University, China.
†Susheng Wu and Bihuan Chen are the corresponding authors.

ICSE '26, Rio de Janeiro, Brazil

ACM ISBN 979-8-4007-2025-3/26/04
https://doi.org/10.1145/3744916.3773171

1 Introduction

Public models on model hubs (e.g., Hugging Face) allow developers to reuse model structures and/or parameters to generate new ones, and substantially improve the productivity in developing artificial intelligence application. Yet the very openness of public models that empowers this productivity also breeds a dense and rapidly evolving *model supply chain*. In this ecosystem, models are continuously created, transformed, and re-distributed. We call the *dependency relation* between two models exists when one model is generated from another through a *transformation operation* (e.g., *fine-tuning*, *merging*, or *quantization*). In such a model dependency, the original model is called the *base model*, and the generated model is called the *derived model*. Despite its benefits, model supply chain still remains poorly understood. Infrequent maintenance, inconsistent quality control, and growing safety and security threats can undermine both the reusability and the trustworthiness of models [3]. A concrete and comprehensive view is therefore needed to consider *not only* upstream providers, who invest heavily in model design and may suffer intellectual property violations, *but also* downstream consumers, who often lack information signals about the provenance, quality, or risks of the models on which they depend.

Although several studies have been conducted on public models, none of them can contribute such a view. For example, some studies reveal the explosive growth in the number of public models and model chains [3, 12, 20], yet fail to explore the transformation operations and dependency topologies that define the real-world model supply chain. Some studies analyze commit histories to show that most model updates concern infrastructure or metadata [2], but they leave unexamined how such updates affect other models in the model supply chain. Additionally, some studies examine the completeness of key fields in model documentation, leaving unaddressed the correctness of critical fields [13, 20, 29]. Moreover, while some studies have demonstrated that risks can propagate from upstream models to downstream models through specific transformations (e.g., parameter-efficient fine-tuning or merging) [4, 21, 30], they typically focus on isolated cases and fail to uncover common risk propagation patterns across different transformation operations. These studies have two key limitations: (i) a limited analysis scale and scope

that fail to reflect the real-world model supply chain, and (ii) a lack of a unified understanding of how model usage, evolution, quality, and risk shape and impact the real-world model supply chain.

To address these limitations, we present the first systematic and quantitative empirical study to investigate *usage*, *evolution*, *quality*, and *risk* across the entire Hugging Face platform. Our study analyzes both models and the dependency relations that interconnect them, which are the most fundamental elements in model supply chain. We aim to answer four key research questions.

- **RQ1: Usage Analysis.** What is the usage distribution of models across different transformation operations in Hugging Face platform, and how do these operations give rise to dependency chains and clusters in the model supply chain?
- **RQ2: Evolution Analysis.** What is the intensity of additions, deletions, and updates of models in the model supply chain, and how does such evolution impact the usage of models?
- **RQ3: Quality Analysis.** How prevalent are quality problems in critical metadata fields of models in the model supply chain, and how do these quality problems affect the usage of models?
- **RQ4: Risk Analysis.** To what extent do performance, safety or security risks in base models propagate to derived models across different transformation operations?

We conduct the *usage analysis* on all the 1.82 million models from Hugging Face as of June 25, 2025, and we model the model supply chain's dependency topology at thee key levels, i.e., dependency relations, chains, and clusters. We find that 31% of models have dependency relations, but they drive 88% of total downloads. Fine-tuning generates 79% of all derived models, while quantization and merging respectively generate 19% and 2%. High-degree hub models (e.g., Qwen/Qwen1.5-0.5B with 32,497 dependents) serve as critical bases.

We conduct the *evolution analysis* over 17 weekly snapshots. We observe 27,311 model additions and 6,190 model deletions per week. Most additions (72.1%) are isolated (having no dependency relation with others); and fine-tuning still dominates relational growth (generating 72.8% of the newly derived models). Most deletions (70.5%) are isolated as well, yet when base models are deleted even at a low rate, they disrupt hundreds of downstream chains and dozens of clusters, threatening reproducibility. While additions vastly outnumber deletions, each deletion of a base model cascades through the model supply chain. One in ten models update in the 17 weekly snapshots, but only 0.1% of the update information is reflected in metadata.

We conduct the *quality analysis* by statistically sampling 378 models from the top 10% most-downloaded text generation models (24,686) due to their wide usage. We observe that large metadata gaps. Specifically, 83% of the sampled models omit performance metrics, and 28% have no license. 34% of those LLMs lack prompt templates, and 77% of data-needed models do not cite their training dataset. While declared dependencies are accurate (97% in precision), recall is only 69%, making one-third of derived models untraceable.

We conduct *the risk analysis* on 105 dependency relations, covering real-world transformation operations fine-tuning, quantization, and merging, whose base and derived models have high downloads. We adopt state-of-the-art benchmarks that provide both datasets and risk evaluation of hallucination, jailbreak, and prompt injection. We find that parameter-efficient fine-tuning (PEFT) preserves risk profiles, whereas standard supervised fine-tuning (SFT) amplifies risks. Low-bit quantization tends to increase hallucination and jailbreak rates, especially at 2-bit. Merged models snap to one base model's risk extreme rather than averaging, yielding a high variance.

Based on our findings, we provide actionable implications for various stakeholders, i.e., upstream model maintainers, downstream model consumers, Hugging Face platform designers, and researchers. For example, maintainers need to publish clearly versioned models and track every change; consumers must test for risks and list all steps they take; platform designers should make it easy to see a model's bill of material including history, license, dependency, performance, and risk metrics; and researchers should build tools to recover missing relations and predict how risk spread, which can make sure this fast moving model supply chain stays trustworthy for everyone.

In summary, this paper makes the following contributions:

- **The First Comprehensive Model Supply Chain.** We built the first comprehensive model supply chain, which consists of 1.82 million models as well as 0.54 million dependency relations, enabling the future reproducibility and extensions.
- **The First Systematic Study on Model Supply Chain.** We conducted the first systematic study to investigate the usage, evolution, quality and risk of the model supply chain, raising awareness of this new direction in the era of AI.
- **Actionable Implications.** We provided actionable implications for various stakeholders in the model supply chain, fostering the safe and secure development of the model supply chain.

2 Related Work

Usage Analysis. Public model hubs like Hugging Face have significantly changed the way models are shared and consumed [12]. In recent years, the number of models available on the Hugging Face hub has grown dramatically [3], enabling developers to reuse pretrained architectures with unprecedented ease [9, 18]. In contrast to classical software dependency management where reused code is typically imported as a passive, version-pinned library [25, 27], model reuse usually entails an *active* operation (e.g., fine-tuning, quantization, or merging) that creates a new descendant artefact inheriting weights and behaviors from one or more upstream models [28, 32]. These heterogeneous operations weave a dense and dynamic model supply chain whose edges encode more than simple "uses" relation. Although Stalnaker et al. [20] describe model provenance as a single path from a base model to a sink model, their linear abstraction omits both the set of admissible transformation operations and the rich graph topology in the real-world model supply chain.

Evolution Analysis. Like software packages, public models also evolve. Developers continuously add, delete, and update models. Castano et al. [2, 3] uncover that most commits concern training infrastructure, generated artefacts, or metadata, and that commit intent shifts across a repository's lifetime. Yet prior work leaves open the question of how such evolution actions reverberate through the model supply chain. Introducing a new base model can spawn entire downstream families; if the newcomer is malicious (e.g., embeds a backdoor behavior [19, 31]) or unreliable (e.g., hallucinates), every descendant inherits that risk [10]. Conversely, deleting a model can also hurt the usability and traceability; e.g., deleting a model can break parameter-efficient fine-tuning (PEFT) adapters that depend on the missing weights. We therefore quantify both the frequency

with which models evolve, and the downstream impact of each evolution event on the model supply chain.

Quality Analysis. With the expand of models in Hugging Face, concerns about model quality have intensified [11, 13, 17, 20, 22, 29]. Existing studies reveal that many entries lack documentation, evaluation metrics, or dataset links, and that over half omit even a free-text description [20, 29]. Additionally, fewer than 5% provide a complete dataset provenance [17], while performance claims are often missing or overstated [11]. Security scans further surface high-severity vulnerabilities in both model code and associated GitHub repositories [13]. Alarmingly, Taraghi et al. [22] find that adoption of Hugging Face's own documentation template has plateaued. In contrast to these existence-oriented studies, we further assess *correctness* of critical fields, enabling a quantitative quality assessment that helps developers identify trustworthy and reusable models.

Risk Analysis. Several works show that flaws in an upstream model can propagate and sometimes be magnified into its descendants [1, 4, 7, 14, 21, 24, 30]. Sun et al. [21] demonstrate that security risks introduced during PEFT are inherited by derived models; Chen et al. [4] document how fine-tuning can entrench social bias and generate unsafe text. Gekhman et al. [7] empirically show that injecting new factual knowledge increases hallucination rates linearly, while Kumar et al. [14] observe that jailbreak success rate often rises in post-fine-tuning. Model merging, as well, can become a vector for multi-task compromise [30]. However, these studies examine isolated operations in controlled settings. Our work complements them by charting risk propagation across the *full spectrum* of dependency relations at scale for three common risks.

3 Empirical Study Methodology

3.1 Study Design

All research questions are considered from two perspectives, i.e., models and the dependency relations that interconnect them.

RQ1: Usage Analysis. To characterize how models are reused in practice, we first enumerate all direct dependencies for models on Hugging Face, and label the transformation operation type for each dependency relation. As a derived model can itself become a base model, these dependency relations grow into chains of an arbitrary length and further coalesce into larger clusters. For every chain, we record its length from the initial base model to the final derived model; for every cluster, we record its total number of models and model chains. This end-to-end holistic analysis reveals which models and corresponding operations constitute the backbone of today's model supply chain, providing information that is essential for both model maintainers and consumers.

RQ2: Evolution Analysis. We track week-by-week additions, deletions, and updates between 1 March 2025 and 1 June 2025, a thirteen-week snapshot that captures a period of intense activity. Each week we log newly published models, permanently deleted models, and updates to existing ones (including models and metadata edits). We then analyze how these events reshape dependency relations, chains, and clusters over time. This time window exposes the stability of popular base models, the churn of derived models, offering a grounded view of maintenance intensity.

RQ3: Quality Analysis. We assess documentation quality by inspecting five critical metadata fields, i.e., declared license, performance, prompt template, dataset, and dependency relations for each sampled model. Beside quantifying field presence, we further verify the correctness of stated dependencies. Finally, we calculate the percentage of models with one or multiple incomplete or incorrect fields. Because incomplete or incorrect metadata obscures provenance and legal status, understanding its prevalence and the breadth of models it touches is vital for trustworthy reuse.

RQ4: Risk Analysis. We investigate how three real-world risk types, i.e., hallucination, jailbreak, and prompt injection, propagate through common transformations. Using established public benchmarks for each risk type, we evaluate every base model, retest its derivatives produced by fine-tuning, quantization, or merging, and compare the resulting score deltas. Knowing whether each operation makes a risk worse or better lets developers better understand the risks of model usage, and raises the design of further mitigation.

3.2 Model Supply Chain Construction

Our analysis is focused on public models hosted on Hugging Face, today's most widely adopted model-sharing platform, significantly larger in both user base and model count than alternatives such as PyTorch hub or TensorFlow hub. To build our model supply chain, we crawl the full model-card metadata for every public model on June 25, 2025, yielding a set of 1,817,398 models, denoted as $\mathcal{M}$.

Then, we extract dependency relations between models by parsing each model-card's metadata. Hugging Face supports four dependency relation (i.e., model transformation operation) types.

- *Quantize:* derive a new model by quantizing a base model;
- *Merge:* derive a new model by merging two or more base models;
- *Fine-Tune:* derive a new model by fine-tuning a base model;
- *Adapter:* derive a new model via a PEFT-style fine-tune, in which only adapter modules are trained.

Because adapter-based tuning is a specialized type of fine-tuning, we merge *Adapter* into *Fine-Tune*, yielding three primary operation types, i.e., *Quantize*, *Merge*, and *Fine-Tune*. Then, we define each dependency relation as a 3-tuple $d = \langle m_b, m_d, op \rangle$, where m_b denotes the base model, m_d denotes the derived model, and $op \in \{Quantize, Merge, Fine\text{-}Tune\}$. From our full metadata dump, we identify a set of 540,838 dependency relations, denoted as $\mathcal{D}$.

Models $\mathcal{M}$ and dependency relations $\mathcal{D}$ form the model supply chain, which underpins all stages of our empirical study.

4 RQ1: Usage Analysis

To systematically analyze how models are used in the model supply chain, we conduct the analysis at three different levels, i.e., dependency relations, dependency chains, and dependency clusters.

4.1 Dependency Relation Analysis

4.1.1 Definitions. We define $\mathcal{M}_b$ as the set of models that serve as base models, and $\mathcal{M}_d$ as the set of derived models. We further define $\mathcal{M}_r = \mathcal{M}_b \cup \mathcal{M}_d$ as the set of *relational models*, and $\mathcal{M}_i$ as the set of *isolated models* that are not involved in any dependency. Formally,

Table 1: Distribution and Downloads of Models

	$\mathcal{M}_b$	$\mathcal{M}_d$	$\mathcal{M}_r$	$\mathcal{M}_i$	$\mathcal{M}$
Size	49,648	540,838	566,022	1,251,376	1,817,398
Downloads	1.4×10^9	2.7×10^8	1.5×10^9	2.5×10^8	1.7×10^9

Table 2: Distribution of Model Tasks by Dependency Types

Relation	NLP	CV	Other	Total
Fine-Tune	**109,591** (57.8%)	63,721 (33.6%)	16,139 (8.6%)	189,451
Merge	**11,092** (94.0%)	551 (4.7%)	160 (1.3%)	11,803
Quantize	**31,855** (93.3%)	687 (2.0%)	1,615 (4.7%)	34,157

$$\mathcal{M}_b = \{\, m_b \in \mathcal{M} \mid \exists \langle m_b, m_d, op\rangle \in \mathcal{D}\}, \quad \mathcal{M}_r = \mathcal{M}_b \cup \mathcal{M}_d,$$
$$\mathcal{M}_d = \{\, m_d \in \mathcal{M} \mid \exists \langle m_b, m_d, op\rangle \in \mathcal{D}\}, \quad \mathcal{M}_i = \mathcal{M} \setminus \mathcal{M}_r.$$

Notice that $\mathcal{M}_b$ and $\mathcal{M}_d$ may overlap, as many models both derive from others and serve as bases for further derivatives.

Then, we classify $\mathcal{M}_d$ by their operation types as follows.

$$\mathcal{M}_f = \{\, m_d \in \mathcal{M}_d \mid \exists\, \langle m_b,\, m_d,\, \textit{Fine-Tune}\rangle \in \mathcal{D}\,\},$$
$$\mathcal{M}_m = \{\, m_d \in \mathcal{M}_d \mid \exists\, \langle m_b,\, m_d,\, \textit{Merge}\rangle \in \mathcal{D}\,\},$$
$$\mathcal{M}_q = \{\, m_d \in \mathcal{M}_d \mid \exists\, \langle m_b,\, m_d,\, \textit{Quantize}\rangle \in \mathcal{D}\,\}.$$

Finally, we use $\deg_{in}(m)$ to denote the *in-degree* of model m, i.e., the number of models that m depends on, and $\deg_{out}(m)$ to denote its *out-degree*, i.e., the number of models that depend on m. Formally,

$$\deg_{in}(m) = |\{m_b \in \mathcal{M} \mid \exists\, \langle m_b, m,\, op\rangle \in \mathcal{D}\}|,$$
$$\deg_{out}(m) = |\{m_d \in \mathcal{M} \mid \exists\, \langle m,\, m_d,\, op\rangle \in \mathcal{D}\}|.$$

4.1.2 Overview of Models. As shown in Table 1, 68.9% models are isolated (i.e., $\mathcal{M}_i$), only accounting for 2.5×10^8 (12%) downloads. However, relational models $\mathcal{M}_r$ account for 1.5×10^9 (88%) downloads with 1.4×10^9 (82%) from base models and 2.7×10^8 (15%) from derived models. This asymmetric usage of models highlights a long-tail phenomenon; i.e., a small set of connected models drives the majority of real-world model consumption.

4.1.3 Overview of Dependency Relations. Among $|\mathcal{M}_d| = 540,838$, $|\mathcal{M}_f| = 427,475$, $|\mathcal{M}_m| = 12,799$, and $|\mathcal{M}_q| = 100,564$. Thus, $\mathcal{M}_f$ dominates the derived models, accounting for approximately 79.0%, followed by $\mathcal{M}_q$ (18.6%) and a small portion for $\mathcal{M}_m$ (2.4%).

We further investigate the task distribution across different dependency relation types. On Hugging Face, a model's task is specified in its model-card via the `pipeline_tag` field. Hugging Face categorizes model tasks into over 50 fine-grained sub-tasks, which are grouped under six broader categories, i.e., *Natural Language Processing (NLP)*, *Computer Vision (CV)*, *Multimodal*, *Audio*, *Tabular* and *Reinforcement Learning*. As the latter four categories together account for only a small portion of all models (about 1%), we lump them together as *Other*. In total, 235,411 models provide valid task annotations. Among $\mathcal{M}_f$, 189,451 are annotated with tasks; among $\mathcal{M}_m$, 11,803 are annotated; and among $\mathcal{M}_q$, 34,157 are annotated.

As summarized in Table 2, models serving NLP tasks dominate across all three dependency types. In particular, 57.8% of annotated $\mathcal{M}_f$ models target NLP, while more than 90% of both annotated $\mathcal{M}_m$ and $\mathcal{M}_q$ models are also NLP-related. This NLP-centric skew heightens the risk of propagating textual biases at scale, and makes the entire ecosystem risky to a single supply chain exploit. It also leaves other modalities under indirect or multi-modality risks, potentially compromising the trustworthiness of multi-modal AI systems.

4.1.4 Top Model Analysis. We report the top-10 models with the highest in-degree and the highest out-degree in Table 3. We observe that 8 of the top-10 models with the highest in-degree are associated with NLP tasks. Similarly, 7 of the top-10 models with the highest out-degree are also primarily used in NLP applications. This indicates that NLP remains the dominant in model reusing.

High-degree models usually play pivotal yet distinct roles in the model supply chain. For example, *Qwen/Qwen1.5-0.5B*, which has the highest out-degree (32,497), is a compact yet versatile language model. As a *text-generation* model, its small size and strong general capabilities make it a popular base model for further development, especially for quantization, low-rank adaptation, and instruction tuning. On the other hand, *Novaciano/BAHAMUTH-PURGED-3.2-1B*, with the highest in-degree (65), is also a *text-generation* model, and serves as a highly merged artifact that integrates content from numerous upstream models. As noted in its model-card, it represents a complex amalgamation of the LLaMA 3.2 series, illustrating the prevalence of large language model merging in the current ecosystem.

Insight. Our analysis uncovers a comprehensive usage dependency in model supply chain. Base models are deriving mainly through fine-tuning for NLP tasks, which speeds up progress but makes it hard to trace and trust the final model artifact because risks can flow from the base model, the tuning code, and the training data, generating a wide attack surface. Maintainers should therefore focus monitoring and governance on these high-degree models. Researchers can help by building automated tools for model lineage tracking and license checking that understand the semantics of models.

4.2 Dependency Chain Analysis

4.2.1 Definition. Models may serve both as base models and as derived models, naturally giving rise to model chains, where each model is linked to the next through a directed dependency. Formally,

$$C = \{c := (m_0, \ldots, m_\ell) \mid \forall\, 0 \le i < \ell :\ \langle m_i, m_{i+1}, op_i\rangle \in \mathcal{D}\}.$$

Let $\text{head}(c) = m_0$ and $\text{tail}(c) = m_\ell$. A chain c iteratively constructs until no further descendant $m_\ell \in \mathcal{M}_b$.

4.2.2 Overview of Dependency Chains. The total number of dependency chains is $|C| = 1,008,871$. The average chain length is 6, suggesting that model dependencies tend to construct moderately deep dependency chains. In particular, 613,853 (60.8%) chains have a length greater than 2, indicating that most models are not merely derived from a single predecessor but participate in a long and layered sequence of dependencies. While a majority of chains are short, with 602,369 (59.7%) chains having a length of 5 or fewer, there is still a substantial presence of moderate chains. Specifically, 346,617 (34.4%) chains span 6 to 15 hops, and 59,885 (5.9%) chains exceed 15 hops. The detailed distribution of chain lengths is available at our website [16] due to space limitation. This distribution indicates not only the prevalence of simple derivations, but also the existence of complex multi-hop derivations in the Hugging Face ecosystem. Surprisingly, the longest chain from *CohereLabs/c4ai-command-r-v01* to *Citaman/command-r-1-layer* spans 40 hops, illustrating the potential depth and intricacy of model supply chain.

Table 3: Top-10 Models by In-Degree and Out-Degree

In-Degree			Out-Degree		
Model	**Task**	$\deg_{in}(m)$	**Model**	**Task**	$\deg_{out}(m)$
Novaciano/BAHAMUTH-PURGED-3.2-1B	text-generation	65	Qwen/Qwen1.5-0.5B	text-generation	32,497
Novaciano/BAHAMUTH-PURGED-3.2-1B-Q6_K-GGUF	–	65	black-forest-labs/FLUX.1-dev	text-to-image	32,173
QuantFactory/Loki-v2.6-8b-1024k-GGUF	–	57	Qwen/Qwen1.5-1.8B	text-generation	30,580
PJMixers-Archive/LLaMa-3-CursedStock-v1.6-8B	text-generation	52	google/gemma-2b	text-generation	23,792
PJMixers-Archive/LLaMa-3-CursedStock-v2.0-8B	text-generation	47	google/gemma-7b	text-generation	9,569
QuantFactory/L3-Deluxe-Scrambled-Eggs-On-Toast	text-generation	36	distilbert/distilbert-base-uncased	fill-mask	9,205
Casual-Autopsy/L3-Deluxe-Scrambled-Eggs-On-Toast	text-generation	36	stabilityai/stable-diffusion-xl-base-1.0	text-to-image	8,266
PJMixers-Archive/LLaMa-3-CursedStock-v1.8-8B	text-generation	34	Qwen/Qwen1.5-7B	text-generation	6,450
SanXM1/Driftwood-12B	text-generation	25	google-bert/bert-base-uncased	fill-mask	5,483
mergekit-community/L3.1-Athena-l-8B	text-generation	25	aubmindlab/bert-base-arabertv02	fill-mask	3,997

Table 4: Top-10 Chains with Their Operation Ratios

Root Model	Length	Operation Ratio		
		FT.	M.	Q.
CohereLabs/c4ai-command-r-v01	40	**100%**	0%	0%
vicgalle/Configurable-Llama-3-8B-v0.2	28	18.52%	**81.48%**	0%
recoilme/recoilme-gemma-2-9B-v0.2	23	0%	**100%**	0%
cgato/Nemo-12b-Humanize-KTO-Experimental-Latest	22	4.76%	**95.24%**	0v
fblgit/una-cybertron-7b-v2-bf16	21	30.00%	**65.00%**	5.00%
ericjiliangli/t5-small-news-summarization	21	**100%**	0%	0%
answerdotai/ModernBERT-Large-Instruct	21	**100%**	0%	0%
lukasdrg/clinical_longformer_same_tokens_180k	20	**100%**	0%	0%
berkeley-nest/Starling-LM-7B-alpha	19	**55.56%**	38.89%	5.56%
FelixChao/Sectumsempra-7B-DPO	18	**52.94%**	41.18%	5.88%

Insight. Our analysis reveals that fine-tuning is the dominant driver of deep dependency chains in the model supply chain. It aligns with earlier observations that fine-tuning is the most common dependency relation across all models. The repeated use of fine-tuning enables rapid reuse especially in NLP tasks. However, it also amplifies risk propagation along chains, as model behaviors and risks can be influenced by any upstream model or dataset. In addition, since the detailed fine-tuning technique is often opaque and poorly documented, this depth adds uncertainty to provenance and trust.

4.2.3 Operations in Dependency Chains. For each chain, we compute the relative ratio of the three operation types (i.e., *Fine-Tune*, *Merge*, and *Quantize*) as the number of occurrences of each operation type divided by the chain length. A chain is considered as *dominant* in an operation type if that operation's ratio is greater than the other two. Among the 1,008,871 chains, 45.4% are *finetune-dominant*, 48.5% are *merge-dominant*, and 6.1% are *quantize-dominant*. This indicates that while fine-tuning remains a significant driver of model reuse, merging is slightly more prevalent across the observed chains.

We also analyze the diversity of operations within each chain, defined as the number of distinct operation types present. We find that 45.9% chains contain a single operation type, 26.1% contain exactly two types, and 28.0% involve all three. That is to say, over half of the chains (i.e., 54.1%) involve more than one type of operations. This indicates that model dependency is often not limited to a single operation, but instead incorporates a diverse set of dependency relation types within the same chain. Such diversity reflects the heterogeneous ways in which models are reused in practice.

To further distinguish usage patterns across different stages of the model supply chain, we divide each dependency chain into three equal-length segments: *front* , *middle*, and *back*. We find that upstream models exhibit an average of **469,382 downloads** and in contrast, downstream models average only **1,029 downloads**. This indicates that upstream models tend to be more widely reused across the ecosystem, whereas downstream models are typically customized or deployment-oriented variants.

4.2.4 Top Dependency Chain Analysis. We analyze the ten longest chains in C. As shown in Table 4, we observe that the dominant operation of 6 of the top-10 chains is *Fine-Tune*. This indicates that deep chains are primarily driven by repeated fine-tuning steps. In terms of task distribution, we find that all of the models in these longest chains are associated with NLP tasks, suggesting that deep model reuse is most prevalent in the NLP domain.

4.3 Dependency Cluster Analysis

4.3.1 Definition. We define a *dependency cluster* $K_m = \{ c \in C \mid \text{head}(c) = m \in \mathcal{M}_b \setminus \mathcal{M}_d \}$ as the set of all the dependency chains rooted at the same base model m (i.e., a model with 0 in-degree). Conceptually, a cluster aggregates *all* transitive descendants of a root, giving a complete view of that model's supply tree. This perspective is crucial for model supply chain because the reusability and trustworthiness of m will propagate to *every* model in K_m. We denote the family of clusters by $\mathcal{K} = \{K_m \mid m \in \mathcal{M}_b \setminus \mathcal{M}_d\}$.

4.3.2 Overview of Dependency Clusters. In the model supply chain, we identify $|\mathcal{K}| = 25,184$ clusters. On average, each cluster contains 29 models and 40 dependency chains, indicating an almost one-to-one ratio of models to chains. Specifically, the majority of clusters are small. Clusters with 5 or fewer models account for 20,208 (80.2%) of all clusters, and clusters with 10 or fewer models account for 22,271 (88.4%). Nevertheless, a long tail exists. 117 clusters exceed 1,000 models, reflecting a few highly developed model families. The largest cluster contains 32,554 models. Since most clusters remain small, the dependency topology tends to be shallow and isolated, suggesting a lower maintenance and security risk for the majority of the ecosystem. The detailed distribution of cluster sizes is available at our website [16] due to space limitation.

4.3.3 Operations in Dependency Clusters. For each cluster, we measure the ratio of the three operation types, i.e., *Quantize*, *Merge*, and *Fine-Tune*, and label the operation type as *dominant* when its ratio exceeds those of the other two. Among all the clusters, 20,639 (82.0%) involve only one single operation type, which contain three models on average; 3,371 (13.4%) involve two operation types, which contain 35 models on average; and 1,174 (4.7%) exhibit all three types, which contain 466 models on average. In multi-type clusters, *Quantize* dominates more than half of the clusters (51.1% for dual-type, and 67.8% for triple-type), dwarfing *Fine-Tune* and especially *Merge*.

Table 5: Top-10 Clusters with Their Operation Ratios

Root Model	Size	Operation Ratio			Task
		FT.	M.	Q.	
Qwen/Qwen1.5-0.5B	32,554	**52.34%**	1.87%	45.79%	text-generation
black-forest-labs/FLUX.1-dev	32,325	**88.51%**	1.69%	9.79%	text-to-image
Qwen/Qwen1.5-1.8B	30,617	**55.74%**	3.28%	40.98%	text-generation
google/gemma-2b	23,872	**73.27%**	3.60%	23.12%	text-generation
meta-llama/Llama-3.1-8B	13,725	**51.23%**	7.42%	41.35%	text-generation
mistralai/Mistral-7B-v0.1	11,187	**48.46%**	18.45%	33.08%	text-generation
Qwen/Qwen2.5-7B	9,932	**57.10%**	6.39%	36.51%	text-generation
google/gemma-7b	9,871	**77.97%**	2.37%	19.66%	text-generation
distilbert/distilbert-base-uncased	9,510	**99.55%**	0.01%	0.43%	fill-mask
meta-llama/Meta-Llama-3-8B	9,261	**48.31%**	12.78%	38.90%	text-generation

4.3.4 Top Dependency Cluster Analysis. Table 5 lists the ten largest clusters. Collectively, they encompass 173,593 models and 174,315 chains, approximately 30.7% of all relational models but only 17.3% of chains, revealing a heavy-tailed size distribution. All ten clusters are dominated by *Fine-Tune*, and nine of the ten roots support NLP tasks. These clusters therefore serve as central adaptation families. Though most clusters are simple chains, a few have many derived models. These large clusters matter most; i.e., a problem in the root model can spread to thousands of others.

Insight. Dependency clusters uncover the topology of the model supply chain. Most roots spawn only a handful of descendants, but a small number of NLP-centric base models underpin nearly one-third of all released models. As clusters grow, they diversify operation types and become increasingly inducing broader attack surfaces. Supply chain governance must therefore focus on (i) keeping continuous watch on these root models in large clusters and (ii) adding rigorous checks to every operation step because mistakes there can also propagate to hundreds of downstream models. Applying assurance and security tooling on those two key points offers the highest return for protecting the entire model supply chain.

5 RQ2: Evolution Analysis

To capture fine-grained yet tractable temporal dynamics, we record a snapshot of Hugging Face once per week on *Wednesday*, which is empirically the platform's lowest-activity day, minimising short-lived release bursts. Our window spans 17 weeks from 2025-03-12 (week 11) to 2025-07-02 (week 27), and follows the four-month horizon suggested by prior software supply chain studies [25].

5.1 Addition and Deletion Analysis

5.1.1 Definition. For the week index $t \in \{1, \ldots, 17\}$, we denote the snapshot as a tuple $\mathcal{S}^t$ as follows,

$$\mathcal{S}^t = \langle \mathcal{M}^t, \mathcal{M}_b^t, \mathcal{M}_d^t, \mathcal{M}_r^t, \mathcal{M}_i^t, \mathcal{D}^t \rangle,$$

where the six components respectively capture all models, base models, derived models, relational models, isolated models, and dependency relations observed on Wednesday of week t.

Addition $\mathcal{S}^{t+}$ and deletion $\mathcal{S}^{t-}$ are then defined as set differences between consecutive snapshots as follows.

$$\mathcal{S}^{t+} = \mathcal{S}^t \setminus \mathcal{S}^{t-1}, \quad \mathcal{S}^{t-} = \mathcal{S}^{t-1} \setminus \mathcal{S}^t.$$

We say a *new chain* c^{t+} or a *new cluster* K_m^{t+} is added when *all* its constituent models appear in $\mathcal{M}^{t+}$. Formally,

$$C^{t+} = \left\{ c^{t+} := (m_0, \ldots, m_\ell) \,\middle|\, \forall\, 0 \le i < \ell : \langle m_i, m_{i+1}, op_i \rangle \in \mathcal{D}^{t+}, m_i, m_{i+1} \in \mathcal{M}^{t+} \right\},$$

$$\mathcal{K}^{t+} = \left\{ K_m^{t+} := \left\{ c^{t+} \in C^{t+} \mid \mathrm{head}(c^{t+}) = m \right\} \mid m \in \mathcal{M}_b^{t+} \setminus \mathcal{M}_d^{t+} \right\}.$$

Table 6: Top-10 Clusters in First Week vs. in Final Week

Top-10 Clusters in First Week	Size	Top-10 Clusters in Final Week	Size
Qwen/Qwen1.5-0.5B	32,535	black-forest-labs/FLUX.1-dev ↑3	32,796
Qwen/Qwen1.5-1.8B	30,653	Qwen/Qwen1.5-0.5B ↓1	32,560
google/gemma-2b	23,896	Qwen/Qwen1.5-1.8B ↓1	30,621
black-forest-labs/FLUX.1-dev	22,725	google/gemma-2b ↓1	23,885
meta-llama/Llama-3.1-8B	10,919	meta-llama/Llama-3.1-8B	13,875
mistralai/Mistral-7B-v0.1	10,288	mistralai/Mistral-7B-v0.1	11,221
google/gemma-7b	9,754	Qwen/Qwen2.5-7B ↑new	9,999
meta-llama/Meta-Llama-3-8B	8,710	google/gemma-7b ↓1	9,871
distilbert/distilbert-base-uncased	8,285	distilbert/distilbert-base-uncased	9,573
stabilityai/stable-diffusion-xl-base-1.0	7,129	meta-llama/Meta-Llama-3-8B ↓2	9,299

Similarly, we define the set of deleted chains C^{t-} and deleted clusters $\mathcal{K}^{t-}$, which consist of each dependency chain and cluster whose models are all deleted between snapshots t and $t-1$.

5.1.2 Addition and Deletion of Models. Across the 17 snapshots, the average number of weekly model addition is 27,311. Of these, 19,687 (72.1%) are *isolated models*, while 7,624 (27.9%) are *relational models*. Among such relational models, only 192 (2.5%) are *base models*; the vast majority (7,432, 97.5%) are *derived models*. A closer look at $\mathcal{M}_d^{t+}$ shows that *Fine-Tune* dominates, contributing 5,413 (72.8%) models per week, whereas *Merge* and *Quantize* only contribute 222 and 1,797 models respectively. Conversely, an average of 6,190 models are deleted per week. Of these, 4,367 (70.5%) are *isolated models*, and 1,823 (29.5%) are *relational models*. Among such relational models, only 127 (7.8%) are *base models*, and 1,771 (97.1%) are *derived models*. Among such derived models, 1,567 (88.5%) are fine-tuned ones, significantly exceeding merged (56) and quantized (148) ones.

5.1.3 Addition and Deletion of Chains and Clusters. Of the 7,624 relational models added per weak, 7,082 models are appended to pre-existing chains, while 538 new chains C^{t+} and 191 new clusters $\mathcal{K}^{t+}$ are created. For deletions, 32 chains and 26 clusters are completely deleted per week on average, which is relatively small in quantity compared to the added ones. However, rather than the complete deletion of chains and clusters, partial deletions of chains and clusters are more frequent, and disrupt the existing topological structure of them. Among the 6,190 models deleted weekly, 127 serve as base models, directly affecting 299 derived models, and per base model affects about 2 derived models on average. These base model deletions disrupt 1,440 chains and 78 clusters, leaving 528 downstream models untraceable. While the number of disrupted clusters is substantially smaller than that of disrupted chains, the severity of disruption at the cluster level is much higher. On average, only one downstream model becomes untraceable in each disrupted chain, whereas seven downstream models become untraceable in each disrupted cluster, reflecting a more extensive propagation of deletions within clusters. Such deletions sever both direct and indirect dependencies on the deleted bases, leaving entire chains of derived models without their upstream models and at risk of becoming untraceable or unusable.

5.1.4 Changes of Top Chains and Clusters. The ranking of the ten longest chains remains unchanged over our four-month period. By contrast, as shown in Table 6, the ten largest clusters stay active and evolve constantly. 1,385 (18.2%) of all new relational models each week join these top clusters, yet only 238 (9.3%) deletions occur within them, underscoring their central role. This "rich-get-richer" pattern of chains and clusters mirrors findings in traditional software package ecosystems [6]. Although the overall composition of the top clusters persists, individual ranks do shift as clusters grow at different rates and new popular models release. For example,

the *Qwen/Qwen1.5-0.5B* cluster slips from rank 1 to rank 2, while *Qwen/Qwen2.5-7B*, released on March 27, 2025 as a direct successor to the 1.5 series, forms a cluster whose rank raises to 7 at final week.

Insight. Model additions outnumber deletions, yet each deletion has outsized impact on all the downstream models. Thus every model becomes a potential single risk point, and could widely affect the whole model supply chain. Developers should mirror critical root models locally (license permitting) to avoid sudden chain disruptions, while researchers can build early-warning dashboards, similar to those in classical package registries, to keep model supply chain transparent and resilient.

5.2 Update Analysis

We classify a model as *updated* when its commit log shows evidence of substantive change in at least one week at the time window. Concretely, a repository is marked updated if

- it contains at least one commit after initial uploads;
- the commit is not mere documentation update (e.g., `README.md`);
- the commit title includes at least one keyword signaling version, file, or bug-fix intent, e.g., `update`, `checkpoint`, or `fix`.

These heuristics rules can capture silent but meaningful update that would be otherwise overlooked by model-card-level metadata [3].

145,445 (8.0%) models show at least one non-trivial post-release commit. Among the updated set, 99,168 (68.2%) are *isolated* models, and 46,277 (31.8%) are *relational*. Within relational updates, 13,666 (29.5%) are base models, while 37,685 (81.4%) are derived models. In derived models, those derived via *Fine-Tune* dominate updates with 30,872 (81.9%), significantly exceeding *Quantize* (5,704, 15.1%) and *Merge* (1,109, 2.9%). Despite Hugging Face platform natively support for version control in model-card, only 1,899 (0.1 %) repositories update the `newer_version` card field. The overwhelming majority of models therefore ship as single-version models with no transparent changelog. For consumers, the practical burden shifts from "*is my dependency up to date?*" to "*does a newer version even exist, and how would I know?*", a non-trivial risk for reproducibility, compliance, and risk scanning.

Insight. About one in ten models evolves in 17 weeks. When they update, the changes are rarely surfaced through official metadata. Effective tools must thus bridge this observability gap, automating update discovery and propagating upgrade deltas to downstream users to avoid outdated model usage.

6 RQ3: Quality Analysis

As metadata fields vary widely across different model domains [8], a unified quality assessment would be challenging. Instead, we restrict our analysis to *text-generation* models. These models not only comprise the largest part of the model supply chain (see Sec. 4 and 5), but also play a pivotal role in today's model supply chain, making any quality problems particularly consequential.

Field Selection. We study five orthogonal fields that affect the security, safety and reusability of a model; i.e., *License*, *Performance*, *Template*, *Dataset*, and *Dependencies*. Prior work shows that missing *License*, *Performance*, or *Dataset* field can hide weak models, propagate bias, or trigger legal trouble [13, 20, 29]. The *Template* field is unique to text-generation models. Without the template, users may mis-prompt the model, leading to low performance usage or failing to defend against prompt attacks like prompt injection. Finally, the *Dependencies* field records whether a model is merged, quantized, or fine-tuned from another model, yet earlier studies omit it. We extract these five fields from each model's model-card.

Model Sampling. We first keep only models labeled with the *text-generation* task. To focus on models that real-world users touch, we keep the top 10% of them by downloads, resulting in 24,686 candidates. We statistically sample 378 models to have a 95% confidence level with a 5% margin of error, denoted as $\mathcal{M}_{sample}$.

6.1 Quality Analysis of Models

Metrics. For each model in $\mathcal{M}_{sample}$, we mark a field as *missing* if the entry is empty or absent. We report the missing rate (*MR*) for fields *License*, *Performance*, *Template*, and *Dataset*.

Findings. *License* is absent for 28.3% of all sampled models, leaving downstream users unsure if they can ship the model in a product. *Performance* is the worst, and 82.8% of all sampled models miss this field, and thus users are not sure if a model will meet their accuracy or latency requirements. Not all *text-generation* models require a template. For example, models with intuitive token-by-token completion or those primarily used via high-level APIs may not publish explicit prompts. However, among models that do support prompt templates, *Template* field is not provided for 33.8% (105/311) models. Similarly, not all *text-generation* models require a dataset. Models built via *Merges* or *Quantize* often derive from base models without retraining on a new dataset. However, *Dataset* fields are still absent in 76.8% (285/371) of data-needed models. For example, among models marked as *fine-tune* in $\mathcal{M}_{sample}$, only 11 of 41 include a dataset reference. Missing dataset hampers reproducibility, and makes it impossible to audit for bias or privacy leaks introduced by dataset.

Insight. These missing-related quality problems force users to incur extra costs, hiring lawyers to untangle license terms, running their own benchmarks to verify performance, writing custom deployment scripts, or manually tracing data provenance, which slows time-to-market, inflates budgets, and breaks any "free" or "open-source" cost advantage.

6.2 Quality Analysis of Dependency Relations

Metrics. We measure (i) the fraction of all models that the tagged dependency relations are actually correct, denoted as precision, and (ii) the fraction of all truly derived models that are tagged in the model-card, denoted as recall. Two authors with over five years experience of deep learning development independently reviewed the extracted dependency relations, drew on official online documentation for each model (e.g., developer websites, GitHub repositories) as well as their domain expertise, and judged whether each relation was correct. Any disagreements were resolved by a third author, resulting in a Cohen's Kappa [26] of 0.921.

Findings. Of the 378 sampled models, 201 (53.2%) do not declare any dependency. Our manual review shows that 79 of them in fact are derived from another model; i.e., about one-third of truly derived models lack a recorded dependency, driving the recall down to 68.5%. Conversely, when a dependency is declared for 177 models, it is

almost always correct; i.e., only 5 dependency relations are wrong (all in *Fine-Tune* operations), yielding a precision of 97.2%. In other words, developers who do fill in *base model* are more reliable (high precision), but many derived models go untagged (low recall).

The reasons why model dependency is low quality are two folds. First, developers may simply overlook the field; e.g., the *google/gemma-2b-it* model omits a link to its fine-tuned base. Second, the current Hugging Face schema offers only a few operation types, and thus developers either cannot tag certain relations or end up mislabeling them. Take *Qdrant/bge-small-en-v1.5-onnx-Q* as an example. It is just a quantized ONNX export of *BAAI/bge-small-en-v1.5*, but because there is no "format-transformed" tag in the schema, it cannot be marked correctly. Likewise, *mlx-community/Qwen3-0.6B* is a format conversion of *Qwen/Qwen3-0.6B*, yet it is (incorrectly) tagged as a fine-tuning relation. These gaps break supply chain traceability, and hide how changes to an upstream model could transfer into downstream by a specific transformation operation.

Overall, of the 378 sampled models, 363 (96.0%) are missing or incorrect in at least one of the five key fields; 303 (80.2%) are missing or incorrect in at least two fields; 172 (45.5%) are missing or incorrect in at least three fields; 54 (14.3%) are missing or incorrect in at least four fields; and 2 (0.5%) are missing or incorrect in all five fields, highlighting the severity of quality problems in model supply chain.

Insight. Dependency fields are often missing for relational models. Adding comprehensive and accurate dependency tags (e.g., *Pruning*, *Convert* or *Distillation*) would let users audit lineage, watch for upstream security fixes, and comply with emerging AI provenance rules. Consequently, high-quality metadata is not optional, as it is foundational to safe reuse and responsible deployment. Model maintainers must take greater responsibility in documenting training sources, dependency relations, and licensing terms. The Hugging Face platform should also enforce stricter metadata requirements and expand its dependency schema to reflect real-world model operations.

7 RQ4: Risk Analysis

Building on the risk taxonomy of Cui et al. [5], which categorizes risks into performance, security and safety dimensions, we adopt three main metrics from three benchmarks, respectively, i.e., *Hallucination Rate (HR)*, *Jailbreak Success Rate (JSR)* and *Prompt Injection Success Rate (PSR)*, to comprehensively quantify the propagation of common risks along dependency relations. Each metric is between 0 (no observed risk) and 1 (maximum observed risk).

Hallucination Rate Evaluation. We evaluate a model's tendency of hallucination using HHEM-2.1-Open [23], a state-of-the-art benchmark built on the CNN/Daily Mail corpus. For each test passage, HHEM-2.1-Open provides a binary judgment; i.e., *faithful* or *hallucinatory*. We define *Hallucination Rate* (HR) as $\frac{C_h}{N_h}$, where C_h is the number of faithful passages, and N_h =10,000 is the total number of evaluated passages. HR ranges from 0 to 1, with a higher value denoting a greater propensity for hallucination.

Jailbreak Success Rate Evaluation. To evaluate the jailbreak attack risk, we utilize AdvBench [33], a state-of-the-art benchmark designed to assess model robustness and safety alignment in adversarial and safety-critical scenarios. AdvBench includes 520 carefully curated adversarial test cases, encompassing both harmful strings and malicious behavior instructions related to topics such as violence and cybercrime. For each test case, this benchmark provides a binary judgment for the models, i.e., *jailbreak* or *not jailbreak*. We quantify jailbreak risk via the *Jailbreak Success Rate* (JSR), defined as $\frac{C_j}{N_j}$, where C_j is the number of successful jailbreaks, and $N_j = 520$ is the total number of test cases. JSR ranges from 0 to 1, with a higher value indicating a greater risk of jailbreak.

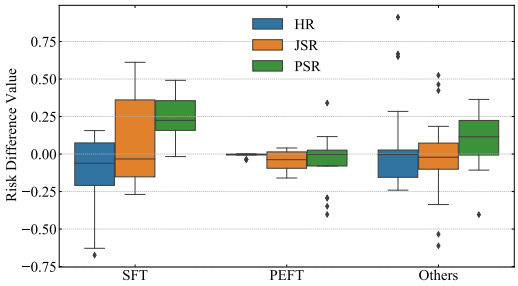

Figure 1: Risk Propagation Along *Fine-Tune*

Prompt Injection Success Rate Evaluation. To evaluate the prompt injection risk, we utilize the state-of-the-art benchmark curated by Liu et al. [15], a comprehensive suite of injection scenarios spanning seven common NLP tasks. This benchmark has 4,900 test cases. For each test case, it provides a binary judgment for the models, i.e., *injected* or *not injected*. We quantify injection risk via the *Prompt Injection Success Rate* (PSR), defined as $\frac{C_i}{N_i}$, where C_i denotes the number of successful prompt injections, and N_i = 4,900 is the total number of test cases. PSR ranges from 0 to 1, with a higher value indicating a greater exposure to prompt injection risk.

7.1 Risk Analysis of Fine-Tune

Model Selection. We first selected the top 100 *Fine-Tune* dependency relations whose base and derived models have the highest combined number of downloads. We then excluded any dependency relation whose base or derived model lacks the template, reducing the set to 75 relations. Next, we partitioned these 75 relations into the following three fine-tuning categories.

- **Supervised Fine-Tuning (SFT):** All model parameters are updated on a labeled dataset, which accounts for 19 relations.
- **Parameter-Efficient Fine-Tuning (PEFT):** Only a small subset of parameters (e.g., adapters, LoRA modules) are trained, while the majority remain frozen, which accounts for 16 relations.
- **Others:** Less common or unclassified fine-tuning operations, e.g., reinforcement learning (RL), RLHF, DPO, or operations with incomplete metadata, which accounts for a total of 40 relations.

Finally, for each of these 75 dependency relations, we ran its base and derived models to compute risk metrics, and computed the risk differences Δ from the derived to the base to quantify the risk propagation along fine-tuning. We plotted box plots to show the distribution of risk propagation across different fine-tuning categories. In box plots, Q2 captures the median of the risk difference distribution, and represents the central tendency of risk propagation.

Findings. As illustrated by the box plots in Fig. 1, there is distinct risk propagation tendency across fine-tuning categories. Specifically, SFT produces the largest and most inconsistent risk propagation. ΔHR ranges from -0.21 to 0.07, with a Q2 of -0.06, indicating

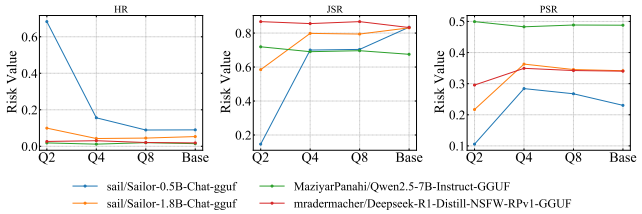

Figure 2: Risk Propagation Along *Quantize*

that hallucination is mostly reduced after fine-tuning. ΔJSR ranges from -0.15 to 0.36, with a Q2 of -0.03, indicating no consistent jailbreak risk propagation pattern. ΔPSR uniformly increases, ranging from 0.16 to 0.36 with a Q2 of 0.22, signaling a pronounced tendency to amplify the prompt injection risk via fine-tuning.

In contrast, PEFT show almost no deviation in hallucination, jailbreak, or prompt injection risks. Specifically, ΔHR from the base to the derived models group tightly around zero (i.e., ranging from -0.01 to 0.00 with a Q2 of 0.00); ΔJSR spans from -0.09 to 0.01 with a Q2 of -0.04; and ΔPSR spans from -0.08 to 0.03 with a Q2 of 0.00. However, there are more outliers in prompt injection, indicating that injection can occasionally jump or drop significantly.

Finally, less common or unclassified operations ("Others") show moderate shifts with a similar propagation pattern to SFT. The Q2 of ΔHR, ΔJSR and ΔPSR are respectively 0.00, -0.02 and 0.11 with a handful of outliers, underscoring that unconventional fine-tuning operations could unpredictably shift a model's risk profile.

> ***Insight.*** The dependency relation shapes how far a derived model diverges in risks from its base model. Developers who fine-tune models should balance accuracy against risks. PEFT costs little compute, and changes hallucination, jailbreak, and prompt injection behaviors only marginally. SFT boosts downstream accuracy, but it often amplifies jailbreak, and prompt injection risks by *up to 50%*. In practice, when stability, safety, and security are top priorities, PEFT is the preferred strategy. Conversely, if performance is paramount, one may choose for SFT, but must mitigate the amplified risks it introduces.

7.2 Risk Analysis of Quantize

Model Selection. Since the downloads of quantized models are highly correlated with their base models, we first identified the top 10 *text-generation* clusters. As vendors may quantize a base model using different bit-widths, we selected representative models from each quantization group by choosing the most downloaded quantized models with 2-bit, 4-bit, and 8-bit quantization from the top 10 clusters. Then, we excluded the quantized models that do not provide prompt templates, as their absence could introduce bias. Finally, four quantization groups were evaluated, each of which contained a base model and three quantized models at 2-bit, 4-bit, and 8-bit.

Findings. Fig. 2 illustrates the risk propagation as the bit-width decreases from FP16 in the base models. Specifically, for hallucination, there is generally an increase in risk as the bit-width decreases for most models, especially for the two *Sailer* models. For jailbreak and prompt injection, they share a similar trend; i.e., there is often a sudden risk drop when the quantization is ran at 2-bit, especially for the two *Sailer* models. Moreover, the 8-bit and 4-bit quantizeed models mostly maintain similar risks to their base models. Besides,

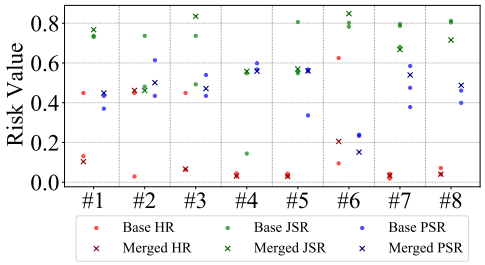

Figure 3: Risk Propagation Along *Merge*

for the base models whose size is moderate (e.g., the *Qwen2.5* model is 7B, and the *Deepseek-R1-Distill* model is 8B), their quantization usually will not change the risk significantly. It is the opposite for the base models whose size is small (e.g., the two *Sailor* models).

> ***Insight.*** Low-bit quantized models (e.g., 2-bit and 4-bit) offer space savings with the potential hallucination risk amplifying. Vendors must balance efficiency with reliability, giving clear guidance on the risks of low-bit models. Downstream developers need to carefully consider these trade-offs, especially in resource-constrained or safety-critical applications, to ensure the right balance of cost, performance, and security.

7.3 Risk Analysis of Merge

Model Selection. Similar to the analysis of *Quantize*, since the downloads of merged models are highly correlated with their base models, we first identified the top 10 *text-generation* clusters. We then selected representative models from each merging group (i.e., a model merged from two or more base models) by choosing the most downloaded merged model from the top 10 clusters. To broaden our evaluation, we also included five merged models (and their base counterparts) drawn from the well-known *Qwen2.5* series. Finally, eight merging groups were selected with their templates provided, which are hereafter referred to by IDs #1-8, corresponding to the following derived models: Qwen2.5-7B-Matrix (derived from 2 base models), Qwen2.5-7B-Gordion-v0.1-Reason (derived from 2 base models), Qwen2.5-3B-Model-Stock-v3.1 (derived from 2 base models), Qwen2.5-3B-RP-Thinker-V2 (derived from 3 base models), Reyna-Mini-1.8B-v0.2-Qwen1.5-1.8B-Merged-Slerp (derived from 2 base models), Beyonder-4x7B-v3 (derived from 3 base models), and Qwen2.5-7B-Qandora-CySec (derived from 2 base models).

Findings. As shown in Fig. 3, across all eight merging groups, the risks of merged models rarely land near the arithmetic mean of those of their bases. Instead, they snap to one extreme or the other, either matching the lowest-risk base or inheriting the highest-risk base's behavior. For example, the hallucination rate of the merged models in group #1 and #3 outperforms that of their worst base models and approaches that of their best base models, whereas the hallucination rate of the merged model in group #2 nearly mirrors that of its worst base. Taken together, these swings show that the risk of the merged models is not an averagely combination of base models. Merging can inherit one base's risk or inheriting its strength, which produces profiles that are sometimes more secure, sometimes more risky, but seldom merely the average of their risks. One possible reason is that merging stacks the full weight tendency of each base model, dropping the result into a highly non-linear zone of the loss

landscape. In that zone, whichever base models' features sit closest to a decision boundary (e.g., for jailbreak tokens or hallucination cues) can overpower the rest. The outcome is therefore "winner-takes-all" on some risks and "loser-drags-all" on others.

Insight. Merging is a high-variance move; i.e., the result often inherits *either* the most risky *or* the least risky of its bases. If developers already know all bases' profiles, the merged outcome is still manageable, but benchmarking only one base can only give little information to predict the other, highlighting that risk evaluation for upstream models remains essential.

8 Implications and Threats

Public models on Hugging Face now form a rapidly evolving supply chain whose behavior cannot be reasoned about by considering individual models in isolation, and is also different from the traditional software supply chain. Among 1.82 million models, Table 1 and 2 reveal that fewer than one-third participate in explicit dependency relations, yet those *relational* models drive more than 88% of all downloads, with *Fine-Tune* relations dominating 79%. Long chains (see Table 4) and wide clusters (see Table 5 and 6) concentrate risk in a handful of NLP-centric roots, while metadata gaps and transformation choices (see Fig. 1-3) directly translate into provenance blindspots and risk propagation. Below, we distill what these findings mean for four key stakeholders and outline concrete future works, and discuss the threats to validity of our study.

8.1 Upstream Model Maintainers

Maintainers of high out-degree base models (see Table 3), are now stewards of entire *supply trees*, not just single model. First, our chain analysis (see Table 4) shows that a single fine-tune can propagate through up to 40 downstream hops. Second, 28% of top-downloaded *text-generation* models ship without a licence (see Sec. 6.1), placing all descendants in a potential legal limbo.

The future work for maintainers therefore lies in enhanced transparency and provenance: (i) sign and version each checkpoint and publish verifiable training recipes, allowing downstream consumers to authenticate model provenance, reproduce training steps exactly, and roll back to known-good versions when under supply chain risks; (ii) maintain a structural BOM listing the base model, performance, license, dataset, risk metrics, etc. so that automated tools can enforce license compliance, reconstruct dependency graphs, and flag risky components; and (iii) implement continuous evaluation pipelines that, upon every model update, automatically re-run a comprehensive suite of risk benchmarks, quantifying performance risks (e.g., hallucination) and safety and security risks (e.g., jailbreak and prompt injection), and maintain an up-to-date risk dashboard. By maintaining a live, quantitative record of performance, safety and security risks, base models become verifiable, auditable, trustworthy anchors for the entire model supply chain.

8.2 Downstream Model Consumers

Consumers who derive new models via fine-tuning, merging or quantization must recognize that each adaptation technique represents a unique trade-off between performance and safety/security risks. Parameter-efficient fine-tuning (PEFT) minimizes computational cost and induces only minor changes in hallucination and jailbreak metrics, whereas full fine-tuning (SFT) can improve downstream task accuracy at the expense of widening the inter-quartile range of those risks by up to 50 % (see Fig. 1). Merging operations often snap derived models to either the least risky or the most risky of their base models (see Fig. 3). *Quantization* operations can match the base's accuracy while overly aggressive bit-width reductions can degrade task performance and open new safety/security gaps.

The immediate call to action is three-fold. (i) Understanding model provenance and supply chains can help to select the most suitable base models and mitigate risks effectively. (ii) Transformation operation choice should be treated as a design decision along with vendors, architecture or hyperparameters; and risk propagation evaluation should be conducted in CI before release. (iii) When publishing derived models, the full dependency graph should be populated, including operations like pruning, distillation, or ONNX conversion to preserve traceability. Beyond these calls, teams can embed lightweight mitigation modules that are tailored to the risks, thereby counteracting the specific performance, safety or security risks.

8.3 Hugging Face Platform Designer

27,311 new models are registered each week in Hugging Face platform (see Sec. 5.1.2), yet only 0.1% of the updated models enable the version-tracking field in model-card (see Sec. 5.2). Although deletions are less common, they still remove base models and root models, orphaning about 528 downstream dependent models every week. Hugging Face is therefore uniquely positioned to enforce model supply chain resilience and prevent these disruptions.

A short-term roadmap should extend the model-card schema: (i) accept a richer and fine-grained taxonomy of operations, e.g., *prune*, *distill*, and *convert*, which will enable precise recording of every transformation operation applied; (ii) require both a license hash and a base model checksum on all relational updates to guarantee license compliance and ensure content integrity; and (iii) provide performance, dependency, and risk metrics, e.g., median JSR alongside downloads so that consumers have immediate, transparent insight into a model's reusability and risks. In the longer term, Hugging Face could introduce *dependency-locking* services, producing immutable snapshots of complete model dependency graphs like `npm shrinkwrap` that protect consumers from breaking changes when a base model is deleted or silently updated.

8.4 Model Supply Chain Researchers

Researchers now have access to a comprehensive, up-to-date view of the model supply chain. Table 5 and 6 confirm a heavily-tailed clustering of models where a few base models underpin most descendants, highlighting that changes to these root or base models can have outsized impact. Additionally, Fig. 1-3 demonstrate that transformations (e.g., merging) can sometimes unpredictably raise or lower performance, safety, or security risks.

Future research direction should proceed in four areas: (i) *Dependency Relation Identification.* Combine model weight and architecture comparisons with behavioral testing to uncover unreported dependencies and recover the roughly 20% of missing dependencies. (ii) *Governance Metric Definition.* Define standardized metrics such as "criticality score" that combines model centrality (see Table 3)

with potential risk shifts, and "risk score" like CVSS to help platform hosts, maintainers, and consumers prioritize which models need rigorous oversight. (iii) *Model Risk Knowledge Database Construction.* Establish a curated risk database (like CVE database) that catalogs known security vulnerabilities (e.g., jailbreak and injection exploits) and performance regressions (e.g., hallucination), along with their dependency contexts and severity ratings, enabling timely, automated alerts, and security-driven model updates. (iv) *Risk Propagation Modeling.* Based on the knowledge database, model and analyze how the risk changed in an upstream model will propagate through dependency graphs of varying depth and structure.

8.5 Insights from Stakeholders

To validate and extend our implications, we conducted short semi-structured interviews with three practitioners: two industrial model developers who train, maintain, and deploy large models at major technology companies, and one academic researcher specializing in AI model supply chains. Each participant has over four years of experience in deep learning and large language models and we integrated cross-stakeholder reflections from the three interviews.

Upstream Model Maintainers. The practitioners emphasized key practical considerations for implementing recommendations on checkpoint signing, version control, and maintaining bills of materials (BOMs). These practices would significantly enhance traceability and transparency within the model supply chain. However, adopting them requires cost-effective strategies that align with organizational goals. While creating detailed training recipes and structured BOMs demands additional effort, the benefits particularly in terms of improving transparency and boosting model trustworthiness, could provide a competitive advantage and increase market appeal.

Downstream Model Consumers. Both practitioners highlighted that large-scale model risk management is struggling to keep up with the rapid adoption of LLMs. A key challenge is the lack of comprehensive performance and risk metadata, which complicates the evaluation of base model risks. The high cost of benchmark testing to ensure models are free from biases or security vulnerabilities makes it unaffordable for most consumers. While automated risk assessment tools could offer significant benefits, their adoption is hindered by cost concerns and the lack of immediate incentives.

Model Supply-Chain Researchers. The academic expert pointed out that researchers can play a central role in addressing these challenges by developing open tools for risk benchmarking, automated dependency tracking, and license-compliance checks. Collaboration between researchers, maintainers, and consumers could promote transparency and resilience in the model supply chain, ensuring that risks such as hallucination and prompt-injection are systematically monitored and mitigated.

Suggestion for Hugging Face Platform Designer. The absence of a clear dependency relations between models represents a significant gap that the platform could address and it is essential to develop a more detailed and comprehensive model card that encourages users to provide additional information.

8.6 Threats to Validity

Incomplete or Incorrect Model Relations. Our extraction of model dependency relations relies entirely on developer-supplied metadata in model-cards, which often omit critical fields. In a manual audit of 378 sampled *text-generation* models, we found that only 177 models declare any upstream dependency with a precision of 97.2%, while about one-third of truly derived models go unrecorded, yielding a recall of only 68.5%. Such incomplete or incorrect relation data can bias our RQ1 usage analysis and RQ2 evolution analysis by underestimating the number of relational models, the depth of dependency chains, and the size of clusters, while exaggerating the apparent impact of fine-tuning. Moreover, for RQ4 risk analysis, missing or misclassified hops may skew our understanding of how risks actually propagate through the supply chain. To mitigate it, we correct any detected relation errors before our risk analysis.

One-Hop Risk Propagation Analysis. In RQ4, we quantify risk shifts only across direct (one-hop) dependency relations by comparing each base model to its immediate derivatives on risk metrics. However, model supply chains on Hugging Face form moderately deep chains (an average length of 6) and can span up to 40 hops in extreme cases. To further examine whether the one-hop risk propagation patterns hold in multi-hop scenarios, we conducted a multi-hop analysis on six representative model chains from the most popular clusters. The result demonstrates that the one-hop risk propagation patterns hold in most of the multi-hop scenarios. All the risk propagation results of the six representative model chains are available at our website [16]. By focusing solely on single-hop propagation, we may overlook cumulative effects of the risk propagation. Addressing this limitation in the future works will require extending our benchmarks to capture multi-hop propagation dynamics throughout the full depth of the model supply chain. Moreover, our risk analysis draws on samples of limited dependency relations; expanding it to include a broader range of dependency relations would support richer and more generalizable insights.

9 Conclusions

Public models have become living, shared artifacts that grow and change. A small number of base models now support most of the work done on Hugging Face, and even tiny changes like lowering precision or merging weights can make a model much safer or much riskier. In this work, we conduct the first systematic study to characterize the usage, evolution, quality, and risk of this model supply chain. To keep this ecosystem healthy, everyone should work together; i.e., maintainers need to publish clear versioned models and track every change; consumers must test for risks and list all steps they take; platform designers should make it easy to see a model's history, license, and risk metrics; and researchers should build tools to recover missing links and model how risk propagates. Full evaluation details are available at our website [16].

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62332005 and 62372114).

References

[1] Suriya Ganesh Ayyamperumal and Limin Ge. 2024. Current state of LLM Risks and AI Guardrails. *arXiv preprint arXiv:2406.12934* (2024).

[2] Joel Castaño, Rafael Cabañas, Antonio Salmerón, David Lo, and Silverio Martínez-Fernández. 2024. How do machine learning models change? *arXiv preprint arXiv:2411.09645* (2024).

[3] Joel Castaño, Silverio Martínez-Fernández, Xavier Franch, and Justus Bogner. 2024. Analyzing the evolution and maintenance of ml models on hugging face. In *Proceedings of the 21st International Conference on Mining Software Repositories*. 607–618.

[4] Pin-Yu Chen and Sijia Liu. 2025. Safety Risks in Fine-Tuning Large Language Models. In *Introduction to Foundation Models*. Springer, 185–194.

[5] Tianyu Cui, Yanling Wang, Chuanpu Fu, Yong Xiao, Sijia Li, Xinhao Deng, Yunpeng Liu, Qinglin Zhang, Ziyi Qiu, Peiyang Li, et al. 2024. Risk taxonomy, mitigation, and assessment benchmarks of large language model systems. *arXiv preprint arXiv:2401.05778* (2024).

[6] Alexandre Decan, Tom Mens, and Philippe Grosjean. 2019. An empirical comparison of dependency network evolution in seven software packaging ecosystems. *Empirical Software Engineering* 24, 1 (2019), 381–416.

[7] Zorik Gekhman, Gal Yona, Roee Aharoni, Matan Eyal, Amir Feder, Roi Reichart, and Jonathan Herzig. 2024. Does fine-tuning LLMs on new knowledge encourage hallucinations? *arXiv preprint arXiv:2405.05904* (2024).

[8] Lina Gong, Jingxuan Zhang, Mingqiang Wei, Haoxiang Zhang, and Zhiqiu Huang. 2023. What is the intended usage context of this model? an exploratory study of pre-trained models on various model repositories. *ACM Transactions on Software Engineering and Methodology* 32, 3 (2023), 1–57.

[9] Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. 2021. Pre-trained models: Past, present and future. *AI Open* 2 (2021), 225–250.

[10] Yujie Ji, Xinyang Zhang, Shouling Ji, Xiapu Luo, and Ting Wang. 2018. Model-reuse attacks on deep learning systems. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 349–363.

[11] Wenxin Jiang, Nicholas Synovic, Matt Hyatt, Taylor R Schorlemmer, Rohan Sethi, Yung-Hsiang Lu, George K Thiruvathukal, and James C Davis. 2023. An empirical study of pre-trained model reuse in the hugging face deep learning model registry. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 2463–2475.

[12] Jason Jones, Wenxin Jiang, Nicholas Synovic, George Thiruvathukal, and James Davis. 2024. What do we know about Hugging Face? A systematic literature review and quantitative validation of qualitative claims. In *Proceedings of the 18th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*. 13–24.

[13] Adhishree Kathikar, Aishwarya Nair, Ben Lazarine, Agrim Sachdeva, and Sagar Samtani. 2023. Assessing the vulnerabilities of the open-source artificial intelligence (ai) landscape: A large-scale analysis of the hugging face platform. In *2023 IEEE International Conference on Intelligence and Security Informatics (ISI)*. IEEE, 1–6.

[14] Divyanshu Kumar, Anurakt Kumar, Sahil Agarwal, and Prashanth Harshangi. 2024. Fine-tuning, quantization, and llms: Navigating unintended outcomes. *arXiv preprint arXiv:2404.04392* (2024).

[15] Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. 2024. Formalizing and benchmarking prompt injection attacks and defenses. In *33rd USENIX Security Symposium (USENIX Security 24)*. 1831–1847.

[16] msc empirical. 2025. *msc-empirical*. Retrieved July 1, 2025 from https://github.com/MSC-empirical/msc-empirical

[17] Federica Pepe, Vittoria Nardone, Antonio Mastropaolo, Gabriele Bavota, Gerardo Canfora, and Massimiliano Di Penta. 2024. How do hugging face models document datasets, bias, and licenses? an empirical study. In *Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension*. 370–381.

[18] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China technological sciences* 63, 10 (2020), 1872–1897.

[19] Jayaram Raghuram, George Kesidis, and David J Miller. 2024. A Study of Backdoors in Instruction Fine-tuned Language Models. *arXiv preprint arXiv:2406.07778* (2024).

[20] Trevor Stalnaker, Nathan Wintersgill, Oscar Chaparro, Laura A Heymann, Massimiliano Di Penta, Daniel M German, and Denys Poshyvanyk. 2025. The ML Supply Chain in the Era of Software 2.0: Lessons Learned from Hugging Face. *arXiv preprint arXiv:2502.04484* (2025).

[21] Zhen Sun, Tianshuo Cong, Yule Liu, Chenhao Lin, Xinlei He, Rongmao Chen, Xingshuo Han, and Xinyi Huang. 2025. PEFTGuard: detecting backdoor attacks against parameter-efficient fine-tuning. In *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1713–1731.

[22] Mina Taraghi, Gianolli Dorcelus, Armstrong Foundjem, Florian Tambon, and Foutse Khomh. 2024. Deep learning model reuse in the huggingface community: Challenges, benefit and trends. In *2024 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 512–523.

[23] vectara. 2025. *HHEM-2.1-Open*. Retrieved July 1, 2025 from https://github.com/vectara/hallucination-leaderboard

[24] Shenao Wang, Yanjie Zhao, Xinyi Hou, and Haoyu Wang. 2025. Large language model supply chain: A research agenda. *ACM Transactions on Software Engineering and Methodology* 34, 5 (2025), 1–46.

[25] Ying Wang, Bihuan Chen, Kaifeng Huang, Bowen Shi, Congying Xu, Xin Peng, Yijian Wu, and Yang Liu. 2020. An empirical study of usages, updates and risks of third-party libraries in java projects. In *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 35–45.

[26] wikipedia. 2025. *Cohen's kappa Wikipedia*. Retrieved March 27, 2025 from https://en.wikipedia.org/wiki/Cohen%27s_kappa

[27] Bowen Xu, Le An, Ferdian Thung, Foutse Khomh, and David Lo. 2020. Why reinventing the wheels? An empirical study on library reuse and re-implementation. *Empirical Software Engineering* 25 (2020), 755–789.

[28] Prateek Yadav, Tu Vu, Jonathan Lai, Alexandra Chronopoulou, Manaal Faruqui, Mohit Bansal, and Tsendsuren Munkhdalai. 2024. What Matters for Model Merging at Scale? *arXiv preprint arXiv:2410.03617* (2024).

[29] Zhou Yang, Jieke Shi, Prem Devanbu, and David Lo. 2024. Ecosystem of large language models for code. *ACM Transactions on Software Engineering and Methodology* (2024).

[30] Jinghuai Zhang, Jianfeng Chi, Zheng Li, Kunlin Cai, Yang Zhang, and Yuan Tian. 2024. Badmerging: Backdoor attacks against model merging. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 4450–4464.

[31] Shuai Zhao, Meihuizi Jia, Zhongliang Guo, Leilei Gan, Xiaoyu Xu, Xiaobao Wu, Jie Fu, Yichao Feng, Fengjun Pan, and Luu Anh Tuan. 2024. A survey of backdoor attacks and defenses on large language models: Implications for security measures. *Authorea Preprints* (2024).

[32] Hongling Zheng, Li Shen, Anke Tang, Yong Luo, Han Hu, Bo Du, and Dacheng Tao. 2023. Learn from model beyond fine-tuning: A survey. *arXiv preprint arXiv:2310.08184* (2023).

[33] Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043* (2023).