HuggingGraph: Understanding the Supply Chain of LLM Ecosystem

Mohammad Shahedur Rahman
Department of Computer Science and Engineering
The University of Texas at Arlington
Arlington, TX, USA
mxr7980@mavs.uta.edu

Peng Gao
Department of Computer Science
Virginia Tech
Blacksburg, VA, USA
penggao@vt.edu

Yuede Ji
Department of Computer Science and Engineering
The University of Texas at Arlington
Arlington, TX, USA
yuede.ji@uta.edu

Abstract

Large language models (LLMs) leverage deep learning architectures to process and predict sequences of words, enabling them to perform a wide range of natural language processing tasks, such as translation, summarization, question answering, and content generation. As existing LLMs are often built from base models or other pre-trained models and use external datasets, they can inevitably inherit vulnerabilities, biases, or malicious components that exist in previous models or datasets. Therefore, it is critical to understand these components' origin and development process to detect potential risks, improve model fairness, and ensure compliance with regulatory frameworks. Motivated by that, this project aims to study such relationships between models and datasets, which are the central parts of the *LLM supply chain*. First, we design a methodology to systematically collect LLMs' supply chain information. Then, we design a new graph to model the relationships between models and datasets, which is a directed heterogeneous graph, having *402,654 nodes* and *462,524 edges*. Lastly, we perform different types of analysis and make multiple interesting findings.

CCS Concepts

• Computing methodologies → Artificial intelligence; **• Theory of computation → Graph algorithms analysis**; **• Applied computing → Supply chain management**.

Keywords

Large language models (LLMs); AI supply chain; Graph analysis

ACM Reference Format:
Mohammad Shahedur Rahman, Peng Gao, and Yuede Ji. 2025. HuggingGraph: Understanding the Supply Chain of LLM Ecosystem. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25), November 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3746252.3761510

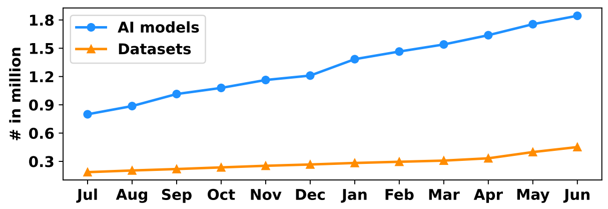

Figure 1: The number of AI models and datasets (in million scale) on Hugging Face from July 2024 to June 2025.

1 Introduction

Large language models (LLMs) are AI models designed to understand and generate human language by learning patterns and relationships within extensive datasets [39, 47], such as GPT (Generative Pre-trained Transformer) [60], BERT (Bidirectional Encoder Representations from Transformers) [30], and T5 (Text-To-Text Transfer Transformer) [5]. These models leverage deep learning architectures to process and predict sequences of words based on context, enabling them to perform a wide range of tasks [7], such as, translation [37], summarization [33], question-answering [3], and content generation [1]. LLMs usually have billions (or even trillions) of parameters [38], enabling them to generate high-quality text.

However, the increasing size and complexity of developing, training, and deploying cutting-edge LLMs demand extensive computational resources [58] and large-scale datasets [54]. This creates a significant barrier for researchers and practitioners, limiting their access to state-of-the-art models [40]. As the demand for democratizing access to such LLM models continues to rise, platforms that host models and datasets have gained widespread popularity, such as, Hugging Face [14], ONNX Model Zoo [41], and PyTorch Hub [44].

Figure 1 shows the number of AI models and datasets (in million scale) on Hugging Face, one of the largest public AI model hosting platforms [14], from July 2024 to June 2025. By the end of June 2025, it has reached over 1.8M models and 450K datasets. In addition, the trend does not show any slowdown. Such platforms provide user-friendly interfaces, APIs, and cloud-based infrastructures that enable researchers and developers to easily share, fine-tune, and deploy models without requiring extensive computational resources.

Based on the tasks, these models can be classified into two broad categories, *base models* and *task-specific models*. (i) *Base models* are large, pre-trained models that can be fine-tuned for specific downstream tasks [50]. They are usually trained on vast datasets and are general-purpose, such as GPT [60], BERT [30], and T5 [5].

(ii) *Task-specific models* are modified versions of base models for a specific task. Taking Hugging Face as an example, there are four types of such models. First, *fine-tuned models* adapt base models for specific tasks by training on additional task-specific datasets [61]. Second, *adapter models* add lightweight and modular layers to the pre-trained models for specific tasks [22]. Third, *quantization models* trade off precision in numerical computations for accelerating inference and reducing memory consumption (e.g., using less precise model parameters) [53]. Fourth, *merged models* integrate multiple models into a single unified model by combining weights or configurations [2]. Besides, such platforms also host many *datasets* used for training or adapting (e.g., fine-tuning) models [46].

1.1 Motivation

As existing LLMs are often built from base models or other pre-trained models and use external datasets, they can inevitably inherit vulnerabilities, biases, or malicious components from previous models or datasets. Thus, *understanding these components' origin and provenance can help better detect potential risks, improve model fairness, and ensure compliance with regulatory frameworks*.

Motivated by that, this paper aims to study such relationships between models and datasets. They are the central parts of the *LLM supply chain* [57], which refers to the entire lifecycle of developing, training, and deploying LLMs, similar to a traditional supply chain in manufacturing or software development [4, 10, 49, 62]. Such a supply chain can help to identify critical insights for both model evolution and dataset origin, as discussed below.

Model evolution. The study of the LLM supply chain gives a clear overview of how LLMs evolve from base models to fine-tuned variants, adapter integration, and quantization models. With that, one can easily keep track of them. For example, a use case is when a security vulnerability is found in one LLM, and we can quickly locate the potential models that might have the same vulnerabilities.

Dataset origin. This supply chain can also help to understand the datasets' origins used for training different models [47]. Dataset origin refers to the source from which the data is collected. For example, for a fine-tuned model, we not only care about which dataset is used for fine-tuning but also what other datasets are involved in training the previous model. Understanding such dataset origin helps to ensure that the dataset used is reliable, and legally compliant.

1.2 Contribution

Our main contributions are threefold. First, we design a methodology to systematically collect the supply chain information of LLMs. In this paper, we mainly study the most popular AI platform, i.e., Hugging Face, but the same strategy applies to other platforms. In particular, we use the APIs from the AI platform to collect the metadata about the hosted models and datasets. To this end, we collected a large dataset as of *June 30, 2025*.

Second, with the collected metadata, we construct a new graph, named ***LLM supply chain graph***[1][2], to model the relationships between models and datasets. It is a directed heterogeneous graph, where a node denotes different types of models and datasets, and an edge denotes the dependency between them. Together, this graph is able to accurately capture the LLM supply chain information. To this end, we get a graph with **402,654 nodes** and **462,524 edges**.

[1]The constructed graph: https://github.com/SC-Lab-Go/HuggingGraph
[2]A demonstration website: https://ai-supply-chain.github.io/

Table 1: The APIs used to extract data from Hugging Face.

API Name	Description
Model hub	Access model hub to list, search, and download models and metadata.
Datasets	Access the datasets for discovery, metadata retrieval, and downloading.
Metrics	Access metrics for model evaluation, e.g., metric discovery.
Search	Search name, tag, or other metadata for model and dataset.

Lastly, we perform different types of analysis, including forward and backward analysis. We study six research questions, including (i) the properties of the LLM supply chain graph, (ii) structural analysis, (iii) supply chain relationships between AI models, (iv) supply chain relationships between models and datasets, (v) dynamic update evaluation, and (vi) generalizability to other AI platforms.

We hope this study can not only provide insights on LLM supply chain, but also raise awareness and future interests in this direction.

2 Preliminary

LLM supply chain encompasses the interconnected processes required for developing, deploying, and maintaining models [57]. This includes sourcing and preparing data to ensure high-quality and diverse datasets [57]. It also involves creating and training models [34]. Finally, it covers making trained models available through APIs [48]. In addition, LLMs can undergo fine-tuning, adaptation, quantization, and merging processes in which they are tuned with domain-specific datasets to maximize performance on specific tasks [55], thus improving their accuracy and applicability. This study mainly focuses on the relationships between models and datasets, which are the central parts of the whole LLM supply chain ecosystem.

3 Methodology

3.1 LLM Supply Chain Information Collection

To analyze the LLM supply chain ecosystem, we need a large dataset with such information. Fortunately, platforms like Hugging Face provide some APIs that allow us to access the model and dataset and collect their metadata, which can be used to construct the LLM supply chain graph.

Table 1 summarizes the four types of APIs we used. In particular, (i) *the model hub APIs* allow access to the hub of existing models, including searching and downloading the model and its metadata. (ii) *The dataset APIs* allow access to the datasets for discovery, metadata retrieval, and downloading. (iii) *The metrics APIs* allow access to the metrics for model evaluation, including metric discovery, metadata retrieval, and calculation. (iv) One can use *the search APIs* to search the name, tag, or other metadata for models and datasets.

Handling missing information. Accurate construction of the LLM supply chain graph depends on the quality of metadata from the LLM platforms (e.g., Hugging Face), which might suffer from missing or incomplete data. To address it, we apply the two following techniques, i.e., cross-reference links, and textual pattern extraction.

(i) Cross-reference links. The model or dataset description could miss the supply chain data fields for API queries, which could be embedded within statically or dynamically rendered HTML pages. In this example URL https://huggingface.co/models?other=base_model:finetune:meta-llama/Meta-Llama-3-8B, one can tell the model "*Meta-Llama-3-8B*" is fine-tuned from the model in the previous webpage. To capture such information, we cross-reference the

links of the filtered model listing webpages, extract model identifiers, enable the reconstruction of supply chain graph edges, and recursively trace model lineage from the leaf node. This scraping step complements API-based extraction and is only employed when reverse dependency data is otherwise inaccessible.

(ii) Textual pattern extraction. When structured metadata is absent, the model and dataset cards might mention dependencies in unstructured text descriptions. To capture that, we employ a named entity recognition (NER) method [36, 52] to extract the dependency relationships from the text. For example, the textual phrase like "fine-tuned from Llama-2" contains the *fine-tuned* keyword, which implies from which model this model is actually fine-tuned. Similarly, we also look into other words, such as, "train", and "adapt".

3.2 LLM Supply Chain Graph

The collected metadata can be accurately modeled by the graph data structure. It is a directed heterogeneous graph, where a node denotes different types of datasets and models, including base, fine-tune, adapter, quantization, and merge models. An edge denotes the dependency relationship between them, including model-model, dataset-dataset, and model-dataset relationships.

Figure 2 shows a simplified supply chain subgraph centering on a base model *Meta-llama*. *(i) Model-model relationship.* To further identify the supply chain relationship, we will check the relevant data fields. In particular, given a model in Hugging Face, there are data fields "`finetune`" that show which models are fine-tuned from this model. Similarly, "*adapter*", "*quantization*", and "*merge*" show which models are adapted, quantized or merged from them, respectively. With such information, we can construct the supply chain relationship between the models. As shown in Figure 2, model *Llama-3.3-70B* is fine-tuned from the base model *Meta-llama*. Then, it is used by the models *Doctor-Shotgun* and *Llama-3.3-70B-4bit* to generate an adapter and quantization model, respectively.

(ii) Dataset–dataset relationship. The datasets within the LLM supply chain might overlap, build upon, or extend from each other. For example, a dataset may be a subset or modified version of another. To capture such information, we connect them with two types of edges. *(1) Subset relationships* arise when a dataset is explicitly documented as a subsample or partition of another. For instance, a dataset named *"C4_200M"* is described as a subset of *"C4"*. *(2) Modified versions* represent updates or enhanced variants of existing datasets. For example, *"TruthfulQA_v2"* incorporates corrections and improvements over an earlier version, *"TruthfulQA_v1"*.

(iii) Model-dataset relationship. To capture this, we use the metadata from both models and datasets. The metadata of a model might specify the datasets used for training or adapting. However, not all the models disclose such information. To capture more information, we find the metadata of a dataset contains a data field of "trained_fine_tune_models" on this dataset. Thanks to that, we can capture the accurate model and dataset relationship. In Figure 2, the datasets *The Pile* and *Chatgpt-prompt* have directed edges to model *Meta-Llama*, meaning that both datasets are used to train the model.

3.3 Supply Chain Graph Analysis

This supply chain graph can help to understand the transformational processes of the models and datasets. In particular, we can understand how base models evolve into their variants, including fine-tuned, adapted, quantized, or merged models, and vice versa. Similar observations can be made for datasets. This would provide a clear view of how the base model (or dataset) is transformed for performing a particular task. In particular, we mainly perform two types of analysis, i.e., forward and backward analysis.

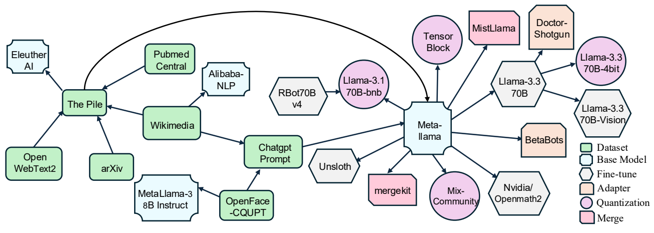

Figure 2: An example subgraph centering on base model "`Meta-llama`" from the complete LLM supply chain graph.

Forward analysis is the method of traversing the supply chain graph following the dependency edges of a chosen node in a forward-going way. This node (known as the root/source node) can be a dataset, a base model, a fine-tuned model, an adapter, a quantized, or a merged model. In particular, given a source node, we apply the graph traversal algorithm (e.g., breadth-first search (BFS) [32]) to traverse all the nodes (including both models and datasets) in a level-by-level pattern. To that end, this forward analysis will identify all the nodes that are reachable from the source node.

Model analysis example. In Figure 2, we analyze the forward supply chain of *Meta-Llama*. In particular, we identify four distinct forward paths: (i) Base model (*Meta-llama*) → fine-tuned model (*Llama-3.3-70B*) → another fine-tuned model (*Llama3.3-70B-Vision*). (ii) Base model (*Meta-llama*) → adapted model (*Doctor-Shotgun*). (iii) Base model (*Meta-llama*) → fine-tuned model (*Llama-3.3-70B*) → quantization model (*Llama-3.3-70B-4bit*). (iv) Base model (*Meta-llama*)) → merged model (*MistLlama*). These paths show the evolution trajectory of the base model *Meta-Llama*, showcasing its progressive specialization and adaptation for various tasks.

Dataset analysis example. For the dataset, our supply chain analysis shows how different datasets connect and form a new dataset. This combination creates flexible resources that show how models perform in various areas. In Figure 2, the dataset *The Pile* is composed of multiple subsets, including *Wikimedia*, *arXiv*, *OpenWebtext2*, and *Pubmed Central*. Together, these datasets form a unified corpus that serves as training data for models like *Meta-LlaMA*.

Backward analysis is the method of traversing the supply chain graph following the edges in a backward way. We accomplish this by traversing the graph also with BFS [32], starting from the selected node and following the incoming edges. To that end, this backward analysis will identify all the nodes that can reach the source node.

Model and dataset analysis example. In Figure 2, analyzing the backward supply chain of model *RBot70Bv4*, we trace its lineage through its development stages. This model is fine-tuned from *Unsloth*, which in turn originates from its base model, *Meta-Llama*, and the datasets used to train the base model are *The Pile* and *Chatgpt-prompts*. Through this analysis, we establish the backward path, starting from the target model, i.e., *RBot70Bv4*, and tracing to its base model, *Meta-Llama*, revealing dependencies and transformations involved in its development. Similarly, we can analyze datasets.

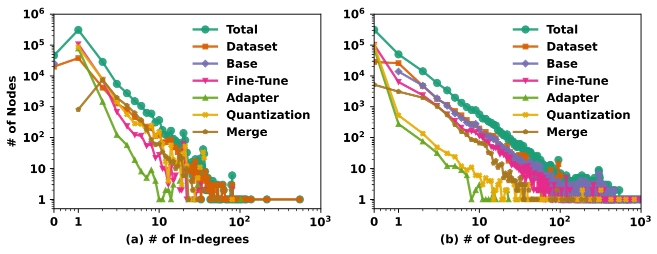

Figure 3: (a) Indegree and (b) outdegree distribution. X-axis shows degrees, and Y-axis shows node count (log scale).

3.4 Accommodating Dynamic Updates

The hosted models and datasets on AI platforms are growing fast as new models are being developed every day. For example, between June 25 and July 15, 2025, we observed 80,703 new models (approximately 3,843 per day), 27,405 new datasets (approximately 1,305 per day) on Hugging Face, which is just one of the many AI platforms. Therefore, we need to accommodate the dynamic update to accurately manage and analyze the AI supply chain.

Particularly, HuggingGraph accommodates the dynamic updates in three steps. *(i) Scoping updated models or datasets.* At a time t, we keep a copy of the hosted models and datasets with their IDs. When it evolved to $t + 1$, we get another copy of the hosted models and datasets with their IDs. The difference between them shows the updated models and datasets, including newly added or deleted. In our current implementation, we are keeping the update on a daily basis. *(ii) Metadata collection for the updated models or datasets.* For the identified updated models or datasets, we will collect their metadata using the same strategy as discussed in Section 3.1. To this end, we get the updated dependencies between models and datasets. That is, for the update at time $t + 1$ compared to t, it can be represented as Δ_{t+1}. *(iii) Δ-based dynamic graph update.* Given the newly updated dependency Δ_{t+1}, and let G_t, G_{t+1} denote the graph at time t, $t + 1$, respectively, then $G_{t+1} = G_t \cup \Delta_{t+1}$.

4 Experiment and Finding

To deeply understand the relationships between models and datasets, we study six critical research questions (RQs) as below.

- **RQ #1**: What are the properties of LLM supply chain graph?
- **RQ #2**: What structural patterns emerge?
- **RQ #3**: What are the supply chain between LLM models?
- **RQ #4**: What are the relationships between models and datasets?
- **RQ #5**: What insights can be gained from the dynamic updates?
- **RQ #6**: How can HuggingGraph be applied to other platforms?

4.1 RQ #1: Supply Chain Graph Properties

This research question aims to understand the critical properties of LLM supply chain graph, i.e., graph basics and degree distribution.

Graph basics. The collected supply chain graph is a medium-scale directed heterogeneous graph with **402,654 nodes** and **462,524 edges** as of *June 30th, 2025*. In particular, there are six different types of nodes, including 28,384 base models, 115,211 fine-tuned, 79,254 adapters, 98,143 quantization models, 13,028 merges, and 68,634 datasets. The average degree is about 1.15, denoting that it is a very sparse graph. Furthermore, we identified substantial metadata missing. As of June 30, 2025, among 1.8 million models, only 50,156 (2.79%) provides a model tree, while ~550K models lack any metadata beyond their names, and nearly another 400K models are empty. Similarly, of the 450K datasets, only 68,634 (15.26%) provide dataset cards, leaving ~380K datasets without any metadata.

This highlights a broader issue in the AI community, where a significant number of models and datasets lack consistent and structured documentation on the supply chain. This reflects the need of more transparent disclosure.

Degree distribution shows how node degrees (the number of edges connected to a node) are distributed across the graph. Figure 3 illustrates the indegree and outdegree distribution of the graph. We show not only the total distribution but also the distribution of six types of nodes, including base models, fine-tuned models, adapter models, quantization models, merged models, and datasets.

We observe that the degree distribution in our supply chain graph shows a **heavy-tailed behavior**. In particular, the indegree distribution shows a large spread across different categories. The outdegree distribution follows a similar pattern but may differ in specific cases (e.g., adapters seem to have a more restricted degree distribution). This heavy-tailed behavior suggests that most nodes have low degrees, while a few central nodes (hubs) dominate the graph. In particular, the dataset *macrocosm-os/images* has the highest indegree 550, and dataset *Mistral-v0.1* from "mistral AI" has the highest outdegree 1,093. Specifically, the base models act as high-degree hub nodes as they are heavily used by other task-specific models.

Finding #1: *The LLM supply chain graph is medium-scale, sparse, and heavy-tailed distribution. However, a significant number of models and datasets lack metadata, highlighting the need for more transparent supply chain documentation.*

4.2 RQ #2: Supply Chain Structural Analysis

This research question aims to understand the topology and evolution of the LLM supply chain. To achieve that, we analyze the structural properties with connectivity and community analysis.

Connectivity analysis. We computed weakly connected components (WCCs), which identify a maximal subset of nodes that remain connected when edge directions are ignored [26, 28]. The total number of WCCs in our supply chain graph is 44,908.

Figure 4 shows the cumulative distribution function (CDF) of the WCC distribution. We observe (i) the largest WCC covers 247,244 nodes, accounting for 61.4% of all the nodes. It reflects the dense interconnections that pervade the ecosystem. This vital element is essential for effective information sharing, resource allocation, and structural support, and is the base of the ecosystem. In the largest WCC, major models are included, such as *Gemma-2B*, *DistilBERT*, and *GPT-2*. (ii) In contrast, the remaining WCCs collectively hold 38.6% of the nodes, with most having 1, 2, or 3 nodes. This indicates a fragmented outer edge characterized by specialized models, rare datasets, or active experimental projects. The prevalence of these

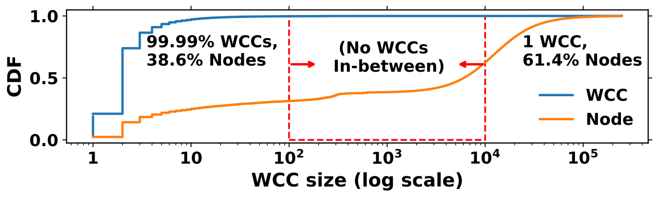

Figure 4: Cumulative distribution function (CDF) of WCC size.

Table 2: Top-10 Louvain communities sorted by size.

ID	Size	E.g. models	E.g. datasets	Modularity
1	9,390	OLMoE, CausalLM	prompt-perfect	0.96
2	7,388	qwen2.5_math	Marco-o1	0.96
3	6,989	Wanxiang	smartllama3.1	0.96
4	6,813	tinyllama	Llama-1B	0.96
5	5,163	Qwen2.5-32B	Matter-0.2	0.96
6	4,554	MedLlama-3-8B	dpo-mix	0.96
7	4,262	Electra, ArliAI	MixEval	0.96
8	3,947	aesqwen1.5b	llava	0.96
9	3,829	bert	TORGO	0.96
10	3,828	Mistral	vicuna_format	0.96

small, isolated pieces suggest niche attempts that lack integration with the overall system. For example, *zhongqy/RMCBench* (benchmarking dataset) and *yigit69/bert-base-uncased-finetuned-rte-run_3* (recognizing textual entailment task) remain disconnected due to limited reuse or insufficient metadata.

We also computed strongly connected component (SCC) [27], a maximal subgraph in which every node is reachable from every other via directed edges, identifying 398,198 SCCs. Remarkably, only 591 of these are non-trivial (size > 1), collectively encompassing 5,047 nodes (1.25% of the graph). The largest SCC comprises 478 dataset nodes, among them *tree-of-knowledge* and *OpenHermes-2.5*, forming a tightly-knit cluster. By contrast, the remaining 99.46% of nodes each reside in trivial (size-1) SCCs.

Community detection identifies node groups with dense internal connections. In LLM supply chains, it reveals aligned subgraphs reflecting reuse patterns and task-specific assets. We apply the Louvain method [13], a greedy algorithm that maximizes modularity to find densely connected communities. Higher modularity indicates stronger intra-community connectivity.

Table 2 summarizes the top-10 communities. (i) We find that each of the top-10 communities achieve a high modularity score of 0.96, indicating strong intra-community connectivity. Collectively, these communities span a wide range of functional domains, underscoring the presence of well-defined, task-aligned clusters within the ecosystem. (ii) The largest community consists of 9,390 nodes and attains a modularity score of 0.96, indicating an extremely cohesive internal structure. It revolves around base models like *OLMoE* and *CausalLM*, and general-purpose datasets like *prompt-perfect*. This suggests a densely connected cluster facilitating widespread reuse and fine-tuning. (iii) Several other communities reflect clear task-based segmentation. For instance, community 2 (7,388 nodes) focuses on solving mathematical problems, with models like *qwen2.5_math* and datasets like *Marco-o1*. Similarly, community 6 (4,554 nodes) centers on instruction-tuning, with model *MedLlama-3-8B* and dataset *dpo-mix*.

Finding #2: *The LLM supply chain graph features a dominant core (61.4% of nodes), while high modularity (0.96) reveals task-aligned, semantically coherent communities amid a fragmented periphery.*

4.3 RQ #3: Supply Chain Analysis of LLM Models

This research question aims to provide a holistic view of the dependencies between the models within the LLM supply chain, particularly from both base and task-specific models.

Base model impact. We would like to understand the impact of base models. Here, we quantify the impact of a base model as

Table 3: Top-10 base models sorted by forward subgraph size.

Base model	Total	Fine-tune	Adapter	Quanti-zation	Merge	Level
Llama-3.1-8B	7,544	1,710	1,542	3,473	1,693	25
Mistral-7B-v0.1	6,744	2,105	2,187	1,435	1,254	27
Qwen2.5-7B	6,733	1,972	1,764	2,516	1,132	11
Meta-Llama-3-8B	5,633	967	1,511	2,220	1,967	21
Llama-3.1-70B	4,063	698	281	2,075	2,519	11
Qwen2.5-32B	3,909	1,086	158	2,311	1,049	12
Qwen2.5-1.5B	3,645	1,300	1,290	949	248	8
Qwen2.5-0.5B	3,521	1,669	1,006	810	46	11
Qwen2.5-14B	3,362	726	411	1,880	1,166	15
Meta-Llama-3-8B-Instruct	3,118	640	405	1,394	1,305	34

Table 4: Top-10 models sorted by backward subgraph size.

Model	Model Type	Total	Fine-tune	Quanti-zation	Level	Base Model
command-r-1-layer	Finetune	40	39	0	39	c4ai
KoModernBERT	Finetune	21	20	0	20	ModernBERT
t5-small	Finetune	21	20	0	20	t5-small
clinical_260k	Finetune	20	19	0	19	clinical_180K
t5-small-finetuned	Finetune	17	16	0	16	t5-small
clinical_300k	Finetune	16	15	0	15	clinical_180K
clinical_259k	Finetune	16	15	0	15	clinical_180K
LeoPARD-0.8.1	Finetune	16	2	13	15	DeepSeek-R1
LeoPARD-0.8.2-4bit	Quantization	16	1	14	15	DeepSeek-R1
LeoPARD-0.8.1-4bit	Quantization	16	1	14	15	DeepSeek-R1

the number of task-specific models that depend on it. The more dependencies, the larger the impact it has. We start with a base model and perform forward analysis by computing BFS following the outgoing edges. This leads to a forward subgraph, which denotes all the models that depend on the base model, including fine-tuned, adapted, quantized, or merged models.

Table 3 shows the top-10 base models sorted by the forward subgraph size, which is the number of impacted task-specific models. We find (i) a base model can significantly impact the LLM supply chain ecosystem. For example, *Llama-3.1-8B* is a base model from Meta used for efficient text generation, code assistance, and research [20]. Due to its relatively small size, which allows for deployment in resource-constrained environments, making advanced AI accessible to broader stakeholders [17]. It has generated up to 7,544 models, including 1,710 fine-tuned versions, 1,542 adapters, 3,473 quantizations, and 1,693 merged models tailored to specific tasks. (ii) For fine-tuning, the base model *Mistral-7B-v0.1* has been fine-tuned the most, totaling 2,105. It is a faster, lighter *Mistral* model trained by Mistral AI with grouped-query and sliding-window attention, enabling efficient text generation, NLP, and code assistance on consumer hardware for low-latency tasks [51].

Task-specific model analysis. Given a task-specific model, we want to understand how it evolves, e.g., what other models it relies on. To achieve that, for each model, we perform a backward analysis by running BFS following the incoming edges. To that end, the derived subgraph shows the models it relies on.

Table 4 shows the top-10 task-specific models sorted by the backward subgraph size, which is the number of models they rely on. We make two interesting observations. (i) A fine-tuned model, *command-r-1-layer*, illustrates the depth and complexity of transformations in the LLM supply chain. This model operates in bfloat16 (BF16) precision for efficient text generation and natural language understanding [43], originates from the base model *c4ai*, and has undergone extensive lineage evolution before reaching its final form.

Table 5: Top-10 datasets sorted by # of models trained.

Dataset	Total	Fine-tune	Adapter	Quantization	Merge
Mistral-v0.1	1,093	300	300	193	300
TinyLlama-1.1B-v1.0	728	300	300	100	28
open_llama_3b	304	15	285	4	0
Yarn-Mistral-7b-128k	301	8	279	14	0
WizardVicuna-open-llama	280	12	261	7	0
TinyLlama-1.1B-v0.6	266	10	243	13	0
Yarn-Mistral-7b-64k	248	0	242	6	0
Nous-Capybara-7B-V1	213	11	174	27	1
MAmmoTH2-7B	213	0	0	3	210
Starling-LM-7B-alpha	210	10	165	18	17

Specifically, it depends on 40 upstream artifacts, including 39 other fine-tuned models, and spans 39 transformation levels in its backward lineage chain, as detailed in Table 4. (ii) We observe that adapters are mainly used for lightweight fine-tuning and merges for model integration, but task-specific models like *command-r-1-layer*, as optimized standalone derivatives, do not evolve from adapters or merges in their backward lineage [43].

Finding #3: *Base models like Llama-3.1-8B dominate the LLM supply chain, spawning thousands of derivatives, while task-specific models such as command-r-1-layer exhibit deep dependencies with other task-specific variants but avoid adapters or merges.*

4.4 RQ #4: Supply Chain of Models and Datasets

This research question aims to explore the interconnections between models and datasets from dual perspectives, including one dataset versus multiple models and one model versus multiple datasets.

One dataset versus multiple models refers to the case when a single dataset is used to train multiple models. Table 5 shows the top-10 datasets based on the number of models trained on them. In particular, (i) *Mistral-v0.1* takes the leading position and is a widely adopted open-source dataset known for its strong performance in general-purpose language understanding and generation tasks. It has been used to train 1,093 models, including 300 fine-tuned variants, 300 adapters, 193 quantized models, and 300 merged models, highlighting its broad adoption across diverse model derivation strategies. (ii) The dataset *TinyLlama-1.1B-v1.0* has been used to train 728 models, featuring 300 fine-tuned variants and 300 adapters. Similarly, *open_llama_3b*, an open-access dataset of Llama, supports 285 adapter-based models, indicating a preference for lightweight, modular adaptation. (iii) Furthermore, *MAmmoTH2-7B* stands out with 210 merged models, showcasing its role in ensemble-style model fusion rather than traditional fine-tuning or adapter strategies.

One model versus multiple datasets refers to the case when an LLM model is trained with multiple datasets. Table 6 shows the top-10 models ranked by the number of datasets used for training. We observe that *DeBERTa-ST-AllLayers-v3.1*, a fine-tuned variant of the *DeBERTa* model, takes the top position, having been trained on 116 different datasets. Its adapter-based counterpart, *DeBERTa-ST-AllLayers-v3.1bis*, also leverages the same number of datasets via adapter-based training, emphasizing modular reuse across tasks.

In addition, models like *static-similarity-mrl-mul-v1* and *static-similarity-mrl-multilingual* are both fine-tuned on 108 datasets, indicating their roles in multilingual and multi-task similarity-based retrieval applications. Interestingly, most of these models are fine-tuned, specifically, 8 out of the top-10, suggesting that fine-tuning remains a dominant strategy for adapting base models to downstream tasks across heterogeneous data sources.

Table 6: Top-10 models sorted by # of training datasets.

Model	Model Type	# of training dataset
DeBERTa-ST-AllLayers-v3.1	Fine tune	116
DeBERTa-ST-AllLayers-v3.1bis	Adapters	116
static-similarity-mrl-mul-v1	Fine tune	108
static-similarity-mrl-multilingual	Fine tune	108
ModernBERT-base-embed	Fine tune	88
Llama-3.2-3B-Instruct	Fine tune	87
Llama-3.2-3B-Instruct-GGUF	Quantization	87
DavidLanz-3.2-3B-Instruct	Fine tune	87
static-retrieval-mrl-en-v1	Fine tune	79
XLMRoBERTaM3-CustomPoolin	Fine tune	72

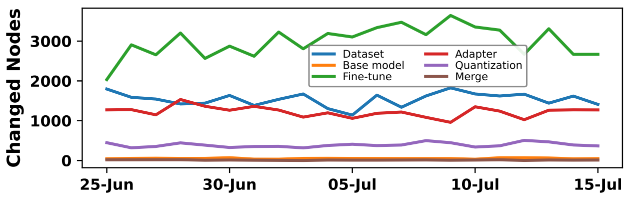

Figure 5: The number of changed datasets and models on Hugging Face from June 25 to July 15, 2025.

Finding #4: *Models and datasets exhibit strong bidirectional interdependence, with datasets like Mistral-v0.1 spawning hundreds of models, while models such as DeBERTa-ST-AllLayers-v3.1 leverage different datasets to enhance adaptability, highlighting the critical roles of dataset-model interactions in advancing AI.*

4.5 RQ #5: Dynamic Update Evaluation

This research question aims to understand the dynamic update of the LLM supply chain. We perform a daily-based data collection by capturing the addition and deletion of nodes and edges each day.

Figure 5 illustrates the sum of daily additions and deletions of six key node categories (base, fine-tuned, adapters, quantized, merge variants, and datasets) from June 25 to July 15, 2025. We make three interesting observations. (i) The daily dynamic update is significant. That is, an average of 4,622 models are changing every day, including ~3,843 model additions and ~779 deletions. In addition, about 1,538 datasets are changing each day, containing ~1,305 dataset additions and ~233 deletions. An addition occurs when a new model or dataset is uploaded to the Hugging Face platform. This includes base models, task-specific variants, and new training datasets. A deletion refers to the removal of such nodes, often due to licensing issues, privacy concerns, or contributor decisions, such as replacing outdated models or withdrawing low-quality or sensitive datasets.

(ii) Fine-tuned models dominate daily activity, averaging over 2,988 changes per day, followed by consistent contributions from adapters (~1,224/day) and datasets (~1,539/day). Noticeable spikes, such as on July 7 and July 9, align with major events like the *Mistral-Fusion-v3* fine-tuning wave and dataset updates such as *HFTime2025-News*. (iii) Furthermore, adapter uploads peaked at 1,249 on June 28, while quantized variants reached 381 on July 9, driven by releases such as *Qwen-GGUF-7B*. These patterns demonstrate HuggingGraph's ability to capture evolving supply chain dynamics at a fine-grained level.

Finding #5: *The LLM supply chain exhibits continuous and high-volume daily changes, driven by frequent additions and deletions of models and datasets. This reflects a rapidly evolving and highly dynamic ecosystem shaped by active contributor behavior.*

4.6 RQ #6: Generalizability to Other AI Platforms

To validate HuggingGraph's generalizability beyond Hugging Face, we applied our pipeline to another AI platform, Kaggle [6]. As of July 25, 2025, Kaggle hosts 470 base models, 3,146 task-specific models, and ~502K datasets. Using Kaggle's kernel and dataset APIs, we collected 2,640 models and 105,867 datasets. This significant gap is due to a lack of models' and datasets' metadata. Of the datasets retrieved (~105K), only 137 datasets were included in our graph, as most lacked standardized documentation or traceable links to models. Many are standalone, poorly described, or lack contextual information, a challenge also observed on Hugging Face.

We follow the same way to construct a heterogeneous graph consisting of 2,777 nodes, which include 2,640 model nodes, comprising 59 base models, 2,410 fine-tuned, 171 quantization models, and 137 dataset nodes. The graph contains a total of 3,990 edges, on which we observed seven types of edges, (i) base model → fine-tuned model (467 edges), (ii) base model → quantization model (62 edges), (iii) fine-tuned model → fine-tuned model (1,696 edges), (iv) fine-tuned model → quantized model (107 edges), (v) quantized model → quantized model (1 edge), (vi) dataset → fine-tuned model (1,614 edges), and (vii) dataset → quantization model (43 edges).

We observed that the average degree is 1.44, indicating that the graph is sparse. We made two interesting observations. (i) The degree distribution is heavy-tailed and skewed: out of 2,777 total nodes, 2,305 nodes (83%) have a total degree of 1. Most nodes have low degrees, while a few highly connected hubs dominate the graph. For example, in-degrees range from 0 to 4, and *tensorflow/mobilenet-v1* has the highest out-degree of 64, followed by *google/nnlm* with 56. (ii) Furthermore, the graph contains 448 WCCs, reflecting high fragmentation. However, the largest WCC includes 65 nodes, suggesting the presence of a moderately sized core subgraph.

Finding #6: *The resulting graph exhibits structural properties consistent with our Hugging Face analysis, including a heavy-tailed degree distribution, sparse connectivity, and strong modular fragmentation, demonstrating the robustness and generalizability of our pipeline across platforms despite metadata limitations.*

5 Use Case

HuggingGraph presents a technique to analyze the supply chain of the LLM ecosystem, which can be used for various applications, e.g., auditing provenance, identifying biases, and revealing trends like quantized model scarcity. We discuss the two use cases.

Use case #1: Tracing lineage and dependencies in the LLM supply chain. Models are frequently built upon others through fine-tuning, adapter training, or quantization, forming complex chains of dependencies. However, when these relationships are not explicitly visible, it becomes difficult to verify where a model comes from, whether it inherits bias from upstream datasets, or if it complies with licensing constraints. HuggingGraph can be used to address this challenge by constructing the supply chain of models and datasets, uncovering both direct and derived dependencies, even when they are not formally documented. For example, it can trace how the model *Meta-llama* indirectly relies on a dataset like *Wikimedia* via *Chatgpt-prompt* (Figure 2). This transparency supports developers, auditors, and policymakers in validating provenance, detecting risks, and enabling trustworthy AI.

Use case #2: Identifying critical nodes and structural vulnerabilities. In the LLM ecosystem, certain models (e.g., *gemma-2b*) and datasets (e.g., *The Pile*) are reused so frequently that they become critical structural hubs, where failure or removal of them could disrupt numerous downstream dependencies. These hidden single points of failure are difficult to detect without a comprehensive view of resource interconnections. HuggingGraph can be used to address this by modeling the supply chain as a graph and analyzing node connectivity to surface highly reused models and datasets with significant inbound or outbound links. This visibility enables maintainers to safeguard vital assets and helps developers mitigate the risk of overreliance on fragile or under-maintained components.

6 Related Work and Discussion

LLM supply chain perspectives in AI. LLMs are advancing across model infrastructure, lifecycle, and applications [57]. Model reuse is widespread, promoting large-scale sharing and adaptation of base models [29]. Open-source ecosystems such as Hugging Face host diverse LLMs and datasets, democratizing AI [45]. Base models [58], trained on broad datasets, enable task-specific variants via fine-tuning [18], reinforcing democratization and innovation [12].

Relationship analysis between LLM models and datasets. A recent study investigates the practical adaptation of base models to specific tasks. Multitask fine-tuning has demonstrated the potential to enhance performance on target tasks with scarce labels [59]. In plant phenotyping, adapting vision-based models by techniques like adapter tuning and decoder tuning has shown results comparable to those of leading task-specific models [9]. The Quadapter technique for language models tackles quantization difficulties by incorporating learnable parameters that scale activations channel-wise, mitigating overfitting during quantization-aware training [42].

In future, we would like to explore more through the following perspectives, (i) collecting more LLM supply chain information accurately and scalably; (ii) understanding the LLM supply chain better via both fundamental graph analytics [15, 21, 35] and graph AI techniques [16, 23, 31, 56]; and (iii) exploring the security and privacy threats on the LLM supply chain [8, 11, 19, 24, 25].

7 Conclusion

This project studies the relationships between models and datasets in the LLM ecosystem, which are the central parts of the *LLM supply chain*. First, we systematically collect the supply chain information of LLMs. With that, we construct a directed heterogeneous graph, having *402,654 nodes* and *462,524 edges*. Lastly, we perform different types of analysis and make multiple interesting findings.

Acknowledgment

This work was supported in part by National Science Foundation grants 2331301, 2508118, 2516003, 2419843. The views, opinions, and/or findings expressed in this material are those of the authors and should not be interpreted as representing the official views of the National Science Foundation, or the U.S. Government.

Disclaimer on Generative AI Usage

This work does not make use of generative AI tools (e.g., ChatGPT, Copilot, Bard, Claude, etc.) for ideation, analysis, code development, or writing. All content, experiments, and results presented in this paper were created and validated solely by the authors. Mentions of generative AI are included only for scholarly context or comparison.

References

[1] Alexandre Agossah, Frédérique Krupa, Matthieu Perreira Da Silva, and Patrick Le Callet. 2023. Llm-based interaction for content generation: A case study on the perception of employees in an it department. In *Proceedings of the 2023 ACM International Conference on Interactive Media Experiences*. 237–241.

[2] Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. 2025. Evolutionary optimization of model merging recipes. *Nature Machine Intelligence* 7, 2 (2025), 195–204.

[3] Dean Allemang and Juan Sequeda. 2024. Increasing the LLM Accuracy for Question Answering: Ontologies to the Rescue! *arXiv preprint arXiv:2405.11706* (2024).

[4] Anonymous. 2022. Review of Supply Chain Management in Manufacturing Organizations. *ResearchGate* (2022). https://www.researchgate.net/publication/377659033_Review_of_supply_chain_management_in_manufacturing_organizations

[5] Mourad Bahani, Aziza El Ouaazizi, and Khalil Maalmi. 2023. The effectiveness of T5, GPT-2, and BERT on text-to-image generation task. *Pattern Recognition Letters* 173 (2023), 57–63.

[6] Casper Solheim Bojer and Jens Peder Meldgaard. 2021. Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting* 37, 2 (2021), 587–603.

[7] Euan Bonner, Ryan Lege, and Erin Frazier. 2023. Large Language Model-Based Artificial Intelligence in the Language Classroom: Practical Ideas for Teaching. *Teaching English with Technology* 23, 1 (2023), 23–41.

[8] Dalton A Brucker-Hahn, Wang Feng, Shanchao Li, Matthew Petillo, Alexandru G Bardas, Drew Davidson, and Yuede Ji. 2024. CloudCover: Enforcement of Multi-Hop Network Connections in Microservice Deployments. In *2024 Annual Computer Security Applications Conference (ACSAC)*. IEEE, 1186–1202.

[9] Feng Chen, Mario Valerio Giuffrida, and Sotirios A Tsaftaris. 2023. Adapting vision foundation models for plant phenotyping. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 604–613.

[10] M. C. Chou, H. Ye, X. M. Yuan, Y. N. Cheng, L. Chua, Y. Guan, S. E. Lee, and Y. C. Tay. 2006. Analysis of a Software-Focused Products and Service Supply Chain. *IEEE Transactions on Industrial Informatics* 2, 4 (2006), 295–303. doi:10.1109/TII.2006.884368

[11] Lei Cui, Jiancong Cui, Yuede Ji, Zhiyu Hao, Lun Li, and Zhenquan Ding. 2023. API2Vec: Learning Representations of API Sequences for Malware Detection. In *International Symposium on Software Testing and Analysis (ISSTA)*.

[12] Jelena Cupać, Hendrik Schopmans, and İrem Tuncer-Ebetürk. 2024. Democratization in the age of artificial intelligence: introduction to the special issue. 899–921 pages.

[13] Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, and Alessandro Provetti. 2011. Generalized louvain method for community detection in large networks. In *2011 11th international conference on intelligent systems design and applications*. IEEE, 88–93.

[14] Hugging Face. [n. d.]. Hugging Face – The AI community building the future. https://huggingface.co/

[15] Wang Feng, Shiyang Chen, Hang Liu, and Yuede Ji. 2023. Peek: A Prune-Centric Approach for K Shortest Path Computation. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–14.

[16] Qiang Fu, Yuede Ji, and H Howie Huang. 2022. TLPGNN: A lightweight two-level parallelism paradigm for graph neural network computation on GPU. In *HPDC*.

[17] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783* (2024).

[18] 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.

[19] Haojie He, Xingwei Lin, Ziang Weng, Ruijie Zhao, Shuitao Gan, Libo Chen, Yuede Ji, Jiashui Wang, and Zhi Xue. 2024. Code is not Natural Language: Unlock the Power of Semantics-Oriented Graph Representation for Binary Code Similarity Detection. In *The 33rd USENIX Security Symposium (USENIX Security)*.

[20] Zhengfu He, Wentao Shu, Xuyang Ge, Lingjie Chen, Junxuan Wang, Yunhua Zhou, Frances Liu, Qipeng Guo, Xuanjing Huang, Zuxuan Wu, et al. 2024. Llama scope: Extracting millions of features from llama-3.1-8b with sparse autoencoders. *arXiv preprint arXiv:2410.20526* (2024).

[21] Runbang Hu, Chaoqun Li, Xiaojiang Du, and Yuede Ji. 2025. Adaptive Optimizations for Parallel Single-Source Shortest Paths. In *Proceedings of the 1st FastCode Programming Challenge*. 53–56.

[22] Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. 2023. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2304.01933* (2023).

[23] Yuede Ji. 2025. High-Performance Computing for Graph AI: A Top-Down Perspective. In *Proceedings of the 15th NSF/TCPP Workshop on Parallel and Distributed Computing Education (EduPar '25)*.

[24] Yuede Ji, Lei Cui, and H. Howie Huang. 2021. BugGraph: Differentiating Source-Binary Code Similarity with Graph Triplet-Loss Network. In *16th ACM ASIA Conference on Computer and Communications Security (AsiaCCS)*.

[25] Yuede Ji, Mohamed Elsabagh, Ryan Johnson, and Angelos Stavrou. 2021. DEFInit: An Analysis of Exposed Android Init Routines. In *30th USENIX Security Symposium (USENIX Security)*.

[26] Yuede Ji and H. Howie Huang. 2020. Aquila: Adaptive Parallel Computation of Graph Connectivity Queries. In *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing (HPDC)*.

[27] Yuede Ji, Hang Liu, and H. Howie Huang. 2018. iSpan: Parallel Identification of Strongly Connected Components with Spanning Trees. In *International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*. IEEE, 731–742.

[28] Yuede Ji, Hang Liu, and H. Howie Huang. 2020. SwarmGraph: Analyzing Large-Scale In-Memory Graphs on GPUs. In *International Conference on High Performance Computing and Communications (HPCC)*. IEEE.

[29] 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.

[30] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, Vol. 1. Minneapolis, Minnesota, 2.

[31] TN Kipf. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint arXiv:1609.02907* (2016).

[32] Donald E. Knuth. 1974. *The Art of Computer Programming, Volume 1: Fundamental Algorithms* (2nd ed.). Addison-Wesley.

[33] Philippe Laban, Wojciech Kryściński, Divyansh Agarwal, Alexander Richard Fabbri, Caiming Xiong, Shafiq Joty, and Chien-Sheng Wu. 2023. SUMMEDITS: measuring LLM ability at factual reasoning through the lens of summarization. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 9662–9676.

[34] Beibin Li, Konstantina Mellou, Bo Zhang, Jeevan Pathuri, and Ishai Menache. 2023. Large language models for supply chain optimization. *arXiv preprint arXiv:2307.03875* (2023).

[35] Chaoqun Li, Runbang Hu, Xiaojiang Du, and Yuede Ji. 2025. Optimized Parallel Breadth-First Search with Adaptive Strategies. In *Proceedings of the 1st FastCode Programming Challenge*. 28–32.

[36] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE transactions on knowledge and data engineering* 34, 1 (2020), 50–70.

[37] Yinquan Lu, Wenhao Zhu, Lei Li, Yu Qiao, and Fei Yuan. 2024. Llamax: Scaling linguistic horizons of llm by enhancing translation capabilities beyond 100 languages. *arXiv preprint arXiv:2407.05975* (2024).

[38] Felix Merrick, Maria Radcliffe, and Rupert Hensley. 2024. Upscaling a smaller llm to more parameters via manual regressive distillation. (2024).

[39] Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. 2023. Large language models as general pattern machines. *arXiv preprint arXiv:2307.04721* (2023).

[40] Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt llm evaluation. *Transactions of the Association for Computational Linguistics* 12 (2024), 933–949.

[41] ONNX Community. 2025. ONNX Model Zoo: Pre-trained Models for ONNX. https://github.com/onnx/models.

[42] Minseop Park, Jaeseong You, Markus Nagel, and Simyung Chang. 2022. Quadapter: Adapter for gpt-2 quantization. *arXiv preprint arXiv:2211.16912* (2022).

[43] Eduardo Pignatelli, Johan Ferret, Tim Rockäschel, Edward Grefenstette, Davide Paglieri, Samuel Coward, and Laura Toni. 2024. Assessing the zero-shot capabilities of LLMs for action evaluation in RL. *arXiv preprint arXiv:2409.12798* (2024).

[44] PyTorch Core Team. 2024. PyTorch Hub. https://pytorch.org/hub.

[45] Matteo Riva, Tommaso Lorenzo Parigi, Federica Ungaro, and Luca Massimino. 2024. HuggingFace's impact on medical applications of artificial intelligence.

Computational and Structural Biotechnology Reports (2024), 100003.

[46] Paul Röttger, Fabio Pernisi, Bertie Vidgen, and Dirk Hovy. 2024. Safetyprompts: a systematic review of open datasets for evaluating and improving large language model safety. *arXiv preprint arXiv:2404.05399* (2024).

[47] Zhiqiang Shen, Tianhua Tao, Liqun Ma, Willie Neiswanger, Zhengzhong Liu, Hongyi Wang, Bowen Tan, Joel Hestness, Natalia Vassilieva, Daria Soboleva, et al. 2023. Slimpajama-dc: Understanding data combinations for llm training. *arXiv preprint arXiv:2309.10818* (2023).

[48] Tanmay Singla, Dharun Anandayuvaraj, Kelechi G Kalu, Taylor R Schorlemmer, and James C Davis. 2023. An empirical study on using large language models to analyze software supply chain security failures. In *Proceedings of the 2023 Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*. 5–15.

[49] Sonatype. 2015. *2015 State of the Software Supply Chain Report*. Technical Report. Sonatype. https://www.sonatype.com/hubfs/White_Papers/2015_State_of_the_Software_Supply_Chain_Report-.pdf

[50] Xin Tan, Taichuan Li, Ruohe Chen, Fang Liu, and Li Zhang. 2024. Challenges of Using Pre-trained Models: the Practitioners' Perspective. *arXiv preprint arXiv:2404.14710* (2024).

[51] Hiren Thakkar and A Manimaran. 2023. Comprehensive examination of instruction-based language models: A comparative analysis of mistral-7b and llama-2-7b. In *2023 International Conference on Emerging Research in Computational Science (ICERCS)*. IEEE, 1–6.

[52] Yuli Vasiliev. 2020. *Natural language processing with Python and spaCy: A practical introduction*. No Starch Press.

[53] Neelay Velingker, Jason Liu, Amish Sethi, William Dodds, Zhiqiu Xu, Saikat Dutta, Mayur Naik, and Eric Wong. [n. d.]. CLAM: Unifying Finetuning, Quantization, and Pruning by Chaining LLM Adapter Modules. In *Workshop on Efficient Systems for Foundation Models II@ ICML2024*.

[54] Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2024. Will we run out of data? Limits of LLM scaling based on human-generated data. *arXiv preprint arXiv:2211.04325* (2024), 13–29.

[55] Kushala VM, Harikrishna Warrier, Yogesh Gupta, et al. 2024. Fine Tuning LLM for Enterprise: Practical Guidelines and Recommendations. *arXiv preprint arXiv:2404.10779* (2024).

[56] Pinhuan Wang, Chengying Huan, Zhibin Wang, Chen Tian, Yuede Ji, and Hang Liu. 2025. Bingo: Radix-based Bias Factorization for Random Walk on Dynamic Graphs. In *Proceedings of the Twentieth European Conference on Computer Systems* (Rotterdam, Netherlands) *(EuroSys '25)*. Association for Computing Machinery, 16 pages.

[57] Shenao Wang, Yanjie Zhao, Xinyi Hou, and Haoyu Wang. 2024. Large language model supply chain: A research agenda. *ACM Transactions on Software Engineering and Methodology* (2024).

[58] Mengwei Xu, Wangsong Yin, Dongqi Cai, Rongjie Yi, Daliang Xu, Qipeng Wang, Bingyang Wu, Yihao Zhao, Chen Yang, Shihe Wang, et al. 2024. A survey of resource-efficient llm and multimodal foundation models. *arXiv preprint arXiv:2401.08092* (2024).

[59] Zhuoyan Xu, Zhenmei Shi, Junyi Wei, Fangzhou Mu, Yin Li, and Yingyu Liang. 2024. Towards Few-Shot Adaptation of Foundation Models via Multitask Finetuning. *arXiv preprint arXiv:2402.15017* (2024).

[60] Gokul Yenduri, M Ramalingam, G Chemmalar Selvi, Y Supriya, Gautam Srivastava, Praveen Kumar Reddy Maddikunta, G Deepti Raj, Rutvij H Jhaveri, B Prabadevi, Weizheng Wang, et al. 2024. Gpt (generative pre-trained transformer)–a comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions. *IEEE Access* (2024).

[61] Wentao Zou, Qi Li, Jidong Ge, Chuanyi Li, Xiaoyu Shen, Liguo Huang, and Bin Luo. 2023. A Comprehensive Evaluation of Parameter-Efficient Fine-Tuning on Software Engineering Tasks. *arXiv preprint arXiv:2312.15614* (2023).

[62] Ajdin Čolaković, Aleksandar Đorđević, Branislav Cvetić, Milan Danilović, and Dejan Vasiljević. 2021. Traditional vs Digital Supply Chains. *ResearchGate* (2021). https://www.researchgate.net/publication/381617842_Traditional_vs_Digital_Supply_Chains