Software Engineering for Large Language Models: Research Status, Challenges and the Road Ahead

HONGZHOU RAO*, Huazhong University of Science and Technology, China
YANJIE ZHAO*, Huazhong University of Science and Technology, China
XINYI HOU, Huazhong University of Science and Technology, China
SHENAO WANG, Huazhong University of Science and Technology, China
HAOYU WANG†, Huazhong University of Science and Technology, China

The rapid advancement of large language models (LLMs) has redefined artificial intelligence (AI), pushing the boundaries of AI research and enabling unbounded possibilities for both academia and the industry. However, LLM development faces increasingly complex challenges throughout its lifecycle, yet no existing research systematically explores these challenges and solutions from the perspective of software engineering (SE) approaches. To fill the gap, we systematically analyze research status throughout the LLM development lifecycle, divided into six phases: requirements engineering, dataset construction, model development and enhancement, testing and evaluation, deployment and operations, and maintenance and evolution. We then conclude by identifying the key challenges for each phase and presenting potential research directions to address these challenges. In general, we provide valuable insights from an SE perspective to facilitate future advances in LLM development.

1 INTRODUCTION

In recent years, large language models (LLMs) have advanced rapidly, leading to their performance exceeding human capabilities in certain domains [93, 307, 316, 329, 457]. Alongside this progress, emerging technologies such as AI-driven code generation [107, 137, 228, 468], multimodal models [294, 333], and AI agents [383] are also evolving at an unprecedented pace. These developments are rapidly expanding the role of LLMs in critical domains such as software development, healthcare [95, 100, 416], and finance [430, 450]. As LLMs become foundational infrastructure for general-purpose intelligence [253], ensuring their reliability, efficiency, and adaptability is important as well as challenging for the software engineering (SE) community. Therefore, a systematic investigation into the development and engineering of LLMs is essential.

The development of LLMs is a multifaceted process, from dataset preparation and model training to deployment and maintenance. SE plays a central role throughout this lifecycle, offering foundational principles and methodologies to manage complexity, ensure robustness, and support scalability. These are increasingly embodied in an ecosystem of specialized tools and frameworks that facilitate each stage of the development process. For instance, Hugging Face provides tools such as Transformers [105] for model training and the PEFT library [75] for efficient fine-tuning methods. Micros ft's DeepSpeed [338] enhances large-scale model training through deep learning optimizations, while OpenAI offers LLM APIs [266] that enable interaction with the GPT family models. Additionally, evaluation frameworks like LM-Eval-Harness [72] and community-driven platforms such as the Open LLM Leaderboard [74] offer standardized benchmarks. Development toolkits like LangChain [28] modularize and streamline the construction

*Co-first authors who contributed equally to this work.
†Haoyu Wang is the corresponding author (haoyuwang@hust.edu.cn).

Authors' addresses: Hongzhou Rao, rhz@hust.edu.cn, Huazhong University of Science and Technology, Wuhan, China; Yanjie Zhao, yanjie_zhao@hust.edu.cn, Huazhong University of Science and Technology, Wuhan, China; Xinyi Hou, xinyihou@hust.edu.cn, Huazhong University of Science and Technology, Wuhan, China; Shenao Wang, shenaowang@hust.edu.cn, Huazhong University of Science and Technology, Wuhan, China; Haoyu Wang, haoyuwang@hust.edu.cn, Huazhong University of Science and Technology, Wuhan, China.

of LLM-based applications. These tools embody SE principles and reflect the increasing need for structured, reliable, and scalable LLM development pipelines.

Beyond model training and development, SE also plays a critical role in optimizing the efficiency and scalability of LLM deployment. We can develop tools that integrate techniques such as model compression, quantization, and inference optimization to reduce latency and resource consumption. For example, TensorRT [263] and vLLM [349] employ these techniques to enable cost-effective and efficient LLM inference in production environments. Furthermore, **specialized protocols like Model Context Protocol (MCP) [14] and Agent-to-Agent (A2A) [335] standardize interactions between LLM-based agents, tools, and multi-agent systems, ensuring interoperability and streamlined integration in production pipelines**. SE practices play a crucial role in implementing these protocols, providing structured approaches to develop and maintain associated SDKs [266], client libraries [28, 105], and server components [246, 339], while ensuring code quality, reliability, and scalability. In summary, from infrastructure development and data/model management to deployment and inference acceleration, SE methodologies and tools continue to shape the rapid evolution and widespread adoption of LLMs.

Despite these advancements, there are unique SE challenges for LLM development. High computational costs [311, 327], non-deterministic testing [324], and continuous model updates in dynamic environments [236] demand a re-evaluation of traditional SE practices. Traditional MLOps methodologies, initially designed for smaller-scale machine learning (ML) models, are no longer well-suited for LLMs, necessitating large language model operations (LLMOps) [59]. Furthermore, LLMs deviate from traditional software paradigms: unlike traditional programs that produce deterministic outputs, LLMs generate responses probabilistically due to their neural network-based reasoning mechanisms. Additionally, their complex architectures and large scale lead to their outputs being challenging to explain, making interpretability and debugging far more difficult than in traditional software systems. These factors highlight the need for engineering solutions to address LLMs' inherent unpredictability, lack of transparency, and unique operational constraints.

However, we found that extensive research has explored LLM capabilities [126, 380, 471], while systematic investigations from an SE perspective are still lacking. To address this gap, we present **the first comprehensive study of the SE challenges encountered throughout the LLM development lifecycle and outline future research directions**. Specifically, we categorize the LLM development lifecycle into six key phases: requirements engineering (RE), dataset construction, model development and enhancement, testing and evaluation, deployment and operations, and maintenance and evolution. For each phase, we analyze the current research status to identify key challenges and propose potential future research directions from an SE perspective.

In summary, our primary contributions are:

- Our work is the first to investigate the role of SE in the development of LLMs, filling the gap in current research.
- We divide the LLM development lifecycle into six phases and systematically analyze the scope and significance of SE for LLMs.
- We analyze the latest research on LLMs, identify current challenges, and propose corresponding future research directions.

The remainder of this paper is structured as follows: In §2, we introduce the scope and significance of SE for LLMs. We then analyze various aspects of LLM development, covering RE (§3), dataset construction (§4), development and enhancement (§5), testing and evaluation (§6), deployment and operations (§7), and maintenance and evolution (§8), as illustrated in Figure 1. Finally, we introduce the related work in §9 and conclude the paper in §10.

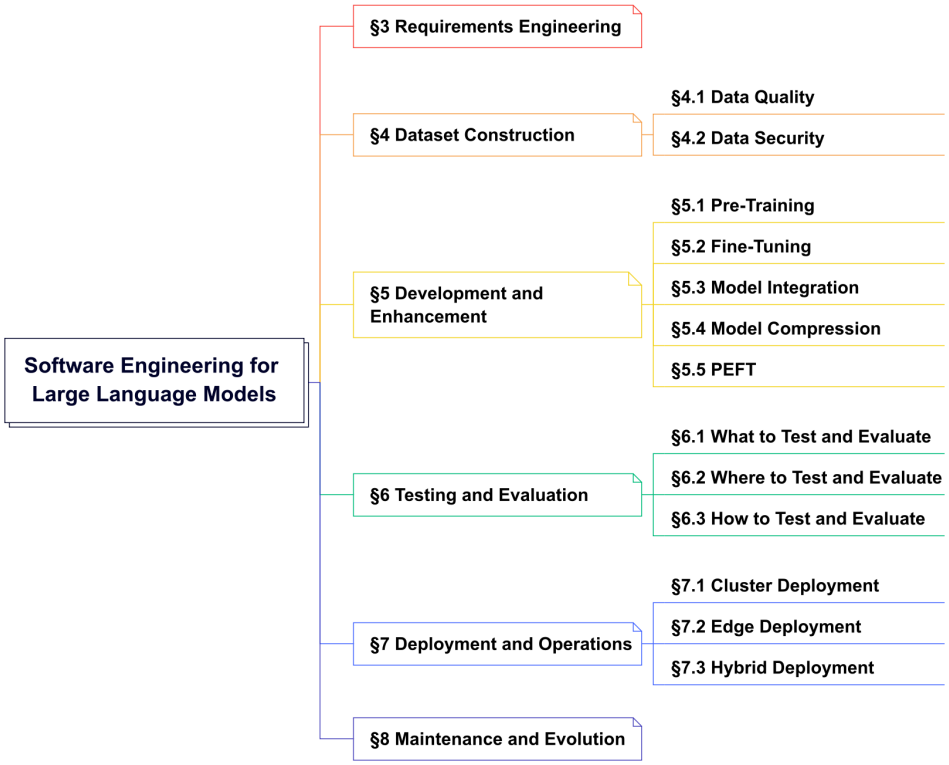

Fig. 1. Phase Organization Overview for the LLM Development Lifecycle.

2 SCOPE AND SIGNIFICANCE

In this section, we introduce the scope of SE for LLMs in §2.1 and its significance in §2.2.

2.1 Scope

As summarized in Table 1, we can see that LLMs share similarities with traditional software while also exhibiting differences. Unlike traditional software, LLMs are built upon neural network architectures, resulting in non-deterministic outputs for identical inputs. Additionally, they differ from traditional software in terms of executability and testing methodologies. Despite these differences, LLMs retain many characteristics of traditional software. Given these hybrid characteristics, SE methodologies can be applied to enhance LLM in phases such as development, deployment, and maintenance. To systematically investigate this intersection, we explore how SE methodologies support various phases of the LLM development lifecycle, which consists of the phases illustrated in Figure 2:

- **RE.** The initial phase of RE for LLMs involves identifying specific performance metrics (e.g., accuracy, latency, energy consumption) and functional capabilities (e.g., reasoning, multimodal understanding) that the model is expected to possess, which is typically followed by a systematic process of requirement refinement, feasibility analysis, and validation to ensure that the specified goals are both realistic and implementable.
- **Dataset construction.** Once requirements are established, vast datasets must be prepared for subsequent pre-training and fine-tuning. The construction of datasets involves data

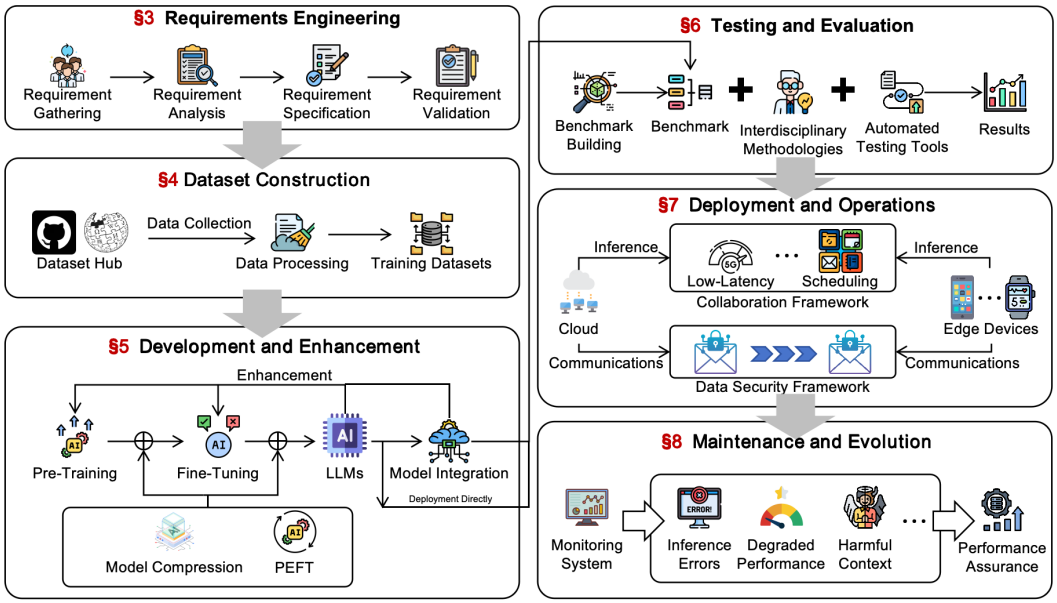

Fig. 2. Detailed Activity Breakdown for the LLM Development Lifecycle.

collection and processing to produce high-quality and secure datasets, as the dataset's quality significantly impacts model performance [220]. Harmful data can lead to unintended biases or malicious content generation.

- **Development and enhancement.** The development of LLMs generally comprises two key stages: pre-training and fine-tuning. Building an LLM from scratch entails designing and implementing foundational architectures (e.g., Transformers) and performing large-scale training, which is a highly complex and resource-intensive engineering undertaking. Fortunately, open-source development frameworks facilitate this process. For example, Hugging Face Transformers [105] facilitates model pre-training and fine-tuning, making it a mainstream tool for LLM development. After development, models can undergo further training to strengthen specific capabilities. Additionally, they can be integrated with tools to evolve into more advanced LLM agents.
- **Testing and evaluation.** Evaluating an LLM requires comprehensive and systematic testing to assess its diverse capabilities across different tasks and scenarios. However, due to the inherent complexity of LLMs and their non-deterministic outputs, traditional software testing methodologies are often not suitable. When evaluating a model, it is essential to consider not only basic performance measures but also practical challenges, such as hallucinations, inconsistent outputs, context sensitivity, and other factors that may impact its reliability. Therefore, further research is needed to improve evaluation methods for LLMs.
- **Deployment and operations.** Once validated, an LLM can be deployed across various application scenarios. Some models are hosted in cloud environments and accessed via APIs (e.g., GPT-4.5 [267], Claude-3.7-Sonnet [13]), while others are deployed on edge devices or within hybrid edge–cloud setups to achieve low-latency and resource-efficient inference. However, this diversity in deployment environments introduces new challenges, ranging from scalability and reliability to resource allocation and system integration. Addressing these

Table 1. Comparison of Characteristics Between Traditional Software and LLM.

Characteristic	Traditional Software	LLM	Similarity
Determinism	Producing consistent outputs for the same inputs.	Generating probabilistic outputs with inherent variation.	Low
Executability	Executing based on explicit, defined logic.	Processing through neural inference with limited transparency.	Medium
Maintainability	Maintaining through code modifications and debugging.	Improving via fine-tuning, retraining, or data augmentation.	High
Reusability	Reusing code components across different projects.	Adapting pre-trained models for various tasks.	High
Testability	Supporting systematic unit and integration testing.	Requiring output-based evaluation with uncertainty tolerance.	Medium
Scalability	Expanding through modular design principles.	Scaling via MoE, LoRA, RAG, and parameter-efficient methods.	High
Deployability	Requiring platform-specific deployment approaches.	Functioning across platforms with similar infrastructure needs.	High

issues requires strong support from SE practices, such as automated deployment pipelines, environment-specific optimizations, and real-time monitoring systems.

- **Maintenance and evolution.** During operation, LLMs require substantial computational resources and may encounter issues such as performance degradation, inference errors, or the need for retraining and knowledge updates. Therefore, LLMs also require systematic maintenance, bringing additional challenges beyond those of traditional software systems.

From this lifecycle perspective, it is evident that SE methodologies are deeply integrated into the construction, deployment, and utilization of LLMs. In dataset construction, specialized tools facilitate data cleaning and synthesis. Model development relies heavily on existing frameworks, while model enhancement often involves integrating LLMs with external tools to extend their functionality. Furthermore, testing, evaluation, and deployment require novel approaches distinct from traditional SE practices due to challenges such as debugging difficulties, output variability, and high computational demands. Beyond engineering complexities, LLMs introduce additional security and ethical concerns. They are vulnerable to adversarial threats, including data poisoning and prompt injection attacks, and may perpetuate biases inherent in their training data. Addressing these issues necessitates SE techniques. **Overall, SE in LLMs aims to support structured and efficient development across the full lifecycle—from requirements engineering and dataset construction to model deployment and maintenance—while also tackling concerns such as security, ethical responsibilities, and regulatory compliance.**

2.2 Significance

LLMs are hard to understand because they have very complex structures and rely heavily on neural networks to make decisions. This lack of explanation makes it difficult to optimize, test, and maintain these models over time. They also introduce new security and privacy issues that are different from those in traditional software. As LLM applications expand across various domains and industries, these risks become increasingly critical. To address these issues, we require robust safety measures and transparent development processes. Given these challenges, the application of SE methodologies can help ensure that LLMs are built, deployed, and maintained reliably and responsibly while addressing associated security, ethical, and regulatory concerns.

2.2.1 SE for LLM Development. SE provides systematic and automated tools and methodologies to support the development of LLM, significantly enhancing both efficiency and reliability. Due to

the massive scale of LLMs and their long training cycles, manually managing parameters, datasets, and code versions is complex and error-prone. SE methodologies, such as LLMOps pipelines, automated hyperparameter management tools, and model version control systems (e.g., Weights & Biases [373]), facilitate automation and standardization in the model development phase. These tools reduce the likelihood of human error and enhance reproducibility.

Beyond automation, LLM development presents challenges that differ from traditional SE, such as stability issues during pre-training [337] and catastrophic forgetting during fine-tuning [179]. These issues can be mitigated through real-time training monitoring and content-detection techniques. Additionally, SE methodologies help manage the complexities introduced by parameter-efficient fine-tuning (PEFT). For example, when integrating adapter layers [161] or Low-Rank Adaptation (LoRA) [118], it is critical to ensure effective management, maintain compatibility between multiple tasks, and preserve performance stability. The principles of SE, such as modular design and continuous integration (e.g., automated adapter testing and parameter compatibility verification), provide structured solutions for efficiently managing and securely applying these fine-tuning techniques.

Furthermore, SE plays an essential role in model versioning and iterative upgrades. Automated tools that compare different model versions help prevent performance degradation and functionality loss, enabling smoother and more reliable updates. Thus, beyond enhancing development efficiency, SE methodologies also help address engineering challenges unique to LLMs, facilitating their continuous improvement and large-scale deployment.

2.2.2 SE for LLM Deployment. SE plays a crucial role in the deployment of LLMs by enabling efficient model compression [423] and automated deployment tools (e.g., Hugging Face Inference Endpoints [77]). While LLMs offer superior performance, their substantial computational demands pose significant challenges for deployment in resource-constrained devices, such as mobile devices and IoT edge devices [58]. SE addresses these challenges by advancing model compression techniques, including quantization [30], knowledge distillation (KD) [465], and pruning [448], as well as adapter-based approaches such as LoRA [445] and Prefix-Tuning [191], effectively reducing parameter size and computational cost. For instance, quantization techniques, such as INT8 [56] and INT4 [201], enable large models to achieve efficient inference on consumer-grade GPUs and even mobile devices [427], significantly expanding their applicability. Additionally, the development of modular and standardized LLM service interfaces (e.g., OpenAI API [268]) allows developers to seamlessly deploy and transition between models across diverse environments, thereby reducing system deployment complexity.

Beyond model optimization, SE also enhances LLM deployment through the development of standardized interaction protocols, such as MCP [14] for structured tool integration and A2A [335] communication for multi-agent collaboration. These protocols streamline integration with external APIs, databases, and distributed AI agents while ensuring interoperability and fault tolerance. Additionally, the development of modular and standardized LLM service interfaces (e.g., OpenAI API [268]) allows developers to seamlessly deploy and transition between models across diverse environments, thereby reducing system deployment complexity.

2.2.3 SE for LLM Maintenance. SE methodologies are essential for the long-term maintenance and evolution of LLMs. Once deployed, LLMs require continuous updates to incorporate new features, adapt to evolving business needs, and respond to environmental changes. As the number of model versions grows, managing version compatibility, tracking dataset evolution [269], and ensuring API stability [234] become critical challenges. SE addresses these challenges through version control systems for models and datasets, modular architecture designs (e.g., LoRA adapters), and automated regression testing frameworks (e.g., LLM-specific continuous integration tools), enabling efficient tracking of model functionality changes and rapid issue resolution. For instance, when releasing

new versions of the Gemini [99] and LLaMA [9] families, Google and Meta employ rigorous SE practices to manage version compatibility, maintain API stability, and ensure seamless migration for downstream users. These engineering-driven maintenance strategies significantly enhance model evolution efficiency, ensuring consistent and reliable performance while supporting the sustained deployment of LLMs.

2.2.4 SE for LLM Security. LLMs process vast amounts of sensitive data during inference and deployment, particularly in API-based services, raising concerns about data leakage and adversarial attacks (e.g., prompt injection [102, 221], backdoor attacks [217]). SE plays a critical role in establishing systematic security mechanisms, including secure access control, privacy-preserving techniques (e.g., differential privacy [27], federated learning [425]), and trusted execution environments (TEE) [249]. For instance, Azure OpenAI Service [246], as an AI service provider, implements strict role-based access control (RBAC) to ensure that users can only access authorized data and functionalities. Concurrently, research efforts are exploring the application of differential privacy in data processing, as demonstrated by Google's work [170], to prevent sensitive training data from being exposed during inference. Moreover, in multi-LLMs deployment scenarios, ensuring secure inference environments through containerization and sandboxing techniques (e.g., Intel SGX [141]) is essential. These approaches isolate user inputs during inference, preventing unauthorized access and adversarial exploitation, thereby significantly enhancing the security and trustworthiness of LLM.

3 REQUIREMENTS ENGINEERING

From this chapter onward, we analyze the research status of each phase in the LLM development lifecycle, identify the challenges, and propose potential future directions. For RE of LLM, the challenges and potential future directions are shown in Figure 3.

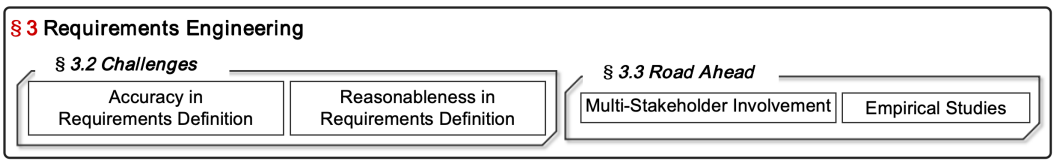

Fig. 3. Challenges and Road Ahead in §3 Requirements Engineering.

3.1 Research Status

To the best of our knowledge, research on RE for LLMs remains relatively limited. Due to the strong natural language processing (NLP) capabilities of LLMs, existing studies primarily focus on leveraging LLMs to support RE tasks, while comparatively fewer efforts investigate RE methodologies for LLM development itself. This imbalance mirrors a similar trend observed in the broader AI domain. As noted by Ahmad et al. [8], between 2011 and 2021, only approximately 43 publications explicitly addressed RE for AI, whereas a substantially larger work explored the use of AI techniques to enhance RE processes.

However, this imbalance does not imply that RE for LLMs is insignificant. As LLMs are increasingly applied across diverse domains, they encounter distinct requirements based on the specific demands of different scenarios. It is essential to thoroughly understand these requirements and tailor the development of LLMs accordingly. Fischer et al. [86] fine-tuned a model based on the requirements of investigative intelligence, yet their understanding of user requirements was mainly

derived from prior research. In contrast, Solomon et al. [323] conducted an RE study to investigate the use of LLMs in digital inquiry processes aimed at enhancing healthcare applications. They proposed a generalizable RE methodology for LLMs that incorporates both qualitative and quantitative analyses. The qualitative analysis involves studying the target population's background, including cultural and linguistic factors, while the quantitative analysis utilizes techniques such as word embeddings and network analysis to construct a semantic framework for the model. Beyond user research methods, Hassani et al. [117] addressed the requirements of a food company by fine-tuning LLMs to enhance their ability to classify legal texts related to food safety, incorporating both food safety system and software requirements. Additionally, Sjostrom et al. [321] proposed meta-requirements for LLM-based knowledge retrieval tools. Although the methodologies proposed in these studies are domain-specific and lack general applicability, they underscore the critical role of RE in LLM development: through rigorous RE, developers can identify the specific functionalities and performance requirements needed, ultimately enabling the customization of powerful, task-specific models.

In addition to the domain-specific requirements for LLMs discussed above, several general requirements recur across diverse scenarios. These include requirements related to dataset quality, energy efficiency, user preferences, and model interpretability. Dataset quality is a key factor influencing the performance of LLMs [220]. Despite its importance, there remains no clear consensus on which specific quality metrics are most relevant or how they should be applied. As a result, researchers continue to face difficulties in consistently evaluating and comparing dataset quality across different tasks and domains. In resource-constrained devices such as edge devices (as discussed in §7.2), LLMs may meet requirements related to energy consumption and computational efficiency. User preference requirements are also increasingly prominent, as policies, cultural values, and ethical standards vary significantly across regions. These contextual factors influence an acceptable LLM response. Finally, explainability remains a key concern. Since LLMs function primarily as black boxes, it is often unclear how they arrive at specific outputs. This lack of transparency raises important questions about the reliability, accuracy, and trustworthiness of their responses [449].

3.2 Challenges

Even before the emergence of LLMs, the impressive capabilities of AI had already given rise to misconceptions that AI could address all problems [8]. With the advent of LLMs, these expectations have grown even further, leading to an increased demand for functional requirements (FRs). Concurrently, the inherent complexity of LLMs has given rise to a diverse array of non-functional requirements (NFRs), such as interpretability, robustness, and efficiency. We categorize the key challenges of RE for LLMs into two dimensions: the accuracy and the reasonableness of requirement definitions.

Accuracy in requirements definition. Clearly defining requirements, whether FRs or NFRs, remains a challenging task. For example, Hassani et al. [117] found it difficult to determine which food safety regulations applied to software requirements, as these laws were not formulated initially with digital systems or AI integration in mind. Similarly, when defining NFRs such as creativity, conceptual ambiguity becomes a significant obstacle. Questions such as "What constitutes creativity in the context of LLMs?", "Which domains should it be evaluated in?" and "How can it be measured objectively?" are still far from being resolved [87].

Reasonableness in requirements definition. Beyond accuracy, it is equally essential to ensure that requirements are reasonable and achievable. Conflicting requirements or unrealistic expectations often necessitate trade-offs to formulate practical and balanced specifications. For instance, in edge deployment scenarios involving resource-constrained devices, a certain degree of

performance degradation is inevitable. So defining acceptable trade-offs, such as how much accuracy can be sacrificed while maintaining robustness, or how much memory and computing resources the model is allowed to use, presents a complex challenge. Output consistency is another challenge. Due to the inherently probabilistic behavior of LLMs, it is often unrealistic to expect identical outputs for the same input, which calls for more flexible and context-aware approaches to defining such requirements. Together, these challenges highlight the need for refined RE methodologies tailored to LLMs—methods that can more accurately and practically define both FRs and NFRs.

3.3 Road Ahead

To address the challenges outlined above, we propose two feasible research directions. First, to improve the accuracy of requirements definition, we recommend adopting a **multi-stakeholder involvement** strategy, which engages users, developers, and domain experts in a collaborative process to reach consensus on requirements. This approach has been successfully applied in several specific scenarios. For instance, Solomon et al. [323] collaborated with medical professionals to validate the accuracy of their analytical framework and established general principles for requirement definition in digital healthcare contexts. Similarly, Chakrabarty et al. [24] recruited writers, volunteers, and domain experts to define LLM creativity, incorporating expert insights alongside standardized creativity assessment methods such as the Torrance Tests of Creative Thinking (TTCT) [344]. However, these examples mainly focus on specific, well-defined user groups. In scenarios involving international or cross-domain user bases, the effectiveness and scalability of such multi-stakeholder approaches become limited. Therefore, a promising future direction is to investigate how to more systematically and scientifically incorporate diverse stakeholder perspectives when defining requirements across broader, more heterogeneous application environments.

Second, to address the challenge of defining reasonable requirements, we advocate for increased emphasis on **empirical studies**. Such studies enable a systematic understanding of the trade-offs involved in LLM deployment by evaluating model performance across diverse scenarios, which facilitates the establishment of practical, evidence-based requirement boundaries. For example, Huang et al. [136] conducted an extensive evaluation of LLaMA3 quantization across 1-8 bits settings, yielding empirical insights into the trade-offs between model performance and memory efficiency. Nevertheless, such empirical efforts are still relatively limited, highlighting the need for further research in this direction.

4 DATASET CONSTRUCTION

In the development of LLMs, training datasets, encompassing both pre-training and fine-tuning corpora, play a crucial role. In this section, we analyze datasets primarily from two critical dimensions: data quality and data security. These aspects not only influence model performance and generalization but also raise technical and ethical challenges. As shown in Figure 4, by focusing on our discussion around these two dimensions, we aim to provide a comprehensive perspective on the construction and utilization of datasets in LLM development.

4.1 Data Quality

4.1.1 Research Status. Data quality directly influences the diversity, relevance, and accuracy of datasets, which are critical factors in improving LLM performance for specific tasks [220]. Feng et al. [85] demonstrated a positive correlation between the frequency of causal relationships in pretraining corpora and LLM performance in causal discovery tasks. Similarly, Rao et al. [293] proposed a pre-training approach that leverages the mapping between code and test files to enhance the relevance of training data, thereby improving LLM-generated test cases. As a result, obtaining high-quality datasets has become a key focus of research.

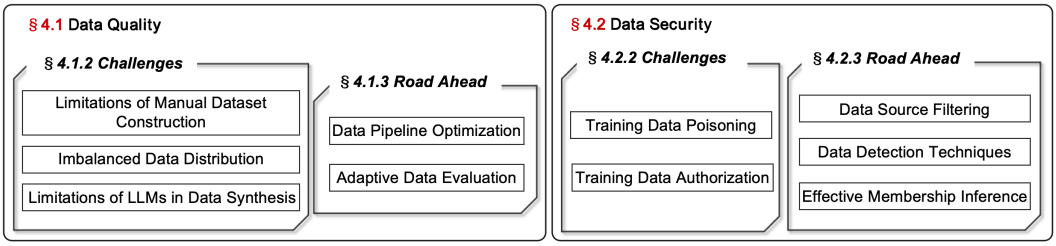

Fig. 4. Challenges and Road Ahead in §4 Dataset Construction.

According to the construction method, we broadly categorize current approaches to improving data quality into two main strategies: *(a)* manual data labeling and rule-based selection, and *(b)* LLM-assisted data construction. **Manual methods** typically yield high-quality datasets [45, 189, 319, 402] but are labor-intensive and often result in relatively small datasets. While **LLM-assisted methods** involve using LLMs to label, synthesize, or filter high-quality data automatically. Due to their strong performance, LLMs have been extensively employed for data construction [42, 214, 257, 366] to facilitate large-scale dataset generation through dedicated pipelines or agents. Correspondingly, there are two primary LLM-based data generation methods: **reference-based methods** and **collaborative LLM methods**. Reference-based methods typically leverage high-quality seed datasets, strong baseline models, or external knowledge sources as references to guide the generation of higher-quality datasets. For instance, Gao et al. [94] introduced a teacher-student framework where an LLM extracts high-quality samples from unlabeled data by comparing them against a reference dataset. Collaborative LLM methods involve multiple LLMs working together to identify high-quality datasets. For example, Liu et al. [223] fine-tuned datasets using human expert-created instructions to produce richer and more precise instruction datasets. Similarly, Huang et al. [132] employed LLMs to optimize fine-tuning datasets for improved code generation efficiency. Despite their advantages, these methods share a common limitation: their effectiveness is inherently constrained by the performance of the LLM itself. If the LLM's capability is suboptimal, the quality of the generated datasets is affected [431].

As an essential component of data quality, data diversity has a significant impact on model generalization and robustness. Thus, it has gained people's attention. Zhou et al. [464] demonstrated the presence of long-tail effects in datasets, where LLMs exhibit lower performance on rare data categories. Traditional approaches, such as Focal Loss [300] and Learning-to-Rank (LTR) [11], have been proven ineffective in mitigating these issues for LLMs.

Therefore, approaches for enhancing data diversity have been widely studied in recent years, which can be broadly classified into two categories: *(a)* preserving diverse samples during data selection and *(b)* employing data synthesis techniques. Although real-world data is inherently diverse [282], its imbalanced distribution poses significant challenges [227], such as minor languages remaining persistently underrepresented, which causes the related corpus to be significantly rare [159, 241]. Due to the high cost of manual efforts, researchers often rely on LLMs to select diverse data samples automatically. However, their capability to directly assess data diversity remains limited [273]. Consequently, researchers have turned to data synthesis techniques to enrich underrepresented categories [226]. Also, due to the high costs associated with manual data synthesis [220], recent efforts have focused on leveraging LLMs for automated data generation [42, 374, 393]. Yuan et al. [433] applied this method to synthesize biographical texts, reducing biases associated with occupation and improving dataset balance. Additionally, LLMs can serve as translators, converting

widely available data from natural languages [466], programming languages [22, 211], or multimedia formats [232] into less common languages, programming paradigms, or textual information. This approach generates rare data types at scale while minimizing human effort. However, due to inherent biases in LLMs, it may inadvertently introduce biases or errors [68, 397].

4.1.2 Challenges. The findings discussed above highlight several **challenges** in improving data quality, which can be categorized into three key aspects:

Limitations of manual dataset construction. Manual data creation is labor-intensive and inherently constrained in scale. A significant drawback is its time-consuming nature, as tasks such as dataset cleaning, labeling, and ensuring balanced data distribution require extensive human effort. Although pipelines and custom rules can assist in data collection and filtering, we still can not achieve full automation to process data. Furthermore, it is impossible to mitigate biases and achieve a well-balanced dataset distribution only through rule-based automation. As a result, the dependence on manual processes limits the speed and scalability of dataset development.

Imbalanced data distribution. Data imbalance is common in both real-world distributions and training dataset distributions. Different languages [159, 227], geographic regions, time periods [241], and data sources contribute to this imbalance, which complicates data acquisition and processing. As a result, imbalanced datasets introduce challenges: *(a)* The long-tail effect [464], where LLMs perform poorly on underrepresented data categories. *(b)* Inherent biases. When sources like social media dominate datasets, they often carry inherent biases [198]. These biases can harm model generalization [185] and raise ethical issues [150].

Limitations of LLMs in data synthesis. Despite their advancements, LLMs exhibit inherent limitations in improving data quality and generating synthetic data [431]. Since many of their abilities are still not comparable to human experts, fully relying on automation for data filtering, labeling, and evaluation is often ineffective. For instance, Pang et al. [273] found that LLMs struggle with accurately assessing data diversity. While LLMs have shown promise in data synthesis, their inherent biases can result in imbalanced synthetic data distributions [68, 397], further complicating the challenge of dataset construction.

4.1.3 Road Ahead. To enhance data quality, we propose two potential research directions. **Data pipeline optimization** is a promising direction. Specialized data pipelines or agents for collecting and processing LLM training data have been explored by Ostendorff et al. [269]. Furthermore, such architectures should incorporate automated tools to efficiently filter or generate target data based on user configurations. In addition to essential automation tools, LLMs can assist in this process. However, given the limitations of a single model, leveraging multi-model collaboration and multimodal data transformation can help overcome these constraints. Furthermore, incorporating human expert feedback can enhance data quality while mitigating the high costs associated with manual dataset construction.

At the same time, **adaptive data evaluation** can further improve data diversity by establishing robust evaluation mechanisms. Potential approaches include: *(a) Dynamic long-tail adaptation*, which adjusts data generation in real time based on distribution patterns to prevent imbalances. We can incorporate models such as support vector machines (SVMs) or clustering algorithms to automatically classify data and infer its distribution. However, data distribution detection faces two main challenges: how to define suitable classification criteria and how to determine the categories to be used. *(b) Multimodal contextual assessment*, which utilizes advanced multimodal translation techniques to generate data across different modalities based on evaluation results. This method facilitates cross-modal data transformation to enhance data diversity and improve overall data quality. *(c) Bias-aware diversity scoring frameworks.* Although LLM-as-a-judger has become increasingly popular, it remains unsuitable for reliably identifying bias. In the future, we could

build a clear and practical framework for evaluating bias, including clear categories of bias and corresponding assessment metrics. Within this framework, LLMs could serve as auxiliary tools to match outputs with relevant bias categories or similar cases. By analyzing the distribution of outputs within the same category, it would then be possible to determine whether a particular LLM response exhibits bias more accurately.

4.2 Data Security

4.2.1 Research Status. As discussed in §4.1, datasets may contain biased or harmful content, making LLMs susceptible to generating incorrect or unsafe outputs [280]. This issue poses significant risks in critical domains such as healthcare [10]. One major cause of this problem is data poisoning [144, 148, 398], where adversarially manipulated data is put into training datasets, leading to unintended behaviors in LLMs. While existing studies [270, 352] have explored methods to mitigate dataset bias, more advanced techniques are still needed to identify increasingly complex bias patterns.

Beyond biased data, the widespread adoption of LLMs for code generation has raised concerns regarding malicious code embedded in training datasets, which can lead LLMs to produce security vulnerabilities [46, 250]. Yan et al. [405] and Liu et al. [216] demonstrated that LLMs can synthesize vulnerable code capable of evading traditional static analysis tools as well as LLM-based vulnerability detection mechanisms. Although researchers have proposed countermeasures at different stages, including during code generation [178, 258] and post-generation analysis [401], the issue of preventing malicious code at the dataset level has received limited attention from researchers.

Additionally, since many LLM training datasets are not publicly disclosed, concerns have emerged regarding the use of unauthorized data, raising complex intellectual property and legal issues [391, 463]. To address the issue of training data authorization, researchers have explored membership inference techniques [38, 54, 238, 312] to determine whether a specific dataset was used in model training. For instance, Shi et al. [312] proposed a method to detect whether a dataset was incorporated into an LLM's training data by analyzing whether it was publicly released after the model's training period. However, this approach does not infer internal relationships within datasets. To overcome this limitation, Maini et al. [238] trained a linear model and used information scores to determine if a dataset was part of the model's training set. However, this method is still not precise enough. Another technique to address data authorization is digital watermarking [47, 108, 272, 371], which embeds markers to trace unauthorized usage and ensure data provenance. However, due to the current limitations of watermarking technology, such as requiring black box access rights [370] or being vulnerable to attacks [274], it still needs to be further improved.

4.2.2 Challenges. The aforementioned research findings highlight two key challenges in ensuring the security of training data: **training data poisoning** and **training data authorization**. Each aspect presents unique risks and requires targeted mitigation strategies to address them.

Training data poisoning. LLM training relies on large-scale datasets, making it highly susceptible to data poisoning attacks. Adversaries can deliberately inject malicious information, misleading content, or backdoor data into training sets to manipulate model behavior or even exert control over its generated outputs. These attacks include backdoor attacks, knowledge contamination, ethical and security pollution, and steganographic attacks. Even if a training dataset contains only a small fraction of malicious code, the trained LLM may still generate vulnerable code with high probability [280]. However, defending against data poisoning remains highly challenging due to LLMs' reliance on extensive, unverifiable datasets. While techniques such as poisoned data detection [17] and secure prompt engineering [413, 441, 459] have been developed to mitigate the impact of data poisoning, limited research has focused on systematically detecting and filtering malicious data within training datasets [51].

Training data authorization. Training data is often sourced from open datasets, web-scraped content, and user-provided data. However, not all data is explicitly authorized for commercial use or model training, posing legal risks related to copyright infringement, privacy compliance, and platform policies. Currently, membership inference and digital watermarking techniques are the primary methods used to address data authorization concerns. However, these techniques have their limitations. For example, some methods rely on disclosure time information or require black-box access to the model, which makes data verification less accurate and limits their applicability.

4.2.3 Road Ahead. Currently, no fully effective solution exists for mitigating training data poisoning [51], but several promising directions are worth further exploration. One potential approach is **data source filtering**, which involves establishing trusted data sources and exclusively collecting data from these sources. When combined with data provenance techniques to track the origin and modification history of data, this method enhances the traceability of datasets. However, it faces notable challenges, including ensuring the reliability of trusted sources and the limited availability of high-quality training data. Another promising direction is the advancement of **data detection techniques**. While some studies [46, 88, 148] have been conducted, more precise detection methods are required to identify malicious data, biased content, and trigger patterns. To address this, a data auditing and risk assessment platform could be developed, integrating automated tools for real-time or periodic security audits to identify data quality and security issues. Given LLMs' strong analytical capabilities and their successful applications across various domains [154], it is viable to leverage them for finer-grained detection. Specifically, a real-time anomaly detection system powered by LLM-assisted analysis could be designed to automatically identify and block anomalous data before it enters the training pipeline, thereby minimizing its potential negative impact.

To address data authorization concerns, integrating data provenance with blockchain technology represents a potential solution. However, storing large-scale datasets on-chain remains a significant challenge. Even if only dataset hashes are recorded on-chain, this approach becomes significantly less effective when there is little to no information available about the datasets used to train the LLMs. Thus, **effective membership inference** is essential. Although recent studies [239, 244] have made notable progress, they fall short of addressing deeper, structural challenges that remain at the core of the problem. The method proposed by Maini et al. [239] requires independently and identically distributed (IID) datasets, while the approach in Meeus et al. [244] struggles with small-scale inference and its accuracy remains limited. Future research should focus on enhancing membership inference methods to achieve higher accuracy and finer granularity.

5 DEVELOPMENT AND ENHANCEMENT

The development of LLMs has revolutionized the field of AI. However, it also introduces unique challenges from an SE perspective. Model enhancement aims to improve a model's capabilities, performance, and reliability. Notably, since models can be enhanced during development through fine-tuning or continual training, a strong coupling exists between the development and enhancement phases. Due to this interdependence, we explore the critical challenges associated with both LLM development and enhancement, focusing on three primary phases of model development: pre-training, fine-tuning, and model integration, as shown in Figure 5. We first analyze existing limitations and emerging solutions, followed by a discussion of key techniques: model compression and PEFT.

5.1 Pre-Training

5.1.1 Research Status. Pre-training is the foundation of LLM development, yet it faces challenges in terms of scale, resource demands, and engineering complexity. Ensuring and enhancing the

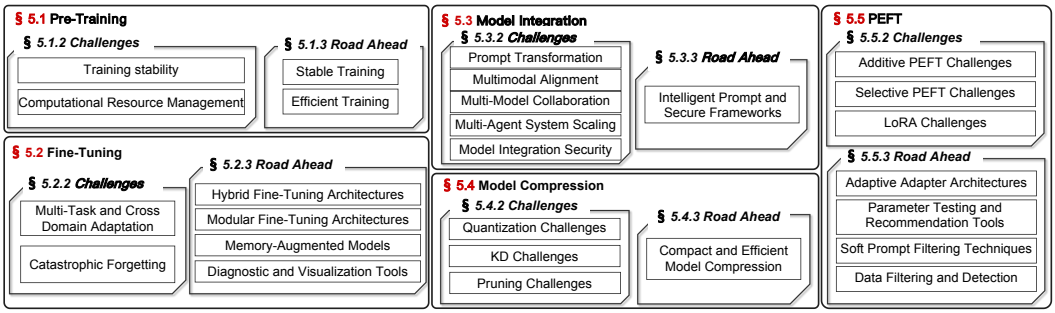

Fig. 5. Challenges and Road Ahead in §5 Development and Enhancement.

effectiveness of pre-training has been a long-standing research focus, with efforts directed towards optimizing training datasets, refining training methodologies, and improving computational efficiency [197, 342]. SE plays a critical role in advancing LLM pre-training, contributing to areas such as automating training pipelines [119, 229], optimizing training architectures [255], enhancing stability and security [34, 248, 368, 387], and reducing energy consumption [15].

As models continue to grow in size and complexity, maintaining training stability has become an increasingly critical challenge [337], particularly in ensuring smooth convergence. Existing research primarily focuses on mitigating gradient explosion [368] and gradient vanishing [248], as well as optimizing learning rates [34, 387], all of which aim to regulate parameter distributions and transformations during training to enhance stability. For instance, Chung et al. [43] controlled output layer embedding variance to prevent gradient explosion, while Agarwal et al. [3] and Woo et al. [377] improved stability by selectively discarding specific backpropagation steps. Nishida et al. [261] attributed loss spikes and convergence failures to uneven parameter distributions and introduced weight scaling as reparameterization (WeSaR) to normalize parameter norms for stable training. Similarly, Wortsman et al. [378] proposed a hybrid AdamW-Adafactor optimizer to mitigate loss spikes. Moreover, parameter precision also impacts training stability. DeepSeek-V3 [203] preserved the original precision of specific model components to ensure stable training.

The exponential growth in model sizes has further intensified computational resource constraints [16, 303, 310, 382, 394]. Training SOTA LLMs requires extensive computational infrastructure, which costs potentially reaching millions of dollars per training run. One commonly adopted approach to alleviate this challenge is model compression [470], which reduces model size to lower resource demands. However, these methods often result in some degree of performance degradation. We will discuss model compression techniques in detail in §5.4.

5.1.2 Challenges. The research mentioned above highlights two key challenges in pre-training:

Training stability. Although various techniques have been proposed to improve training stability, most approaches rely on heuristics rather than systematic frameworks. For instance, learning rate (LR) warm-up is commonly employed to mitigate gradient explosion and enhance stability. However, there is currently no general model for evaluating its effectiveness across different LLM architectures and training setups [390]. In practice, parameter settings are often chosen through trial and error, based on the needs of individual cases [109, 140]. Moreover, as model sizes continue to increase, the training process becomes increasingly complex and difficult to regulate; therefore, it is necessary to conduct systematic research into training stability.

Computational resource management. As LLMs continue to scale, managing computational resources has become a major challenge—training a single large model can cost millions of dollars

in hardware, energy, and time [67, 395]. This economic burden limits the capability of smaller research teams and companies with constrained resources to develop proprietary LLMs, limiting opportunities for smaller research teams to innovate and allowing a few major proprietary models to maintain their leading positions in the field. Lowering training costs would make it easier for smaller labs and open-source communities to build and improve their models, helping to reduce the dominance of a few major players. In the industry, the cost-efficient training approach adopted in DeepSeek-V3 [203] has already garnered significant attention, further facilitating the development of more accessible and cost-effective models.

5.1.3 Road Ahead. To address these challenges, potential solutions can be explored from two key directions: **stable training** and **efficient training**.

Stable training. One of the primary causes of training failures in LLMs lies in internal parameter updates. As discussed in §5.1.1, most existing approaches rely on heuristic adjustments rather than rigorous theoretical foundations. Zucchet et al. [476] explored optimization challenges in RNNs and identified fundamental causes of gradient explosion. Similarly, **achieving a deeper theoretical understanding of training dynamics in LLMs** is essential for identifying key instability factors, such as the underlying triggers of loss spikes. Beyond theoretical advancements, the development of **real-time monitoring and analysis tools** for LLM training could enable the detection and prediction of anomalous behaviors by tracking stability-related metrics. Additionally, it is important to design user-friendly visualization tools and interactive interfaces that help researchers interpret model behavior and monitor the training process more effectively.

Efficient training. Due to the high computational costs associated with pre-training, improving LLM efficiency is a crucial research direction. GPUs used for model training are extremely costly, which makes it essential to have tools that can monitor usage in real time and adjust workloads to avoid waste. **These tools should minimize GPU usage without increasing training time or compromising model performance**, thereby significantly reducing overall computational costs. Additionally, model growth techniques from ML in which smaller models are leveraged to accelerate the training of larger ones hold promise for improving efficiency. While this approach has not yet been widely adopted in LLM pre-training, Du et al. [65] conducted an empirical study providing insights into its potential application. Therefore, **it is considered valuable to explore the application of model growth techniques in LLM development in the future.**

5.2 Fine-Tuning

5.2.1 Research Status. Fine-tuning enables pre-trained LLMs to adapt to specific tasks while balancing adaptation, knowledge retention, computational efficiency, and deployment constraints. However, it presents several challenges, including PEFT (will be discussed in detail in §5.5), catastrophic forgetting prevention, and cross-domain generalization.

Fine-tuning has been widely employed to enhance model performance across diverse tasks [285, 306]. While fine-tuned models often exhibit significant improvements in individual tasks, effectively fine-tuning LLMs for multi-task scenarios remains a major challenge [204]. Existing approaches frequently struggle with task interference, optimal resource allocation across different objectives, and maintaining consistent performance across diverse domains [194, 341, 385, 388]. To address these challenges, researchers have explored techniques such as LoRA [6, 231] and MoE [163, 414, 469], further enhanced by optimization techniques [174, 355] and resource allocation strategies [215, 264]. However, one major challenge is improving task-specific performance while still preserving the model's general ability to work across different domains.

Additionally, fine-tuning introduces the risk of catastrophic forgetting, wherein models lose previously acquired knowledge during adaptation. Recent studies suggest that what appears to

be forgetting may be caused by the model becoming less aligned with the task, rather than truly losing the knowledge it learned [456]. However, the underlying causes are still unclear [179]. This problem becomes even more noticeable when the model needs to be updated regularly [111] or adapted to new domains [205], as it often struggles to retain its original capabilities while learning new ones.

LoRA has demonstrated potential in mitigating catastrophic forgetting and has achieved notable success [64, 259, 297]. However, recent studies indicate that LoRA still struggles with certain limitations, such as instruction-following constraints [146] and scaling challenges [155]. Continual learning approaches have been proposed to enable iterative knowledge acquisition while preventing forgetting [80, 325]. Nevertheless, research has shown that continual learning can lead to significant performance degradation after repeated training cycles [134, 176, 190]. Furthermore, these approaches are not universally applicable across different modalities in multimodal LLMs [437], underscoring the need for modality-specific fine-tuning solutions.

5.2.2 Challenges. Beyond PEFT, which will be discussed in §5.5, we identify two key challenges in fine-tuning.

Multi-task and cross-domain adaptation. Fine-tuning LLMs for multiple tasks or domains simultaneously presents several challenges [204], including task interference, optimal resource allocation, and maintaining consistent performance across diverse domains. While existing approaches such as LoRA and MoE have demonstrated effectiveness in mitigating these issues, there are still further improvements needed in areas such as training efficiency [364], model generalization [194], and system overhead during task switching [385].

Catastrophic forgetting. The underlying mechanisms behind catastrophic forgetting in LLMs remain largely unexplored, and no highly effective solutions have been developed to address this issue comprehensively. Although existing techniques, such as LoRA and continual learning, can help mitigate forgetting, they often come at the expense of performance degradation or the loss of other learned knowledge. Furthermore, these approaches are ineffective in multimodal models, which highlights the importance of developing fine-tuning methods tailored to each modality.

5.2.3 Road Ahead. The challenges of catastrophic forgetting and multi-task, multi-domain adaptation underscore the lack of universality in current fine-tuning methods, which have yet to achieve the goal of adapting to diverse tasks [285]. Future fine-tuning approaches should aim to enable both effective multi-task adaptation and mitigate catastrophic forgetting.

For **multi-task and multi-domain adaptation**, a promising direction is to integrate various fine-tuning techniques, such as LoRA, with MoE, quantization, and resource optimization strategies, forming a **hybrid fine-tuning architecture**. Specifically, designing an architecture capable of real-time monitoring of task interference could enable the rapid detection of adverse transfer effects, allowing for automated adjustments in task allocation and fine-tuning strategies. For instance, dynamically allocating additional resources to tasks with higher interference could mitigate performance degradation. This approach has the potential to minimize or eliminate task interference while preserving model generalization and maintaining inference efficiency. In addition to combining different fine-tuning methods, a **modular fine-tuning architecture** could be designed to make it easier for LLMser to switch between, plug in, or combine modules tailored to specific tasks or domains. Such an approach would reduce the coupling between fine-tuning techniques, thereby lowering maintenance costs and improving adaptability. Notably, this idea aligns with the design principles of industrial MoE and LoRA modules, which primarily focus on enhancing LLM performance. Recent studies on MoE and LoRA [156, 183, 403] have begun exploring intelligent scheduling mechanisms to support multi-task adaptation further.

For **catastrophic forgetting mitigation**, an essential research direction is the development of comprehensive methods to assess the extent and content of knowledge forgotten by LLMs. Such evaluations are crucial for facilitating targeted recovery of forgotten information and improving model retention strategies. Additionally, knowledge retention techniques tailored for multimodal models are urgently needed, as traditional PEFT and continual learning methods have demonstrated limited effectiveness in this context. Future research could explore **memory-augmented models** that leverage external memory mechanisms, such as knowledge graphs and vector databases, to reduce reliance on parameter-based memory. Furthermore, inspired by LR warm-up strategies, progressive adaptation [124] could be investigated as a gradual fine-tuning approach to prevent large learning rates from causing gradient explosion or vanishing, thereby mitigating the effects of forgetting. Another significant challenge is that catastrophic forgetting often remains undetected until post-fine-tuning evaluation, making real-time performance assessment difficult. This lack of visibility underscores the need for **diagnostic and visualization tools** specifically designed to monitor catastrophic forgetting. Future research could focus on building a diagnostic and visualization platform that helps developers track and understand how well knowledge is retained during the fine-tuning process. As Zheng et al. [456] point out, catastrophic forgetting does not mean that the model has truly lost the knowledge. With such a platform, developers can detect signs of forgetting on time, roll back to a previous version of the model, and adjust the training strategy for retraining. In addition, automated rehearsal mechanisms or adaptive prompting strategies [134] could also be integrated to proactively reduce the risk of forgetting.

5.3 Model Integration

5.3.1 Research Status. Unlike model development and enhancement through pre-training and fine-tuning, alternative approaches such as expanding a model's knowledge base via Retrieval-Augmented Generation (RAG) or knowledge graph techniques, developing multimodal models, and enabling multi-model collaboration focus on integrating external models or tools with base models, known as LLM-based agents, to accomplish more complex tasks. Collectively, we refer to these approaches as **model integration**, wherein SE plays a crucial role in multiple aspects, including transforming information into prompts, coordinating interactions between models or with external tools, and optimizing routing decisions.

In LLM integration, the inclusion of RAG, knowledge graphs, additional sensors, and multimodal base models introduces a diverse and extensive range of information sources [113]. Consequently, effectively transforming this information into prompts suitable for LLM task execution is important. Current research primarily focuses on preprocessing external information before generating prompts. For instance, Li et al. [192] proposed a method for summarizing contextual information before submitting it to the LLM. Similarly, in RAG, prompts are generated by combining retrieval-based methods with knowledge graphs to provide more task-relevant contextual knowledge [209, 242, 399]. Moreover, tools such as EasyTool [434] consolidate diverse tool-related information into unified interfaces for LLMs to process. However, these methods remain constrained by the limitations of the context window and the inherent capabilities of LLMs, preventing fundamental optimization. To address this issue, Koh et al. [164] proposed a more foundational approach, mapping textual information into the embedding space of vision models to enhance image representation. Nevertheless, this approach is still limited by the constraints of the model's embedding space. Therefore, to make real progress, new strategies are needed that go beyond the current methods.

Given the emergence of recent protocols such as A2A, MCP, the **Agent Communication Protocol (ACP)**[4], and the **Agent Network Protocol (ANP)**[5], which all emphasize communication and collaboration among models and tools, it is evident that the future focus of model integration

lies in multi-model, multimodal processing and interactive cooperation with external tools. Accordingly, we highlight three representative forms of model integration: **multimodal models**, **multi-model collaboration**, and **LLM-based agents**. Multimodal models are capable of receiving inputs from various modalities and producing outputs across multiple modalities. Multi-model collaboration refers to leveraging cooperation, competition, or cascading among different models to accomplish complex reasoning tasks. LLM-based agents, in contrast, are built upon LLMs as the central reasoning component, and are capable of planning, decision-making, and executing tasks by interacting with external tools and knowledge sources. As illustrated in Figure 6, although these three paradigms differ in focus, there is overlap among them. For instance, in multi-model collaboration, complex tasks can be decomposed and distributed across models via multi-agent systems [82]. LLM-based agents can integrate multimodal models to mitigate hallucinations and enhance reasoning capabilities [145], or dynamically select from a pool of multimodal models of varying types and sizes to suit different task requirements [440]. For clarity, we analyze these three integration paradigms separately, focusing on their unique characteristics and technical challenges, while leaving their areas of overlap outside the scope of this discussion.

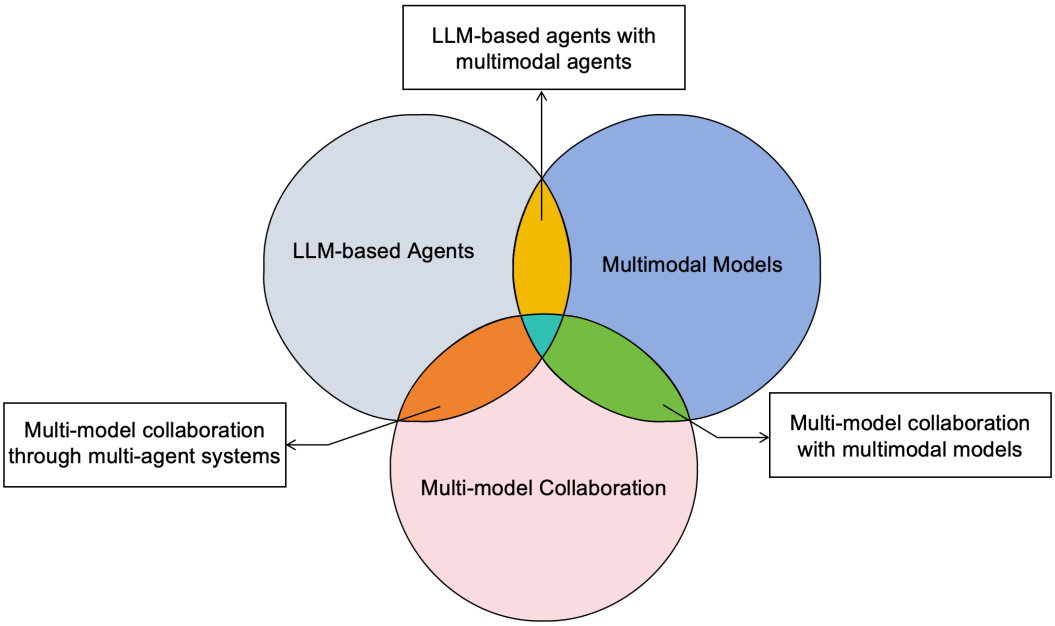

Fig. 6. Overlap between LLM-based agents, multimodal models, and multi-model collaboration.

For multimodal models, a fundamental challenge lies in achieving effective alignment across different modalities (e.g., text, vision, and audio) [35]. This challenge involves several key aspects. First, semantic consistency ensures that meaning is preserved across modalities [187, 367]. Second, representation alignment focuses on aligning embeddings and features from different modalities to support better understanding and knowledge sharing [173, 305]. Third, cross-modal understanding aims to bridge the gaps between modalities, enabling smoother knowledge transfer and more effective interaction [139]. Liu et al. [207] highlight that failure to address these issues can result in severe performance degradation or hallucination effects. Furthermore, these challenges extend across various stages of multimodal model development, including data processing [196, 460] and

pre-training strategies [195], both of which face significant hurdles. Closing these gaps is crucial for developing multimodal models that can perform effectively across a broad range of real-world tasks.

In contrast, multi-model collaboration has been explored as a means of enhancing reasoning capabilities [213, 278, 309]. However, differences in what models can do, how they respond, and the values they reflect make it necessary to align them carefully to ensure they can work well together. Without such alignment, issues such as hallucinations may mislead reasoning processes, potentially resulting in incorrect or failed outcomes [83, 84]. Although several approaches have been proposed to mitigate these challenges, such as fine-tuning [396], uncertainty estimation [428], and probing other LLMs to address knowledge gaps [83], there still lacks enough research in this area. Another significant challenge lies in the coordination of multi-model systems, which includes managing workflows, optimizing inter-model communication, and efficiently allocating computational resources. While novel routing strategies have been introduced, such as MARS [129], TO-Router [330], Eagle [455], and C2MAB-V [49], these approaches often fail to consider critical real-world constraints, such as computational resources and network bandwidth. In practice, these resource limitations introduce additional complexities [279], which will be further discussed in §7.

Another common integration form is the **LLM-based agent**. Leveraging the powerful reasoning and decision-making capabilities of LLMs, LLM-based agents can autonomously perceive their environment, adapt to changes, and take actions when interacting with external systems, such as web services, databases, or local files. As a result, they have been widely applied in different areas [33, 133, 165, 237, 356]. Currently, LLM-based agent frameworks typically consist of four core modules: plan, perception, memory, and action [39, 210, 230, 383]. Specifically, the plan module is responsible for strategy formulation and execution planning, the perception module handles environmental input sensing and its transformation into representations that LLMs can understand, the memory module stores historical information to support context retrieval and ensure coherent decision-making, and the action module executes specific operations and tool invocations.

Although single-agent systems have shown impressive capabilities, they often struggle to handle complex tasks that require diverse skills, parallel processing, or coordinated decision-making [210]. To address these challenges and scale to more sophisticated problems, **multi-agent systems (MAS)** are explored, enabling division of labor and collaborative problem-solving by coordinating the efforts of multiple autonomous agents [39, 230], which requires robust agent-to-agent communication and collaboration mechanisms between the agents themselves. Protocols designed for agent-to-agent communication facilitate such inter-agent collaboration, enabling agents to discover each other's capabilities, delegate tasks, and exchange information. For instance, the **A2A** [335] specifically supports peer-like task outsourcing and dynamic interaction between autonomous agents, often within enterprise-scale workflows [71]. Beyond inter-agent communication, a fundamental requirement for both single and multi-agent systems is effective interaction with diverse external tools and resources for task execution and data retrieval. However, it is a significant challenge to standardize this agent-to-external system interaction. Additionally, existing tool integration methods are often fragmented, relying on complex manual wiring and platform-specific approaches that limit scalability and interoperability [39, 127, 230]. To address this fragmentation and standardize AI model-to-external system interaction, the **MCP** [14] introduces a unified communication framework for LLMs to interact with external tools and resources, which simplifies tool invocation and enhances interoperability across diverse systems. Although introduced only recently, MCP has gained significant attention [127, 317]. It enables LLM-based agents to interact with external tools and systems more easily through a unified interface, facilitating the completion of complex tasks more efficiently.

Another critical issue in model integration that has garnered significant attention is security. Due to the diverse and complex nature of real-world scenarios, even LLMs that have undergone safety alignment remain vulnerable to prompt injection attacks [102, 221]. Suo et al. [334] proposed Signed-Prompt as a defense against sensitive prompt injection attacks. However, Liu et al. [217] identified an attack technique that exploits prompts to achieve remote code execution (RCE). Similarly, Evertz et al. [73] outlined two methods for extracting confidential information through malicious prompts: *(a)* inducing the model to leak data by disguising malicious input as legitimate user queries and *(b)* injecting malicious data into external tools to hijack the model's behavior, leading it to execute unsafe operations. These examples demonstrate the vulnerability of prompt environments in model integration settings, as observed by Zhan et al. [438]. Although various studies have proposed effective defense mechanisms, including benchmarks [222, 286, 438], fine-tuning [286], prompt filtering [281], and LLM-based defenses [286, 458], recent studies [55, 138, 172, 214, 276] highlight their limitations, which underscores the need for further investigation into emerging attack vectors and the development of more robust security strategies to safeguard LLMs in the real world.

In addition to direct attacks targeting the models themselves, the communication protocols facilitating model integration and agent interaction can also become potential entry points for security threats. Protocols such as the **MCP** (for agent-tool interaction), **ACP**, **A2A**, and **ANP** (for inter-agent communication in various contexts) each introduce specific security risks throughout their lifecycle. For instance, MCP servers, which mediate agent access to external tools, face risks including code injection, backdoor implantation, and installer spoofing, which may lead to information leakage or the incorrect execution of actions by LLMs [127]. Radosevich et al. [292] identified that MCP servers are particularly vulnerable to malicious code execution (MCE), remote access control (RAC), and credential theft (CT). Protocols like A2A, ACP, and ANP, designed for communication between agents, face distinct challenges such as identity spoofing (e.g., Agent Card spoofing in A2A, DID spoofing in ANP), message tampering, unauthorized capability injection, and session hijacking, impacting secure task delegation and coordination [71]. To mitigate these protocol-specific vulnerabilities, research is exploring various defense mechanisms tailored to each protocol's interaction model and lifecycle, including authentication, auditing, secure configuration strategies based on theoretical threat analysis, cryptographic signing of manifests and messages, and robust access control mechanisms [71, 168, 254]. Although research on the security of these emerging protocols remains limited, existing studies demonstrate that their security issues should not be overlooked, and future work should place greater emphasis on addressing these challenges across the diverse landscape of agent interoperability protocols.

5.3.2 Challenges. We categorize the key challenges in model integration into four main aspects: **prompt transformation**, **alignment in multimodal systems**, **multi-model collaboration**, and **security concerns**.

Prompt transformation. In both multimodal settings and LLM integration with external models and tools, prompts are no longer limited to a single information source. As user input environments become increasingly complex, inputs may include images, audio, text, and so on, which indicates they need to be converted into a common modality (e.g., image-to-audio transformation). Similarly, RAG and knowledge graphs contain vast amounts of structured and unstructured data, requiring the extraction of relevant content for prompt construction. Consequently, transforming diverse information into an input format that LLMs can effectively process is crucial.

Multimodal alignment. Achieving effective alignment across different modalities (e.g., text, vision, and audio) remains a fundamental challenge in multimodal models. Failure to accurately transform and synchronize information between modalities can lead to severe performance degradation or hallucination effects. This alignment process requires not only semantic consistency but

also the synchronization of internal representation spaces to facilitate cross-modal understanding. While significant research has been conducted on aligning visual and textual information, studies on other data types, such as sensor data and time-series information, remain underexplored.

Multi-model collaboration. The challenges associated with multi-model collaboration can be broadly categorized into two key aspects. First, differences in model capabilities, preferences, and underlying values must be aligned. Otherwise, discrepancies in understanding may lead to reasoning errors during collaboration. Second, coordination challenges arise in workflow management, inter-model communication, and the allocation of computational resources. For instance, assigning simpler tasks to less resource-intensive models while reserving complex tasks for more powerful models can improve inference efficiency and reduce computational costs. However, real-world implementations involve additional complexities, such as quantifying task difficulty, designing model selection strategies, and managing concurrent execution across models. Addressing these challenges requires significant advancements in multi-model coordination and scheduling mechanisms.

Multi-agent system scaling. During the scaling of multi-agent systems (MAS), two primary challenges arise: the rapid increase in computational resource demands and the significant growth in complexity of communication and coordination. When the number of agents grows, the system becomes more resource-intensive. This is because even a single LLM consumes a lot of resources, and adding more agents means extra computational and storage overhead for each one. Additionally, the complexity of communication and collaboration also escalates rapidly with system expansion. Since agents have autonomous decision-making and execution capabilities, ensuring the correctness and consistency of their decisions becomes increasingly complex, exhibiting a nonlinear and potentially exponential growth in complexity. Furthermore, cross-agent communication, complex decision-making, task decomposition, and scheduling become critical and significantly more challenging in large-scale systems. Thus, controlling computational and collaboration costs while scaling remains a fundamental issue that must be addressed in the design of multi-agent systems.

Model integration security. A primary security threat in model integration is prompt injection attacks, which can be exploited to extract sensitive information, inject misleading content, or manipulate models into executing unintended actions. While model integration enhances overall system capabilities, it also introduces the potential risks associated with such attacks, underscoring the necessity for robust defense mechanisms. In addition to prompt injection targeting the models, the various interaction protocols used in model integration also introduce new attack surfaces. Protocols standardizing agent-to-external system communication, such as MCP, are susceptible to risks like code injection, remote access control, and credential theft within their server implementations and communication channels, which can result in information leakage or incorrect execution of actions by LLMs. Protocols facilitating agent-to-agent or multi-model communication (e.g., A2A, ACP, ANP) face distinct security challenges related to ensuring message authenticity, authorizing interactions between peers, and preventing malicious coordination. Although research on the security of these emerging protocols is still in development, existing studies highlight the need to develop appropriate authentication, auditing, and configuration mechanisms tailored to each protocol's specific interaction model to mitigate vulnerabilities across the integrated system.

5.3.3 Road Ahead. Although the challenges mentioned above may appear distinct, they are inherently interconnected due to the nature of model integration, which involves coordinating multiple models or integrating models with external tools. To collectively address these issues, we propose an **intelligent prompt and secure framework**. This framework consists of three core modules: a **prompt module**, a **routing module**, and a **security module**, each of which presents opportunities for future research and advancements.

Prompt module. This module serves two primary functions: **prompt generation** and **prompt filtering**. The prompt generation process must facilitate cross-modal information transformation, extract relevant knowledge from external sources, and ultimately generate optimized input prompts. To achieve this, the module must incorporate system monitoring, contextual understanding, and adaptive decision-making. Furthermore, to ensure effective multimodal alignment, the framework should align diverse data types [164] with LLM representation spaces and potentially leverage token-free Transformer architectures [271] to enhance expressive capacity. In addition to prompt generation, robust **prompt filtering** mechanisms are essential for preventing prompt injection attacks. The module must filter potentially malicious or biased prompts before processing, whether they originate from external sources or are generated by the LLM itself. Such defenses are particularly critical in mitigating advanced attack strategies, as identified by Lee et al. [172], where LLM-generated outputs can inadvertently function as adversarial prompts. To achieve this, advanced content-aware techniques are needed to detect and filter malicious or biased content, either based on predefined rules, LLM outputs, or patterns learned by LLMs from data.

Routing module. The routing module is responsible for orchestrating LLMs and other subsystems within the framework. It must perform task decomposition, analyze task complexity, generate execution pathways, and select appropriate models or subsystems to execute tasks either in parallel or sequentially. Additionally, it dynamically manages computational resources, enabling efficient scheduling and adaptive reasoning across multi-model, multimodal, and multi-agent systems, thereby enhancing the model integration framework's overall performance and problem-solving capabilities. To achieve these objectives, different protocols and techniques are needed depending on the nature of the interaction.

For standardizing the interaction between an AI model (agent) and external tools, data sources, or services, often referred to as agent-tool invocation, leveraging the **MCP** is key. MCP provides a unified interface standard for accessing tools, resources, and prompts, enabling the efficient management of tool and model invocation processes. This standardizes the flow of requests from agents to external capabilities, allowing agents to seamlessly integrate diverse external functionalities by simply adhering to the protocol. For coordination and collaboration between multiple agents or models, often referred to as agent-to-agent or multi-model communication, specific communication protocols are required to handle message exchange, task delegation, negotiation, and collaborative execution. Protocols like **A2A**, **ACP**, and **ANP** are designed for various forms of inter-agent communication, providing frameworks for performative messaging, capability discovery, and secure peer interaction in different deployment contexts (e.g., within trusted organizational boundaries for A2A, brokered communication for ACP, decentralized open networks for ANP). These protocols are crucial for mitigating semantic inconsistencies and information transmission errors resulting from heterogeneous communication methods among interacting agents or models, thereby enabling the scalable coordination of multi-agent systems. Beyond communication protocols, achieving robust coordination in multi-agent systems requires sophisticated scheduling and collaboration mechanisms. Techniques from **swarm intelligence** such as bee algorithm, ant colony optimization (ACO), and particle swarm optimization (PSO), as well as multi-agent game-theoretic models from **game theory**, can be adopted to optimize task allocation, cooperation strategies, and autonomous decision-making among agents (or models) within large-scale environments. These approaches complement the communication protocols by providing the intelligence layer for complex multi-agent coordination, thus driving LLM-based multi-agent systems toward greater efficiency, scalability, and resource optimization.

Security module. Closely interacting with both the prompt and routing modules, the security module acts as a critical protective layer against various threats in the integrated system. It mitigates direct model risks such as prompt injection through techniques including self-supervised anomaly

detection, static analysis, sandboxed inference, and reinforcement learning-driven adversarial defense. Furthermore, it supports data isolation and secure interaction mechanisms within complex workflows (including multi-model or multi-agent collaboration), preventing sensitive information leakage and inhibiting the propagation of attacks between components.

Besides, securing the diverse interactions within this framework relies heavily on addressing vulnerabilities in the underlying communication protocols. For **agent-to-external system interactions** standardized by protocols like MCP, security concerns focus on securing the channel between the agent and the tool server, which involves mitigating risks such as code injection, remote access control, credential theft, and ensuring the integrity of tool invocations and data exchange [127, 292]. To address these concerns, the security module can incorporate measures such as firewall-based server protection, access control, and authentication for tool usage, as well as logging and auditing of tool interactions [168, 254]. For **agent-to-agent communication and collaboration** facilitated by protocols such as ACP, A2A, and ANP, security measures are needed to ensure trustworthy interactions between autonomous agents. Challenges here include verifying agent identity, ensuring message authenticity and integrity, authorizing peer-to-peer actions, preventing unauthorized capability injection, and securing the coordination process itself. These protocols incorporate mechanisms like digital signatures, strong authentication (e.g., DIDs, mutual TLS), access control policies for agent skills, and secure session management [71]. The security module can integrate and manage these protocol-specific security features to provide threat detection and defense across all interaction types within the integrated system.

5.4 Model Compression

5.4.1 Research Status. Model compression aims to reduce the number of parameters or the memory footprint of a model. We categorize it into three main approaches: **quantization**, **KD**, and **pruning**, the same as [29, 206, 470].

Quantization techniques reduce the bit-width of LLM parameters to lower memory usage. Post-training quantization methods typically target weights, activations, or KV cache, using either floating-point or integer formats to reduce runtime memory requirements significantly. For instance, Dettmers et al. [56] compressed both weights and activations to 8-bit integers. More aggressive approaches, such as QuIP [30] and KIVI [225], further reduce weights and KV cache to as few as 2 bits. To enhance compression effectiveness, methods such as COMET [212] and QServe [201] simultaneously quantize weights, activations, and KV cache. However, aggressive quantization often results in significant performance degradation. To address this issue, researchers have explored retraining quantized models to mitigate performance loss, even if there is a cost of additional training overhead. Techniques such as LLM-QAT [224], L4Q [143], and QLoRA [57] integrate KD or PEFT to alleviate this overhead. L4Q [143] achieves post-training performance comparable to fine-tuned models but is limited to weight quantization, restricting its overall compression potential. Additionally, Huang et al. [136] reported severe performance degradation in quantized versions of the LLaMA3 model, highlighting the limitations of current quantization techniques in balancing model compactness and performance. These findings underscore the ongoing need for advancements in quantization methodologies to ensure both efficiency and effectiveness.

KD aims to transfer knowledge from a more capable teacher model (TM) to a smaller student model (SM), enabling the latter to achieve comparable performance with reduced computational costs. The challenges in KD can be broadly categorized into two key aspects: *(a)* how to extract knowledge and *(b)* how to learn the extracted knowledge effectively. A straightforward approach to knowledge extraction is to provide the TM with input data and use its outputs as distilled knowledge [283, 452, 465]. However, this method heavily depends on the capabilities of the TM [465] and is constrained by the diversity and scale of the instruction set [186], potentially leading to

poor generalization in the SM [314, 315]. To address these limitations, researchers have integrated data cleaning with synthetic data generation to improve instruction set quality [60, 360, 432]. Additionally, self-knowledge techniques, wherein the SM generates new knowledge without relying on the TM, have been explored [37, 409]. However, self-knowledge methods are susceptible to inherent biases and hallucinations [326, 406], posing significant challenges to their effectiveness. Instruction following (IF) is a widely used technique in KD, yet it faces limitations, such as dependency on high-quality datasets [186] and difficulty in capturing the TM's reasoning process. To overcome these challenges, techniques like chain-of-thought (CoT) prompting [260, 362] have been employed to enhance the SM's reasoning capabilities. Recently, Choi et al. [40] proposed a method that integrates LLM reasoning decomposition and planning capabilities with knowledge graph-augmented reasoning. Their approach, implemented in the lightweight framework DeDer, successfully distills reasoning skills into a compact model, demonstrating the potential of KD to enable resource-constrained devices to perform complex tasks.

Pruning aims to enhance model efficiency and reduce computational overhead by removing redundant neurons or weights. It can be categorized into structured pruning, semi-structured pruning, and unstructured pruning. Structured pruning methods focus on removing entire structures, such as neurons, channels, or attention heads, to maintain computational efficiency. Ma et al. [235] proposed a notable structured pruning approach, LLM-Pruner, which constructs a structural dependency graph of the LLM, groups parameters accordingly, identifies unimportant groups for pruning, and subsequently restores model performance using LoRA. This method requires only 590k samples and three hours of training. However, combining structured pruning with PEFT introduces additional training overhead and can lead to performance degradation [453]. To mitigate these issues, Zhao et al. [448] introduced an adaptive pruning strategy, which removes parameters irrelevant to fine-tuning tasks from the pretrained model while incorporating task-specific parameters via distillation, thereby improving LLM performance with reduced computational overhead. Semi-structured pruning provides a balance between flexibility and efficiency by enforcing structured sparsity patterns at the matrix level. NVIDIA introduced a 2:4 structured sparsity technique, which retains 2 out of every 4 weights in matrix computations, effectively accelerating inference [247]. However, this approach can inadvertently remove critical weights, resulting in a noticeable degradation of accuracy [336]. To address it, Tan et al. [336] proposed a method that selectively retains essential weights while maintaining the 2:4 sparsity ratio, thereby improving accuracy retention.

5.4.2 Challenges. The aforementioned discussion on model compression highlights several key challenges associated with **quantization**, **KD**, and **pruning**.

For **quantization challenges**, while existing techniques effectively reduce memory consumption, they often lead to performance degradation, posing a fundamental trade-off between compression efficiency and model accuracy. Balancing these two aspects remains an open challenge. Recent advancements have pushed parameter bit-width to its lower limits, with some approaches reducing it to as few as 2 bits. However, the precise relationship between quantization levels and performance degradation remains unclear. Several studies [98, 135, 136] have conducted empirical analyses on the impact of quantization on model performance. However, current investigations remain insufficient, particularly in open-source models such as the LLaMA family. As noted by Jin et al. [153] and Yao et al. [422], existing research primarily focuses on a limited range of models and quantization techniques, leaving significant gaps in understanding the broader implications of extreme quantization.

Regarding **KD challenges**, the primary challenges lie in both knowledge extraction and knowledge learning. For knowledge extraction, direct input-based methods offer a straightforward

approach; however, they often yield suboptimal training outcomes and weak generalization in SMs. Alternative strategies, such as self-knowledge and instruction following, are hindered by biases, hallucinations, and dataset limitations. In terms of knowledge learning, acquiring abstract capabilities such as reasoning and generalization remains a significant challenge. While CoT prompting has been employed to enhance reasoning abilities in SMs, research on improving other abstract skills remains scarce, highlighting an area for further exploration.

Finally, **pruning** presents challenges primarily in parameter selection. The removal of essential parameters can lead to severe performance degradation, yet accurately distinguishing between critical and redundant parameters remains an open research problem. Developing more reliable pruning criteria and adaptive selection mechanisms is crucial to mitigating the risks associated with aggressive parameter reduction.

5.4.3 Road Ahead. The challenges mentioned above highlight the key limitations in model compression. While these challenges are primarily studied within the domain of ML, they can also be addressed from an SE perspective. Overall, achieving **compact and efficient model compression** remains a critical objective for future advancements, necessitating deep optimization of quantization, KD, and pruning, as well as their integration with other optimization techniques to achieve complementary benefits.

Quantization. Existing quantization techniques struggle to simultaneously support weight, KV cache, and activation quantization while maintaining model performance. If all three components can be effectively quantized while leveraging KD or PEFT to recover performance losses, it would enable a better trade-off between computational efficiency and model compactness. Additionally, conducting a comprehensive empirical study on the impact of different quantization levels on model performance could provide valuable insights for future research. Further investigation is needed to assess the effects of quantization across various downstream tasks, quantization techniques, and model sizes. Developing an automated evaluation framework or plugins for quantization impact that is capable of fine-grained performance loss analysis and automated optimization recommendations would be a significant step forward.

KD. Beyond single-model KD, future research should explore the efficient integration of knowledge across multiple models and diverse sources. Given the dynamic and multi-source nature of real-world knowledge, designing a distributed KD training framework that supports multi-node collaboration and asynchronous updates could significantly enhance LLMs' generalization and knowledge update capabilities. Such a framework would be particularly beneficial in federated learning scenarios, enabling efficient KD in decentralized environments. However, despite extensive research in this area [275, 290], several engineering challenges remain, including issues related to data heterogeneity [131, 418, 442, 472], device heterogeneity [252, 318], and high communication costs [90, 91]. Furthermore, to incentivize knowledge sharing among different nodes, blockchain technology could be integrated into the framework, which would not only provide a mechanism for incentivization but also enable knowledge traceability, mitigating the risks associated with malicious nodes injecting harmful information.

Pruning. Semi-structured pruning techniques have demonstrated promising potential, and future research may focus on intelligent weight selection, which requires a deeper understanding of which weights and knowledge are essential for model functionality, which could potentially leverage loss functions, activation functions, or other indicators. Zhang et al. [444] explored the use of loss functions to distinguish between domain-specific and general knowledge, paving the way for more precise pruning strategies. Additionally, integrating model testing techniques could enhance pruning effectiveness by monitoring performance, which would help prevent severe degradation

or the emergence of errors, thereby enabling dynamic pruning adjustments to maintain model reliability while maximizing efficiency.

5.5 PEFT

5.5.1 Research Status. PEFT is a technique that updates only a subset of a model's parameters during fine-tuning, thereby achieving high efficiency while avoiding full-parameter modifications. We categorize PEFT methods into three main types: **additive**, **reparameterized**, and **selective** approaches, the same as [114, 389].

Additive PEFT techniques preserve the original model parameters while introducing additional trainable parameters to adapt the model to downstream tasks. Representative methods in this category include adapter layers and soft prompt. The trained adapter modules are inserted into the model as additional Transformer layers, reducing the number of modified parameters. Based on the insertion strategy, adapters can be classified into sequential adapters and parallel adapters. Sequential adapters primarily focus on adapting to specific tasks, while parallel adapters combine the outputs of both the adapter and the main model, making them better suited for complex scenarios [161]. However, adapters introduce additional modules, increasing model complexity, maintenance overhead, and inference latency [302]. In contrast, the soft prompt method appends a set of learnable vectors, aligned with the embedding layer, to the input prompt. These vectors guide the LLM to perform downstream tasks more effectively without modifying the model architecture [354]. Unlike adapters, soft prompt avoid additional structural complexity and inference overhead and can be transferred between different models and tasks [351, 400]. However, soft prompt do not fundamentally enhance the model's capabilities, as they primarily rely on the LLM's inherent reasoning capabilities [365]. Moreover, soft prompt are vulnerable to adversarial attacks, particularly prompt injection attacks [258, 419]. Compared to standard prompt-based attacks, soft prompt manipulations are more likely to bypass a model's safety alignment mechanisms and induce unintended behaviors [308, 415]. For example, the malicious soft prompt can lead to unintended data leakage, including the inadvertent exposure of sensitive information from the training corpus [162].

Selective PEFT fine-tunes a model by masking a portion of its parameters, similar to pruning. It can be broadly categorized into structured masking and unstructured masking, both of which aim to identify an optimal subset of parameters for fine-tuning. Representative approaches include gradient-based methods [31, 181, 411], data-driven methods [52, 63], and search-based methods [25, 462]. The primary advantage of selective PEFT is that it does not increase the inference cost of the LLM, making it an attractive alternative to other PEFT techniques. However, the complexity of parameter selection strategies introduces significant challenges in model development and debugging. Additionally, Ploner et al. [287] observed that randomly selected parameter subsets such as those employed in LoRA often yield performance comparable to carefully designed selection strategies. Their findings raise questions regarding the practical benefits of extensive debugging efforts, given the marginal improvements achieved over random parameter selection.

Reparameterized PEFT modifies the model's parameterization to enable more efficient adaptation. Among these methods, LoRA is a widely adopted approach. LoRA employs low-rank decomposition to train a separate module, which is then used to reparameterize specific model weights. A single model can incorporate multiple LoRA-trained modules, allowing for the selection of different module combinations during inference based on specific requirements. Despite its efficiency, LoRA presents two primary challenges: improving LoRA's performance and selecting appropriate LoRA modules. *(a) Performance optimization.* Numerous techniques have been proposed to enhance LoRA's effectiveness, including dynamic rank adjustment [346, 445], earning rate optimization [118], and regularization strategies to mitigate overfitting [200, 359]. However, the extent of these improvements remains constrained. Zhang et al. [439] found that LoRA's performance

is predominantly influenced by the inherent capabilities of the base model, suggesting that optimizations at the LoRA level offer only limited benefits. *(b) Module selection and inference efficiency.* Optimizing LoRA module selection can significantly reduce LLM inference latency and enhance overall performance. For instance, Kong et al. [166] observed that the insertion of LoRA modules leads to fragmented CUDA kernel calls, severely degrading inference efficiency. To address this, they proposed a novel token-wise routing strategy to minimize unnecessary kernel invocations. Similarly, Wu et al. [379] introduced a dynamic switching mechanism between merged and unmerged models to reduce inference latency in model as a service (MaaS) scenarios. Their approach further integrates batching techniques and a request-adapter co-migration strategy to improve GPU resource utilization and overall service performance. *(c) security considerations.* Despite its advantages, LoRA introduces security concerns due to the additional fine-tuning it requires. Liu et al. [208] demonstrated that open-source LoRA adapters are vulnerable to backdoor attacks. Moreover, Xu et al. [128] highlighted that even when training datasets do not contain malicious data, aligned LLMs remain susceptible to adversarial threats. To mitigate these risks, they proposed Safe LoRA, which constrains LoRA updates using a projection operation, ensuring that parameter updates align with a predefined security-preserving matrix, thereby enhancing robustness against adversarial manipulations.

5.5.2 Challenges. The research mentioned above highlights several key challenges associated with PEFT. We categorize these challenges as follows:

Additive PEFT challenges. While adapter layers facilitate seamless integration with LLMs, they also introduce additional complexity in system maintenance. The selection, combination, and interconnection of different adapter layers pose significant engineering challenges, yet they also present opportunities for further advancements. Moreover, the insertion of adapter layers inevitably increases inference latency, which can be detrimental to latency-sensitive applications. In contrast, soft prompt mitigate these performance concerns but raise security and privacy risks, as they may inadvertently expose sensitive or private data from the training process.

Selective PEFT challenges. The primary challenge in selective PEFT lies in selecting the appropriate parameters. Simple selection strategies may fail to achieve optimal fine-tuning results, whereas more sophisticated selection mechanisms not only introduce additional development and debugging overhead but also do not necessarily outperform random selection, which raises concerns regarding the feasibility and practical benefits of selective PEFT.

LoRA challenges. As a representative reparameterized PEFT method, LoRA enables efficient fine-tuning but faces limitations related to inference efficiency and security vulnerabilities. The integration of LoRA could lead to fragmented CUDA kernel calls, thereby reducing inference efficiency. Additionally, LoRA-trained adapters have been demonstrated to be susceptible to backdoor attacks. Although ongoing research aims to enhance LoRA's robustness and efficiency, substantial challenges remain, limiting its applicability in security-critical and latency-sensitive scenarios.

5.5.3 Road Ahead. To address the challenges mentioned above, there are several research directions for further exploration. For challenges associated with adapter layer insertion, an **adaptive adapter architecture** could be developed to facilitate adapter selection, composition, and integration. Hu et al. [130] proposed a broad adapter integration framework. However, their approach lacks considerations for composition design and optimization at deployment. Future research could focus on automated optimization, modularization, and standardization of adaptive adapter architectures. For instance, standardized APIs for adapter layers, dynamic adapter loading mechanisms that activate specific adapters only when necessary, and optimized caching strategies could significantly enhance inference efficiency. Additionally, automatic search techniques could be employed to determine the optimal adapter combinations, adapter depth, and activation strategies. While such

techniques have been extensively studied in vision models [426, 429, 467], their application in LLMs remains underexplored. Notably, research on adaptive architectures is also applicable to LoRA, since it can be seen as a special form of adapter that modifies the original Transformer's weights instead of introducing additional layers. For example, Wu et al. [379] proposed a method for dynamically merging and migrating LoRA adapters to enhance the throughput of LLM servers.

For challenges related to parameter selection in selective PEFT, similar to pruning, **parameter testing and recommendation tools** could be leveraged to evaluate the impact of different hyperparameter configurations on model performance, which would enable the development of more effective parameter selection strategies while also facilitating the identification of optimal hyperparameter combinations.

PEFT security presents two key challenges: *(a)* soft prompt may expose private data, and *(b)* adapters (including LoRA adapters) are vulnerable to backdoor attacks. To mitigate risks associated with soft prompt, **soft prompt filtering techniques**, as discussed in §5.3.3, could be extended to prevent inadvertent data leakage. These techniques must ensure that filtered prompts retain both security and semantic fidelity. However, unlike standard prompt-based attacks, soft prompt manipulate the embedding layer, making them more challenging to defend against using traditional LLM security mechanisms [308]. Therefore, further research on protective measures for embedding layers is necessary. Regarding backdoor vulnerabilities in adapters, a potential solution could involve **data filtering and detection** inspired by existing research on mitigating data poisoning in open-source models. For instance, verifying whether LoRA's training data has been compromised could leverage data provenance techniques, such as traceability analysis and trusted data sources, as outlined in §4.2.3. Additionally, adversarial training could be employed to enhance the robustness of LoRA-based adaptations, thereby improving security and reliability.

6 TESTING AND EVALUATION

The testing and evaluation of LLMs pose multifaceted challenges, as shown in Figure 7. Inspired by Chang et al. [26], we propose that these challenges can be systematically analyzed through three critical dimensions: **what**, **where**, and **how** to test and evaluate LLMs. These dimensions extend beyond the technical assessment of model performance to encompass broader considerations related to model deployment and real-world usage. The complex relationship between these factors highlights the need for systematic methodologies and innovative evaluation frameworks to ensure the reliability, robustness, and fairness of LLMs.

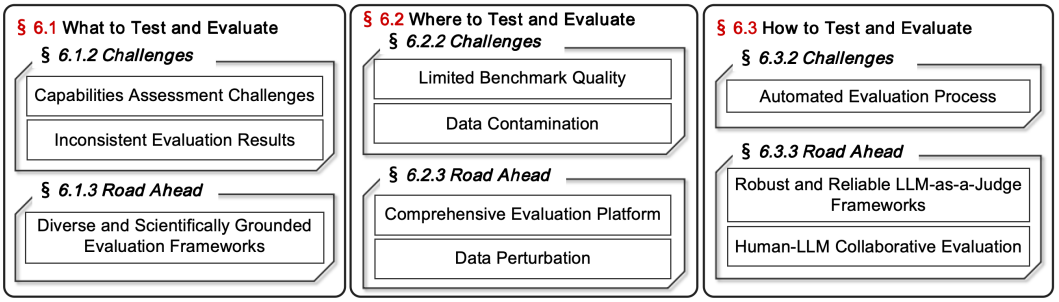

Fig. 7. Challenges and Road Ahead in §6 Testing and Evaluation.

6.1 What to Test and Evaluate

6.1.1 Research Status. The evaluation of LLMs spans multiple dimensions. From an application-driven perspective, researchers have explored LLM performance across diverse domains, including medicine [19, 50], education [404], SE [211], and finance [386]. In domains requiring advanced cognitive capabilities or creativity, studies have assessed LLMs' effectiveness in scientific research [332] and creative tasks [24]. Additionally, evaluations have focused on intrinsic model attributes, such as bias [301], reasoning capabilities [369], and planning capabilities [251]. Despite these efforts, there are still two significant challenges:

Difficulty in assessing certain capabilities. Many LLM capabilities are inherently difficult to quantify. Generative tasks, such as dialogue generation and writing, exhibit a high degree of subjectivity, making them challenging to evaluate using traditional objective metrics (e.g., accuracy). Scientifically quantifying abstract factors such as "creativity," "relevance," or "user satisfaction" remains an open research problem [158]. Additionally, certain attributes, such as reasoning, are difficult to observe directly. It is often unclear whether an LLM derives its responses through actual reasoning processes or merely retrieves relevant information from its knowledge base [12, 347].

Inconsistency in evaluation results. Variations in evaluation methodologies and metrics across different domains and tasks frequently result in inconsistencies in model performance assessments. For example, Gandhi et al. [92] identified discrepancies in LLM reasoning test results, highlighting underlying limitations in existing evaluation frameworks. Moreover, Greenblatt et al. [101] documented an issue where models adhere to training objectives during fine-tuning but fail to maintain this alignment in different scenarios, a phenomenon referred to as *alignment faking*. These inconsistencies raise concerns regarding the validity and robustness of current evaluation methodologies [92], underscoring the need for more comprehensive and rigorous evaluation.

6.1.2 Challenges. **Capabilities assessment challenges**. While certain LLM capabilities, such as code completion and mathematical computation, can be evaluated using manually designed test cases, more abstract skills such as reasoning, writing, and planning pose significant challenges for traditional evaluation methodologies. This difficulty arises from the inherent complexity of designing effective test cases, as well as the absence of well-defined quantitative metrics for objectively measuring these higher-order cognitive capabilities.

Inconsistent evaluation results. Another challenge is that assessments of whether an LLM possesses a particular capability or adheres to alignment expectations often yield inconsistent results. One possible explanation is the absence of scientifically rigorous evaluation methodologies, which can lead to discrepancies in judgment across different evaluation frameworks. Another factor is the phenomenon of *alignment faking* [101], where models appear to comply with expected behaviors during evaluation but deviate from them in different scenarios, raising concerns about the reliability of existing evaluation techniques.

6.1.3 Road Ahead. The challenges mentioned above highlight the limitations of current evaluation methodologies. Future research could focus on developing **diverse and scientifically grounded evaluation frameworks** that incorporate cross-domain methodologies to ensure comprehensive, reliable, and objective assessments of LLM performance across varied capabilities and tasks.

Cross-domain methodologies for capability assessment. A promising direction for improving the evaluation of LLM capabilities is the integration of cross-domain methodologies, which can enhance the scientific rigor of assessment techniques. This cross-domain approach holds significant potential. For example, in terms of KD, insights from educational science could be leveraged to assess a TM's effectiveness in knowledge transfer or an SM's capability to acquire and generalize learned information. Similarly, logic-based analysis could provide a more rigorous

framework for evaluating reasoning capabilities. By adopting scientifically rigorous evaluation methods, researchers can not only improve the assessment of abstract capabilities but also mitigate inconsistencies in evaluations.

Enhancing alignment evaluation through different testing environments. The emergence of *alignment faking* [101] underscores the influence of evaluation environment inconsistencies, wherein models demonstrate different behaviors under varying conditions. This phenomenon suggests that future evaluation methodologies should account for the impact of varying environments on evaluation outcomes, ensuring consistency and reliability across real-world usage scenarios. Developing adaptive evaluation frameworks that dynamically adjust test scenarios based on user inputs and interaction patterns could improve the robustness of alignment assessments. Additionally, integrating longitudinal evaluation strategies, where models are assessed over extended periods rather than in isolated test cases, could provide deeper insights into the stability of alignment and behavioral consistency. By grounding evaluations in different interaction environments, researchers can ensure more reliable assessments of LLM alignment and generalization capabilities.

6.2 Where to Test and Evaluate

6.2.1 Research Status. Numerous benchmarks have been developed to evaluate the capabilities of LLMs, each exhibiting distinct characteristics. Many of these benchmarks focus on assessing only a subset of LLM capabilities. For instance, code-related benchmarks such as xCodeEval [160], CoderUJB [436], and CrossCodeEval [62] primarily evaluate code understanding, generation, translation, and retrieval. Similarly, reasoning-oriented benchmarks like PlanBench [347] and LegalBench [106] target reasoning capabilities, while multimodal benchmarks such as SEED-Bench [175] and MM-SafetyBench [219] assess the performance of multimodal LLMs. Even within these specialized domains, benchmarks often focus on narrower subtasks. For example, within code-related evaluations, benchmarks like HumanEval [36], ClassEval [66], and EvoCodeBench [182] assess Python code generation, whereas JavaBench [20] and SWE-Bench-Java [435] evaluate Java code generation and repair, respectively.

Despite their utility, existing benchmarks suffer from several limitations. First, limited test case coverage restricts their comprehensiveness, as many benchmarks contain only a small number of test cases, reducing their capability to provide a holistic assessment of LLM capabilities. Additionally, the lack of standardized benchmark construction guidelines results in inconsistent dataset quality, leading to fragmented evaluation methodologies. As noted by McIntosh et al. [243], many existing benchmarks fail to capture nuanced aspects such as bias, genuine reasoning, and adherence to cultural and ideological norms.

Another significant challenge is data contamination, where test cases from benchmarks may have been seen by LLMs during training, leading to unrealistically high evaluation scores and overestimated model capabilities. This issue arises due to overlaps between real-world data used for benchmark construction and LLM training datasets. Even manually curated benchmarks such as HumanEval have been found to contain instances that newer models have encountered during training [298, 410].To mitigate this issue, some studies have proposed ensuring that benchmark data is sampled after the release of LLMs [81, 160]. However, these efforts have proven insufficient, as contamination can still occur when newer models are trained on datasets that include older benchmark samples [21, 410]. Data augmentation has gained attention as a potential solution, as it can generate novel data instances that were not present in the original dataset. Zhu et al. [466] introduced a psychometric-inspired data augmentation method to evaluate LLMs from multiple perspectives by modifying existing datasets. Additionally, researchers have leveraged LLM-based methods [384] to generate augmented datasets, taking advantage of LLMs' advanced language understanding and generation capabilities.

Ultimately, the lack of standardized benchmark usage guidelines leads to significant variability in evaluation quality and scope, resulting in divergent and sometimes inconsistent assessment outcomes. To address this, it is essential to develop a standardized evaluation framework that objectively assesses the reliability, scope, and validity of benchmark-based evaluations. In general, these limitations highlight the need for comprehensive research to develop more robust benchmarks, establish effective anti-contamination mechanisms, and create standardized assessment methodologies to enhance the reliability and fairness of LLM evaluations.

6.2.2 Challenges. **Limited benchmark quality.** While the number of benchmarks for evaluating LLMs has grown rapidly, many of them still face fundamental quality limitations. First, their scope is often restricted, even within a specific capability domain; existing benchmarks frequently fail to provide comprehensive coverage. Second, there is a notable difficulty distribution imbalance in test cases—some benchmarks are excessively challenging, while others are too simplistic, making it difficult to accurately assess an LLM's actual capabilities. Third, benchmarks rapidly become outdated, as LLMs continue to advance, many existing benchmarks lose their effectiveness over time, necessitating continuous updates and refinements to remain relevant.

Data contamination. As discussed before, data contamination can significantly distort evaluation results by introducing test cases that LLMs may have seen during training. Although mitigation strategies, such as post-release sampling and sample rephrasing, have been proposed, these approaches remain imperfect, highlighting the need for more robust methodologies that can effectively minimize data contamination while ensuring the validity and reliability of assessments.

6.2.3 Road Ahead. As LLMs continue to evolve, the limitations of existing evaluation benchmarks will persist, posing ongoing challenges. However, the development of a **comprehensive evaluation platform** could significantly mitigate these issues. Such a platform would provide a dedicated infrastructure for benchmark maintenance, facilitating continuous updates and the integration of new evaluation datasets. By enhancing benchmark reliability and diversity, this approach could help address the inherent shortcomings of current evaluation methodologies. While platforms such as Hugging Face [76] offer shared evaluation datasets, they lack effective dataset management, systematic benchmark updates, and continuous integration capabilities. Consequently, evaluation dataset quality remains inconsistent, limiting their ability to ensure benchmark reliability. To overcome these challenges, future research should focus on designing an adaptive benchmark management system that enables automated dataset curation, real-time benchmark refinement, and the dynamic integration of newly proposed evaluation metrics.

Additionally, since many LLM training datasets are not publicly available, the risk of data contamination remains a critical concern. To address this, **data perturbation** techniques could provide a potential solution. By transforming existing benchmark samples into novel representations while preserving their original semantic meaning, these techniques could generate test cases that are distinct from those encountered during model training. This approach would help reduce the likelihood of evaluation biases, regardless of whether the original benchmark data is sourced from real-world datasets or synthetically generated.

6.3 How to Test and Evaluate

6.3.1 Research Status. In the evaluation of LLMs, certain capabilities, such as fill-in-the-middle (FIM) performance in the coding domain [61, 97, 218, 381], can be assessed through the automated execution of test cases, yielding objective pass rate metrics. However, more abstract capabilities, such as creativity and reasoning, are significantly more challenging to evaluate automatically and often require human judgment [24, 320, 353]. This reliance on manual evaluation introduces two primary issues. First, human judgment is subjective, leading to inconsistencies and reduced reliability in

evaluation outcomes. Second, manual assessment is both time-consuming and labor-intensive, making large-scale evaluations impractical and limiting the comprehensiveness of assessments. These challenges highlight the need for developing more scalable and objective evaluation methodologies.

6.3.2 Challenges. **Automated evaluation process.** While certain LLM capabilities can be assessed using automated tools, more abstract capabilities such as creativity and reasoning remain challenging to evaluate fully automatically. As a result, manual assessment is often required, introducing subjectivity that reduces the accuracy and reliability of evaluation outcomes. Moreover, the reliance on human judgment limits the feasibility of conducting large-scale assessments, posing a significant barrier to comprehensive and scalable LLM evaluation.

6.3.3 Road Ahead. Given the powerful capabilities of LLMs, LLM-based evaluation methods, commonly referred to as LLM-as-a-Judge, have gained increasing attention [41, 116, 180, 202, 358]. However, these methods are inherently influenced by the biases present in LLMs [169, 372, 424], as well as their intrinsic limitations [348, 372], which may lead to inaccurate evaluation outcomes. Therefore, a key research direction is the development of more **robust and reliable LLM-as-a-Judge frameworks**, which includes strategies to mitigate biases in evaluation and the integration of multi-model and multi-modal approaches to enhance fairness and reliability. By leveraging diverse models and modalities, these methods could reduce individual model biases and improve overall assessment accuracy. Another promising direction is **human-LLM collaborative evaluation**, which serves as a compromise between full automation and evaluation accuracy. The goal of this approach is to balance the efficiency of automated evaluation with the careful judgment of human evaluators, thereby producing LLM assessments that are both reliable and interpretable.

7 DEPLOYMENT AND OPERATIONS

As discussed in §5.3, the deployment of LLMs presents several challenges, including computational resource constraints, deployment architecture design, and security and privacy concerns. To systematically explore these challenges, we classify LLM deployment into three categories based on the location of computational resources: **cluster deployment**, **edge deployment**, and **hybrid deployment**, which is shown in Figure 8.

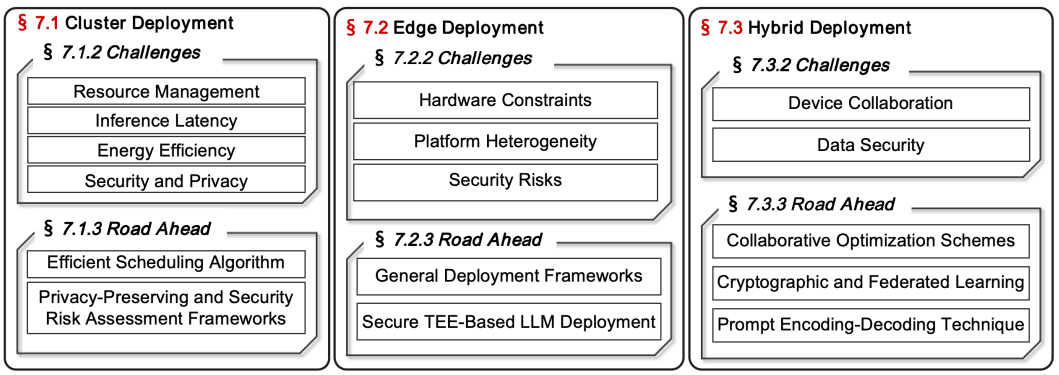

Fig. 8. Challenges and Road Ahead in §7 Deployment and Operations.

7.1 Cluster Deployment

7.1.1 Research Status. Cluster deployment refers to deploying models in high-performance computing clusters, such as the cloud, to leverage distributed computing for large-scale inference. While

this approach enables efficient processing of large-scale requests, it also introduces significant technical and operational challenges.

Resource management. Managing computational resources in a distributed environment is inherently complex, particularly due to the need for efficient parallelization across heterogeneous hardware (e.g., GPUs, TPUs). To keep systems responsive while evenly distributing tasks, there's a need for smart scheduling methods and flexible resource scaling [451]. Hisaharo et al. [123] redesigned the cluster computing architecture of GPT-Neo and optimized software implementations to enhance inference efficiency. Similarly, Zhao et al. [451] proposed an adaptive algorithm for optimizing LLM inference in heterogeneous environments by dynamically adjusting mixed-precision quantization and GPU allocation strategies to improve throughput. Additionally, LLM-Pilot [171] introduced a predictive model that recommends cost-effective hardware configurations, further improving resource utilization.

Inference latency. Despite the high throughput of cluster-based deployment, ensuring consistently low-latency responses under high-concurrency conditions (e.g., thousands of simultaneous requests) remains a significant challenge. Optimized batching strategies, model partitioning techniques (e.g., pipeline heterogeneity [120]), and hardware-aware kernel fusion are commonly employed to mitigate computational bottlenecks. Local checkpoint storage [89] and splitwise techniques [279] have been proposed to enhance inference efficiency. However, while checkpoint storage reduces redundant computation, it increases storage overhead, whereas inter-stage computation transfers in splitwise techniques offer only limited latency reductions. Sarathi-Serve [7] introduced chunked-prefill and stall-free scheduling techniques, significantly reducing inference latency under high-throughput conditions, with greater optimizations observed for larger models, suggesting their scalability benefits. Zhang et al. [443] explored collaborative edge computing to partition clusters and employed dynamic programming to minimize latency and maximize throughput. However, cross-partition data transfers introduce potential privacy risks, which will be discussed later.

Energy efficiency. The substantial energy consumption associated with training and inference on GPU/TPU clusters necessitates effective hardware utilization and memory optimization. To address this issue, Wilkins et al. [376] proposed a technique that dynamically allocates computational resources based on token input-output ratios, thereby reducing energy consumption. Hisaharo et al. [123] and Stojkovic et al. [328] introduced optimization algorithms for dynamically managing inference resources to lower energy consumption. However, the search space of such algorithms remains large, prompting Maliakel et al. [240] to explore key parameters affecting energy efficiency across different LLMs and tasks. Their findings highlight the need for task-specific hardware optimizations to enhance efficiency further. Additionally, Hewage et al. [122] identified CPU aging as a contributing factor to increased energy consumption and proposed optimization strategies aimed at extending CPU longevity to reduce energy usage.

Security risks. Cluster-based LLM deployment is susceptible to security threats, including adversarial attacks on exposed APIs and data leakage in multi-tenant environments [53]. Yang et al. [408] found that LLM service providers often optimize inference efficiency by sharing KV caches, inadvertently exposing user privacy. Furthermore, Soleimani et al. [322] demonstrated vulnerabilities in speculative decoding optimizations, introducing a novel side-channel attack capable of extracting LLM token information from encrypted transmissions, such as token size. To mitigate these risks, encryption techniques such as multi-party computation (MPC) [295], homomorphic encryption (HE) [474], and TEE [249] have been applied to LLM inference. However, while these methods enhance privacy protection, they introduce significant computational overhead, resulting in increased inference latency, which limits their practicality in real-world applications.

Additionally, handling non-linear computations such as the `softmax` function remains a significant challenge in encrypted LLM inference [474].

In addition, weak authentication mechanisms and the imperfect of LLM deployment frameworks introduce further security concerns. Pesati et al.[284] found that unreliable authentication mechanisms can lead to LLMs being hijacked or manipulated, posing serious security risks. Hou et al.[125] analyzed several popular LLM deployment frameworks such as ComfyUI [44] and Ollama [265], and identified widespread model information disclosure vulnerabilities. These issues can also lead to unauthorized access, exploitation of system vulnerabilities, and other security threats.

7.1.2 Challenges. As discussed in §7.1.1, the challenges of LLM cluster deployment can be categorized into the following four aspects.

Resource management. Efficient resource management is crucial in heterogeneous and distributed environments, requiring sophisticated scheduling algorithms and dynamic scaling mechanisms to balance concurrency and inference latency. The challenge of handling GPUs, TPUs, and other specialized hardware makes performance tuning more difficult, requiring flexible resource allocation methods that adapt to different system demands.

Inference latency. Minimizing inference latency is essential not only for improving user experience but also for reducing operational costs. However, optimizing LLM inference presents a systemic engineering challenge that involves architectural design, workload distribution, and hardware utilization. Techniques such as model partitioning, batching strategies, and pipeline parallelism have led to noticeable improvements in performance. However, they still face limitations in scalability and hardware efficiency, leaving many areas open for further refinement.

Energy efficiency. LLM inference demands substantial computational resources, resulting in high energy consumption. Existing energy-saving strategies include optimizing resource allocation, extending the lifespan of hardware, and implementing efficient scheduling policies. However, given the vast optimization space, determining the optimal configuration remains a significant challenge.

Security and privacy. Cluster-based LLM deployment introduces security and privacy risks, including potential data leakage due to shared KV caches and API calls. Issues such as weak authentication, information disclosure, and unauthorized access further highlight the diversity and complexity of LLM security challenges. Addressing these issues cannot rely on patching individual vulnerabilities or adopting traditional security technologies such as encryption. Instead, it requires a comprehensive security strategy that combines robust access control, fine-grained API governance, secure configuration management, and continuous monitoring. Moreover, securing LLM deployments should be treated as a system-level problem, involving coordination across model architecture, serving infrastructure, and user interaction layers to ensure a defense-in-depth approach.

7.1.3 Road Ahead. We categorize the challenges mentioned above into two key areas: **LLM cluster inference optimization** and **LLM cluster inference privacy and security concerns**.

For LLM inference optimization, existing approaches to addressing high concurrency, low latency, and energy efficiency primarily focus on scheduling design and architectural optimization, making these areas critical for further research. However, the vast number of optimization factors results in a huge search space, complicating the identification of an optimal configuration. A crucial future direction is the development of an **efficient scheduling algorithm** capable of dynamically exploring this search space and autonomously determining resource allocation and real-time scheduling strategies. Such an algorithm should be designed to optimize multiple objectives simultaneously, ensuring high concurrency, low latency, and reduced energy consumption while adapting to varying workloads and hardware configurations.

Concerning privacy and security concerns, establishing a comprehensive **privacy-preserving and security risk assessment framework** will be essential in the future. Regarding privacy preservation, differential privacy techniques [343, 447] show promise by introducing controlled noise into prompts or token representations, obscuring sensitive information. Such techniques offer configurable privacy settings that balance data confidentiality with model utility, thus enhancing the practicality of secure LLM inference. Regarding security risk assessment, it is equally essential to incorporate mechanisms for identifying, assessing, and mitigating security threats. This includes developing unified identity and access management frameworks to enforce fine-grained, role-based permissions, ensuring minimal privilege assignment, and providing standardized authentication interfaces to facilitate integration with third-party services. Additionally, specialized security analysis tools, such as static configuration analyzers, automated endpoint scanning, and risk assessment utilities, should be introduced to systematically address issues related to configuration leakage and interface exposure. These integrated efforts collectively reinforce the robustness of LLM deployment frameworks.

7.2 Edge Deployment

7.2.1 Research Status. Edge deployment refers to deploying LLMs on edge devices near the data source, such as smartphones, IoT devices, and edge servers, referred to as on-device LLMs, rather than relying on centralized cloud infrastructure. This approach alleviates the computational burden on cloud-based LLM services while enhancing the quality of commercial LLM applications. However, it introduces challenges, primarily due to memory and computational resource constraints [58].

Hardware constraints. Edge devices typically have limited computing resources (e.g., mobile CPUs and GPUs), restricted memory capacity, and energy constraints, necessitating the use of model compression techniques. However, these techniques often come at the cost of reduced accuracy or robustness. To address this issue, Ma et al. [233] proposed a quantization technique where each parameter takes values from {-1,0,1}, achieving significant improvements in memory efficiency, inference latency, energy consumption, and throughput. Similarly, Li et al. [184] introduced four techniques, providing diverse strategies for improving LLM deployment on mobile devices.

Platform heterogeneity. The deployment of LLMs on edge devices is further complicated by platform heterogeneity, as models must be compatible with diverse architectures (e.g., ARM-based chips, NPUs). Several existing solutions facilitate cross-platform deployment. Llama.cpp [96] is a C++ library that supports LLM deployment across various hardware platforms, integrating integer quantization and GPU acceleration. MNN [149], a mobile neural network framework, enables deployment across different backends, with its extension MNN-LLM specifically designed for LLM deployment on mobile devices, PCs, and embedded systems. ExecuTorch [288] is an end-to-end edge inference framework tailored for deploying PyTorch models on edge devices.

Security risks. Edge deployment also introduces heightened security risks, as models are exposed in a white-box manner, making them more susceptible to physical tampering, adversarial inputs, and model stealing [188]. However, processing sensitive user information locally reduces the risk of data leakage compared to cloud-based deployment [188, 443], highlighting a trade-off between security threats and privacy benefits.

7.2.2 Challenges. Edge deployment presents three primary challenges. **Hardware Constraints.** Unlike centralized servers, edge devices have limited computational capabilities, necessitating model compression before deployment. However, this process involves a trade-off between model size and performance, as aggressive compression techniques can degrade accuracy and robustness. **Platform Heterogeneity.** The diversity of operating systems and hardware architectures across edge devices complicates deployment, requiring additional driver support and optimization for

compatibility. While certain open-source tools facilitate cross-platform deployment, most solutions are tailored to specific LLM ecosystems or hardware platforms, limiting their general applicability. **Security Risks.** Models deployed on edge devices are effectively exposed in a white-box manner, making them vulnerable to threats such as model extraction and adversarial attacks. Ensuring robust security mechanisms while maintaining efficient inference remains a challenge.

7.2.3 Road Ahead. Both **hardware constraints** and **platform heterogeneity** are deployment-related challenges that can be addressed through SE solutions. A key direction for future research is the development of **generalized deployment frameworks**, such as Llama.cpp and MNN, which facilitate seamless LLM deployment across diverse edge devices. These frameworks should not only support cross-platform compatibility but also integrate common model compression and fine-tuning techniques, allowing users to optimize models according to specific deployment requirements. Furthermore, to enhance their applicability, these tools should be extended beyond the LLaMA family to support a broader range of models, such as the DeepSeek family and the Gemma family, among others.

Regarding **security risks** in edge LLM deployment, existing research remains limited. Given the heterogeneous and dynamic nature of edge computing environments, potential attack vectors are diverse and complex. One promising approach is executing models within a TEE, which provides hardware-based isolation for secure model inference. However, this method introduces additional challenges, as it requires specialized hardware support and remains vulnerable to various security threats [23], including side-channel attacks such as CipherFix attacks [375] and cache side-channel attacks [473]. Therefore, future research on **secure TEE-based LLM deployment** should focus on two key aspects: *(a)* ensuring compatibility across diverse hardware architectures to promote widespread adoption, and *(b)* developing robust defense mechanisms against known vulnerabilities, such as side-channel attacks [375, 473], to enhance the security and reliability of on-device LLMs.

7.3 Hybrid Deployment

7.3.1 Research Status. Hybrid deployment, such as edge-cloud collaborative computing, integrates the advantages of both edge and cloud deployment, offering additional benefits. On one hand, it enhances the flexibility of model deployment and task execution—computationally intensive tasks can be offloaded to cloud clusters, whereas lightweight tasks can be processed on edge devices. On the other hand, it broadens the application scenarios of LLMs, enabling cross-regional task execution and real-time decision-making in latency-sensitive applications.

Device collaboration. Effective collaboration between cloud and edge devices requires dynamic task partitioning and computation offloading while balancing inference latency, bandwidth constraints, and data privacy [121, 407, 443]. He et al. [121] proposed an active inference approach using reinforcement learning for resource scheduling in cloud-edge LLM inference. CE-CoLLM [151], a cloud-edge collaborative inference framework, employs early-exit mechanisms, a cloud context manager, and quantization to reduce high communication overhead, achieving both low-latency edge standalone inference and high-accuracy cloud-edge collaborative inference. Additionally, Hao et al. [115] introduced a hybrid inference method that leverages small models on edge devices in conjunction with large cloud-based models to enhance inference performance.

Data security. Hybrid deployment involves extensive data exchange, increasing the risk of data leakage. Common privacy-preserving techniques include federated learning [32, 167], differential privacy [27, 262, 313], and split learning [291]. Federated learning protects user data by enabling local model training while updating a global model in the cloud. Differential privacy introduces noise into data to obscure sensitive information, whereas split learning transmits only partial computation results to the cloud, thereby minimizing data exposure. However, there still are privacy risks, as

adversarial servers may attempt to reconstruct users' original data [291]. Furthermore, as previously discussed, attacks such as leveraging soft prompt to expose LLM training data could also manifest in hybrid deployment settings. Ensuring data security in hybrid deployments remains an open research challenge that requires further investigation.

7.3.2 Challenges. Hybrid deployment introduces additional **challenges** beyond those encountered in cluster and edge deployment. **Device collaboration.** Unlike cluster deployment, hybrid deployment requires efficient coordination between edge devices and cloud servers. The heterogeneity of edge devices, each with varying computational capabilities, further complicates scheduling, making task offloading and resource allocation more challenging. Additionally, optimizing these processes must account for inference latency, bandwidth constraints, and dynamic workload distribution, increasing the complexity of system coordination. **Data security.** Compared to edge deployment, hybrid deployment involves frequent data exchanges with cloud servers, heightening the risk of data exposure. Unlike in cluster deployment, this risk is made worse by the diverse and less controlled nature of edge environments. Moreover, since both cloud servers and edge devices participate in computations, either party could act maliciously, making security threats more unpredictable and the deployment environment more complex.

7.3.3 Road Ahead. To address the challenges associated with **device collaboration**, future research can explore **collaborative optimization schemes** from three key perspectives: low-latency communication, performance optimization, and intelligent scheduling. For **low-latency communication**, advancements in networking and data transmission technologies are crucial for accelerating information exchange within collaborative networks and minimizing communication delays. From a SE perspective, optimizing the inference process itself can significantly enhance communication efficiency between cloud and edge devices. Future solutions could integrate techniques such as vector database caching [421] and MoE architectures [152] to reduce latency and improve inference performance. Regarding **performance optimization**, traditional LLM enhancement techniques (e.g., RAG, MoE, and prompt engineering [152, 289]) can be leveraged to improve hybrid inference efficiency. Additionally, novel strategies, such as constraint satisfaction mechanisms for complex decision-making in edge-cloud collaboration [417] and the integration of quantization with efficient local inference methods [296], may further enhance the computational capabilities of edge devices. For **intelligent scheduling**, future research can focus on developing more adaptive task and resource allocation strategies [417, 443]. These strategies should ensure system robustness by dynamically adjusting to inference failures, resource constraints, and evolving workloads in real-time.

To enhance **data security**, **cryptographic methods** such as HE, zero-knowledge proofs (ZKP) [331], MPC [475], and blockchain-based security mechanisms [193, 363] present potential solutions. However, these cryptographic methods significantly increase the computational overhead of LLM inference, which limits their applicability on edge devices with limited resources. Alternative approaches such as **federated learning** [167, 425] and confidential computing [475] have gained attention due to their compatibility with hybrid deployment environments, with differential privacy playing a critical role in federated learning. Inspired by this, a promising direction involves injecting noise into prompts while ensuring that LLMs can still correctly interpret the intended information. Since prompts primarily consist of natural language, a **prompt encoding-decoding technique** could be explored to transform prompts into structured representations for noise injection, followed by decoding them back into natural language when needed. A related approach is prompt obfuscation [199, 277], which can also protect sensitive information within prompts from being extracted or exploited by adversarial entities. These techniques enhance prompt security while preserving the effectiveness of LLM interactions.

8 MAINTENANCE AND EVOLUTION

8.1 Research Status

LLM maintenance and evolution encompass the ongoing operation, monitoring, and upgrading of models post-deployment, ensuring stable inference, addressing emerging issues, and continuously improving model performance. Unlike the development and enhancement phases, which primarily focus on model training and fine-tuning, maintenance and evolution require long-term management strategies to sustain the efficiency, reliability, and compliance of LLMs in real-world applications. We list the challenges and potential future directions in Figure 9.

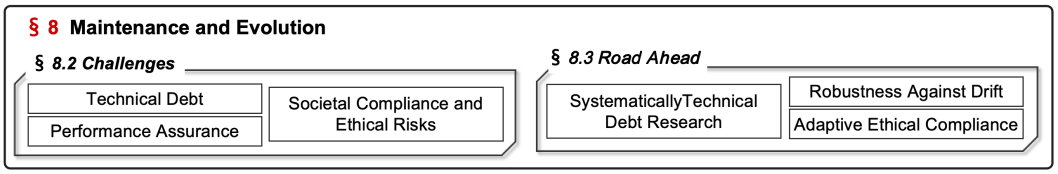

Fig. 9. Challenges and Road Ahead in §8 Maintenance and Evolution.

The rapid advancement of LLMs often leads to the accumulation of **technical debt**, as ad-hoc solutions (e.g., memory management, model compression, and attention optimization) are implemented to address short-term challenges [245]. However, these solutions may hinder long-term sustainability by increasing system complexity and maintenance overhead. In addition to these architectural challenges, iterative model updates introduce versioning complexities, including API compatibility issues and dependency conflicts, further increasing maintenance efforts. As Ma et al. [234] highlighted, APIs may be no longer used during model development, making regression testing to be a crucial concern. While tools such as MLflow Model Registry [1] and PEFT [75] mitigate some of these challenges, critical gaps remain in quantifying technical debt and designing unified lifecycle frameworks that balance incremental learning with catastrophic forgetting.

Ensuring consistent model performance in dynamic environments requires addressing **model drift**, such as task [2], data [142], semantic [69, 299, 304], concept [412], and knowledge [79] drift, which are common in multi-model collaboration. Existing techniques, such as drift detection (e.g., the Kolmogorov-Smirnov test [104]) and model compression, offer partial solutions which may need full model retraining, remaining computationally expensive, and semantic-level drift is often detected with significant delays. Furthermore, The increasing complexity of multi-model collaboration makes these drifts more severe. Maintaining model performance in such settings requires not only early drift detection but also the development of robust adaptation mechanisms to mitigate its effects. However, current solutions remain insufficient, either relying on costly retraining or failing to address higher-level semantic and knowledge drift, which can cause subtle yet significant deviations in model behavior over time.

Beyond technical considerations, LLMs must also comply with evolving legal frameworks (e.g., the EU AI Act [70]) and ethical standards, necessitating the development of automated compliance mechanisms that integrate regulatory constraints into model behavior. However, automating compliance remains a significant challenge. While post-hoc filters and bias assessment tools help mitigate immediate risks [350, 397], they are insufficient for preventing long-term societal harms, such as the reinforcement of biases from incremental data updates or cross-cultural misalignment [18, 48, 345].

Thus, maintaining and evolving LLMs need comprehensive solutions that integrate **technical, operational, and regulatory** considerations. Future research must explore systematic strategies for managing technical debt, improving drift adaptation mechanisms, and developing more proactive

compliance frameworks to ensure that LLMs remain reliable, efficient, and aligned with ethical and legal standards.

8.2 Challenges

The maintenance and evolution of LLMs present multifaceted challenges that extend beyond traditional SE paradigms. These challenges can be categorized into three interconnected dimensions, encompassing both technical and societal complexities: **technical debt**, **performance assurance**, and **societal compliance and ethical risks**.

Technical debt. The rapid advancements in model compression, fine-tuning, and continual learning have led to the accumulation of hidden technical debt. For instance, the increasing complexity of model architectures reduces post-training interpretability, while continual learning may exacerbate rather than mitigate biases [245]. These technical debts pose significant risks to model improvement, inference reliability, and security. However, due to the lack of systematic studies in this area, a comprehensive understanding of LLM technical debt remains elusive, making it hard to develop effective mitigation strategies.

Performance assurance. Model drift has emerged as a critical challenge, often resulting in unexpected inference errors, degraded performance, or even the generation of harmful content. Given the dynamic nature of deployment environments and evolving user interactions, mitigating model drift requires robust adaptation mechanisms. Yet, existing methods remain limited in their ability to detect and counteract drift efficiently, particularly at the semantic level, where subtle but significant changes in model behavior can occur over time.

Societal compliance and ethical risks. As LLMs are increasingly deployed in real-world applications, ensuring their outputs align with ethical and social values is imperative. However, current alignment efforts remain insufficient, as adversarial attacks can manipulate even aligned models into producing undesirable outputs. Furthermore, the emergence of *alignment faking* [101] raises concerns that models may exhibit alignment during evaluation but deviate in other scenarios, casting doubt on the reliability of existing alignment techniques. Addressing ethical alignment in LLMs thus remains an ongoing and pressing research challenge.

8.3 Road Ahead

As of the date of writing, research on model maintenance remains limited, particularly in the context of technical debt. However, it addresses a critical issue that deserves greater attention. For the technical debt of LLM, we suggest that the first step in future research should be to do work like **systematically technical debt research** to assess its impact, and subsequently develop effective mitigation strategies. Additionally, LLMOps [59] has emerged as a promising paradigm for enabling automated lifecycle management and standardized control in LLM development, effectively mitigating common technical debt issues. For instance, LLMOps facilitates real-time model monitoring and continuous feedback mechanisms, allowing for the timely detection and correction of model degradation or knowledge loss, thereby preventing the long-term accumulation of quality debt. By leveraging automation, standardization, and optimization techniques, LLMOps holds significant potential for addressing various technical debt challenges in industrial LLM applications, making it a compelling research direction.

Similarly, **robustness against drift** necessitates systematic solutions to mitigate the effects of drift in LLMs. Future research may explore the alignment of multi-modal and multi-model representation spaces, prompt refinement, and techniques for preserving semantic integrity in long and sequentially evolving contexts. Additionally, the development of an automated model evolution framework or a performance monitoring system could provide continuous assessment of LLM performance across tasks, both during regular operation and after knowledge updates. By

detecting model drift, these systems would allow early action to fix issues, ensuring reliability and stability throughout the LLM development lifecycle, and preventing unexpected performance drops while maintaining accuracy across different deployment scenarios.

Bias and ethical concerns in LLMs must not be overlooked, making **adaptive ethical compliance** a crucial area of study. While numerous techniques have been proposed to address societal compliance and ethical risks during LLM training, future research should focus on dynamically and precisely adjusting ethical and bias-related concepts during model maintenance, which may involve integrating knowledge unlearning and continual learning techniques while simultaneously addressing the challenge of *alignment faking* [101]. Furthermore, as regulatory and ethical constraints on LLMs continue to evolve, translating legal frameworks and ethical norms into enforceable model constraints will be essential for ensuring sustained compliance with regulatory changes and societal expectations. Importantly, these adaptations must be achieved without introducing excessive system complexity, thereby preserving the efficiency and scalability of LLM deployment and operation.

9 RELATED WORK

With the rapid advancement of LLMs and their success across various applications, research on LLMs has experienced explosive growth in recent years. To systematically summarize existing achievements and outline future directions, a substantial number of survey studies have emerged. Overall, the existing surveys can be categorized into two groups: those focusing on the fundamental aspects of LLMs and those emphasizing the applications of LLMs in different domains.

On one hand, as LLMs include multiple dimensions such as model architecture, training methodologies, and security evaluation, many existing surveys focus on specific aspects of LLM research. Zhao et al.[454] and Naveed et al.[256] primarily concentrate on the development trajectory of LLMs, providing detailed reviews of key technological advancements and major research milestones. Chang et al.[26], Xu et al.[392], and Guo et al.[110] focus specifically on evaluation techniques for LLMs. In addition, as concerns over security risks grow with the increasing scale of models, Yao et al.[420], Wang et al.[361], and Das et al.[51] provide systematic analyses of LLM security issues, covering impact assessment, domain-specific vulnerabilities, and overarching security challenges, respectively.

On the other hand, survey studies focusing on the application of LLMs have also been increasing. In particular, the use of LLMs for SE has emerged as a highly active area of research in recent years. Hou et al.[126] conducted one of the systematic studies in this field, followed by further investigations by Fan et al.[78] and Liu *et al.* [210]. Beyond SE, several surveys have summarized the applications of LLMs across various industries, such as telecommunications [461], medicine [340], and education [157]. Additionally, research has explored LLMs in roles as human judges (LLM-as-a-judge)[103, 177, 177], as well as in SE subfields such as code generation[147, 357] and code repair [446], alongside comprehensive analyses of LLM applications [112, 154]. These studies highlight the critical role of LLMs in today's society and underscore their potential for future development, emphasizing the continuing importance of advancing LLM technologies.

However, existing studies have not provided a systematic analysis of the LLM development lifecycle from an SE perspective. To the best of our knowledge, **we presents the first comprehensive survey in this work** that examines the LLM development lifecycle through the lens of SE. Our study systematically reviews the key SE challenges associated with LLM development and proposes critical directions for future research, offering valuable insights to guide subsequent work in this emerging area.

10 CONCLUSION

This paper presents a comprehensive analysis of the challenges associated with LLMs from an SE perspective. By systematically examining each phase of the LLM development lifecycle, we provide an in-depth review of the current research landscape, identify key challenges, and present future research directions. Our findings provide valuable insights to facilitate further advancements in this field, contributing to the development of more efficient, robust, and scalable LLMs.

REFERENCES

[1] [n. d.]. MLflow Model Registry. https://mlflow.org/docs/latest/model-registry/. Accessed: 2025-04-09.

[2] Sahar Abdelnabi, Aideen Fay, Giovanni Cherubin, Ahmed Salem, Mario Fritz, and Andrew Paverd. 2024. Are you still on track!? Catching LLM Task Drift with Activations. *arXiv preprint arXiv:2406.00799* (2024).

[3] Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*.

[4] Agent Communication Protocol Project. 2025. Introduction to the Agent Communication Protocol. https://agentcommunicationprotocol.dev/introduction/welcome Accessed: 2025-05-12.

[5] Agent Network Protocol Project. 2025. Agent Network Protocol (ANP) Official Website. https://agent-network-protocol.com/ Accessed: 2025-05-12.

[6] Ahmed Agiza, Marina Neseem, and Sherief Reda. 2024. Mtlora: Low-rank adaptation approach for efficient multi-task learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 16196–16205.

[7] Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav Gulavani, Alexey Tumanov, and Ramachandran Ramjee. 2024. Taming {Throughput-Latency} tradeoff in {LLM} inference with {Sarathi-Serve}. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*. 117–134.

[8] Khlood Ahmad, Mohamed Abdelrazek, Chetan Arora, Muneera Bano, and John Grundy. 2023. Requirements engineering for artificial intelligence systems: A systematic mapping study. *Information and Software Technology* 158 (2023), 107176.

[9] Meta AI. 2025. Llama: Open-Source AI Models. https://www.llama.com/.

[10] Daniel Alexander Alber, Zihao Yang, Anton Alyakin, Eunice Yang, Sumedha Rai, Aly A Valliani, Jeff Zhang, Gabriel R Rosenbaum, Ashley K Amend-Thomas, David B Kurland, et al. 2025. Medical large language models are vulnerable to data-poisoning attacks. *Nature Medicine* (2025), 1–9.

[11] Shaden Alshammari, Yu-Xiong Wang, Deva Ramanan, and Shu Kong. 2022. Long-tailed recognition via weight balancing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 6897–6907.

[12] Maryam Amirizaniani, Elias Martin, Maryna Sivachenko, Afra Mashhadi, and Chirag Shah. 2024. Can llms reason like humans? assessing theory of mind reasoning in llms for open-ended questions. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 34–44.

[13] Anthropic. 2025. Claude 3.7 Sonnet and Claude Code. https://www.anthropic.com/news/claude-3-7-sonnet Accessed: 2025-04-20.

[14] Anthropic. 2025. MCP: Agents and Tools Overview. https://docs.anthropic.com/en/docs/agents-and-tools/mcp Accessed: 2025-04-26.

[15] Daiyaan Arfeen, Zhen Zhang, Xinwei Fu, Gregory R Ganger, and Yida Wang. 2024. PipeFill: Using GPUs During Bubbles in Pipeline-parallel LLM Training. *arXiv preprint arXiv:2410.07192* (2024).

[16] Guangji Bai, Zheng Chai, Chen Ling, Shiyu Wang, Jiaying Lu, Nan Zhang, Tingwei Shi, Ziyang Yu, Mengdan Zhu, Yifei Zhang, et al. 2024. Beyond efficiency: A systematic survey of resource-efficient large language models. *arXiv preprint arXiv:2401.00625* (2024).

[17] Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, and Jaehoon Amir Safavi. 2017. Mitigating poisoning attacks on machine learning models: A data provenance based approach. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*. 103–110.

[18] Peter J Barclay and Ashkan Sami. 2024. Investigating Markers and Drivers of Gender Bias in Machine Translations. *arXiv preprint arXiv:2403.11896* (2024).

[19] Yan Cai, Linlin Wang, Ye Wang, Gerard de Melo, Ya Zhang, Yanfeng Wang, and Liang He. 2024. Medbench: A large-scale chinese benchmark for evaluating medical large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 17709–17717.

[20] Jialun Cao, Zhiyong Chen, Jiarong Wu, Shing-Chi Cheung, and Chang Xu. 2024. JavaBench: A Benchmark of Object-Oriented Code Generation for Evaluating Large Language Models. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*. 870–882.

[21] Jialun Cao, Wuqi Zhang, and Shing-Chi Cheung. 2024. Concerned with Data Contamination? Assessing Countermeasures in Code Language Model. *arXiv preprint arXiv:2403.16898* (2024).
[22] Federico Cassano, John Gouwar, Daniel Nguyen, Sydney Nguyen, Luna Phipps-Costin, Donald Pinckney, Ming-Ho Yee, Yangtian Zi, Carolyn Jane Anderson, Molly Q Feldman, et al. 2022. Multipl-e: A scalable and extensible approach to benchmarking neural code generation. *arXiv preprint arXiv:2208.08227* (2022).
[23] David Cerdeira, Nuno Santos, Pedro Fonseca, and Sandro Pinto. 2020. Sok: Understanding the prevailing security vulnerabilities in trustzone-assisted tee systems. In *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1416–1432.
[24] Tuhin Chakrabarty, Philippe Laban, Divyansh Agarwal, Smaranda Muresan, and Chien-Sheng Wu. 2024. Art or artifice? large language models and the false promise of creativity. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–34.
[25] Aofei Chang, Jiaqi Wang, Han Liu, Parminder Bhatia, Cao Xiao, Ting Wang, and Fenglong Ma. 2024. BIPEFT: Budget-Guided Iterative Search for Parameter Efficient Fine-Tuning of Large Pretrained Language Models. *arXiv preprint arXiv:2410.09079* (2024).
[26] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology* 15, 3 (2024), 1–45.
[27] Zachary Charles, Arun Ganesh, Ryan McKenna, H Brendan McMahan, Nicole Mitchell, Krishna Pillutla, and Keith Rush. 2024. Fine-tuning large language models with user-level differential privacy. *arXiv preprint arXiv:2407.07737* (2024).
[28] Harrison Chase and contributors. 2022. LangChain: Build context-aware reasoning applications. https://github.com/langchain-ai/langchain.
[29] Arnav Chavan, Raghav Magazine, Shubham Kushwaha, Mérouane Debbah, and Deepak Gupta. 2024. Faster and Lighter LLMs: A Survey on Current Challenges and Way Forward. *arXiv preprint arXiv:2402.01799* (2024).
[30] Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M De Sa. 2024. Quip: 2-bit quantization of large language models with guarantees. *Advances in Neural Information Processing Systems* 36 (2024).
[31] Viktoriia Chekalina, Anna Rudenko, Gleb Mezentsev, Alexander Mikhalev, Alexander Panchenko, and Ivan Oseledets. 2024. SparseGrad: A Selective Method for Efficient Fine-tuning of MLP Layers. *arXiv preprint arXiv:2410.07383* (2024).
[32] Chaochao Chen, Xiaohua Feng, Jun Zhou, Jianwei Yin, and Xiaolin Zheng. 2023. Federated large language model: A position paper. *arXiv e-prints* (2023), arXiv–2307.
[33] Dake Chen, Hanbin Wang, Yunhao Huo, Yuzhao Li, and Haoyang Zhang. 2023. Gamegpt: Multi-agent collaborative framework for game development. *arXiv preprint arXiv:2310.08067* (2023).
[34] Jieneng Chen, Luoxin Ye, Ju He, Zhaoyang Wang, Daniel Khashabi, and Alan L Yuille. 2025. Efficient large multi-modal models via visual context compression. *Advances in Neural Information Processing Systems* 37 (2025), 73986–74007.
[35] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. 2025. Sharegpt4v: Improving large multi-modal models with better captions. In *European Conference on Computer Vision*. Springer, 370–387.
[36] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374* (2021).
[37] Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-Play Fine-Tuning Converts Weak Language Models to Strong Language Models. arXiv:2401.01335 [cs.LG] https://arxiv.org/abs/2401.01335
[38] Zitao Chen and Karthik Pattabiraman. 2024. Catch Me if You Can: Detecting Unauthorized Data Use in Deep Learning Models. *arXiv preprint arXiv:2409.06280* (2024).
[39] Yuheng Cheng, Ceyao Zhang, Zhengwen Zhang, Xiangrui Meng, Sirui Hong, Wenhao Li, Zihao Wang, Zekai Wang, Feng Yin, Junhua Zhao, et al. 2024. Exploring large language model based intelligent agents: Definitions, methods, and prospects. *arXiv preprint arXiv:2401.03428* (2024).
[40] Wonje Choi, Woo Kyung Kim, Minjong Yoo, and Honguk Woo. 2024. Embodied CoT Distillation From LLM To Off-the-shelf Agents. arXiv:2412.11499 [cs.AI] https://arxiv.org/abs/2412.11499
[41] Zhumin Chu, Qingyao Ai, Yiteng Tu, Haitao Li, and Yiqun Liu. 2024. Automatic Large Language Model Evaluation via Peer Review. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 384–393.
[42] John Joon Young Chung, Ece Kamar, and Saleema Amershi. 2023. Increasing diversity while maintaining accuracy: Text data generation with large language models and human interventions. *arXiv preprint arXiv:2306.04140* (2023).
[43] Woojin Chung, Jiwoo Hong, Na Min An, James Thorne, and Se-Young Yun. 2024. Stable Language Model Pre-training by Reducing Embedding Variability. *arXiv preprint arXiv:2409.07787* (2024).
[44] Comfy. 2025. ComfyUI. https://www.comfy.org/zh-cn/. Accessed: 2025-05-15.

[45] Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm. *Company Blog of Databricks* (2023).

[46] Domenico Cotroneo, Cristina Improta, Pietro Liguori, and Roberto Natella. 2024. Vulnerabilities in ai code generators: Exploring targeted data poisoning attacks. In *Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension*. 280–292.

[47] Yingqian Cui, Jie Ren, Yuping Lin, Han Xu, Pengfei He, Yue Xing, Lingjuan Lyu, Wenqi Fan, Hui Liu, and Jiliang Tang. 2025. Ft-shield: A watermark against unauthorized fine-tuning in text-to-image diffusion models. *ACM SIGKDD Explorations Newsletter* 26, 2 (2025), 76–88.

[48] Sunhao Dai, Chen Xu, Shicheng Xu, Liang Pang, Zhenhua Dong, and Jun Xu. 2024. Bias and unfairness in information retrieval systems: New challenges in the llm era. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 6437–6447.

[49] Xiangxiang Dai, Jin Li, Xutong Liu, Anqi Yu, and John Lui. 2024. Cost-Effective Online Multi-LLM Selection with Versatile Reward Models. *arXiv preprint arXiv:2405.16587* (2024).

[50] Arghavan Moradi Dakhel, Amin Nikanjam, Vahid Majdinasab, Foutse Khomh, and Michel C Desmarais. 2024. Effective test generation using pre-trained large language models and mutation testing. *Information and Software Technology* 171 (2024), 107468.

[51] Badhan Chandra Das, M Hadi Amini, and Yanzhao Wu. 2025. Security and privacy challenges of large language models: A survey. *Comput. Surveys* 57, 6 (2025), 1–39.

[52] Sarkar Snigdha Sarathi Das, Ranran Haoran Zhang, Peng Shi, Wenpeng Yin, and Rui Zhang. 2023. Unified Low-Resource Sequence Labeling by Sample-Aware Dynamic Sparse Finetuning. In *Conference on Empirical Methods in Natural Language Processing*. https://api.semanticscholar.org/CorpusID:265043702

[53] Bibhu Dash. 2024. Zero-Trust Architecture (ZTA): Designing an AI-Powered Cloud Security Framework for LLMs' Black Box Problems. *Available at SSRN 4726625* (2024).

[54] Daniel DeAlcala, Aythami Morales, Julian Fierrez, Gonzalo Mancera, Ruben Tolosana, and Javier Ortega-Garcia. 2024. Is my data in your ai model? membership inference test with application to face images. *arXiv preprint arXiv:2402.09225* (2024).

[55] Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. 2025. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. *Advances in Neural Information Processing Systems* 37 (2025), 82895–82920.

[56] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems* 35 (2022), 30318–30332.

[57] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems* 36 (2024).

[58] Nobel Dhar, Bobin Deng, Dan Lo, Xiaofeng Wu, Liang Zhao, and Kun Suo. 2024. An empirical analysis and resource footprint study of deploying large language models on edge devices. In *Proceedings of the 2024 ACM Southeast Conference*. 69–76.

[59] Josu Diaz-De-Arcaya, Juan López-De-Armentia, Raúl Miñón, Iker Lasa Ojanguren, and Ana I Torre-Bastida. 2024. Large Language Model Operations (LLMOps): Definition, Challenges, and Lifecycle Management. In *2024 9th International Conference on Smart and Sustainable Technologies (SpliTech)*. IEEE, 1–4.

[60] Bosheng Ding, Chengwei Qin, Ruochen Zhao, Tianze Luo, Xinze Li, Guizhen Chen, Wenhan Xia, Junjie Hu, Luu Anh Tuan, and Shafiq Joty. 2024. Data augmentation using llms: Data perspectives, learning paradigms and challenges. In *Findings of the Association for Computational Linguistics ACL 2024*. 1679–1705.

[61] Yangruibo Ding, Zijian Wang, Wasi Ahmad, Hantian Ding, Ming Tan, Nihal Jain, Murali Krishna Ramanathan, Ramesh Nallapati, Parminder Bhatia, Dan Roth, et al. 2023. Crosscodeeval: A diverse and multilingual benchmark for cross-file code completion. *Advances in Neural Information Processing Systems* 36 (2023), 46701–46723.

[62] Yangruibo Ding, Zijian Wang, Wasi Ahmad, Hantian Ding, Ming Tan, Nihal Jain, Murali Krishna Ramanathan, Ramesh Nallapati, Parminder Bhatia, Dan Roth, et al. 2024. Crosscodeeval: A diverse and multilingual benchmark for cross-file code completion. *Advances in Neural Information Processing Systems* 36 (2024).

[63] Ming Dong, Kang Xue, Bolong Zheng, and Tingting He. 2024. Data-oriented Dynamic Fine-tuning Parameter Selection Strategy for FISH Mask based Efficient Fine-tuning. *arXiv preprint arXiv:2403.08484* (2024).

[64] Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Wei Shen, Limao Xiong, Yuhao Zhou, Xiao Wang, Zhiheng Xi, Xiaoran Fan, et al. 2024. LoRAMoE: Alleviating world knowledge forgetting in large language models via MoE-style plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1932–1945.

[65] Wenyu Du, Tongxu Luo, Zihan Qiu, Zeyu Huang, Yikang Shen, Reynold Cheng, Yike Guo, and Jie Fu. 2025. Stacking your transformers: A closer look at model growth for efficient llm pre-training. *Advances in Neural Information*

Processing Systems 37 (2025), 10491–10540.
[66] Xueying Du, Mingwei Liu, Kaixin Wang, Hanlin Wang, Junwei Liu, Yixuan Chen, Jiayi Feng, Chaofeng Sha, Xin Peng, and Yiling Lou. 2023. Classeval: A manually-crafted benchmark for evaluating llms on class-level code generation. *arXiv preprint arXiv:2308.01861* (2023).
[67] Jiangfei Duan, Shuo Zhang, Zerui Wang, Lijuan Jiang, Wenwen Qu, Qinghao Hu, Guoteng Wang, Qizhen Weng, Hang Yan, Xingcheng Zhang, et al. 2024. Efficient training of large language models on distributed infrastructures: a survey. *arXiv preprint arXiv:2407.20018* (2024).
[68] Yucong Duan. 2024. The Large Language Model (LLM) Bias Evaluation (Age Bias). *DIKWP Research Group International Standard Evaluation. DOI* 10 (2024).
[69] Kristina Dzeparoska, Ali Tizghadam, and Alberto Leon-Garcia. 2024. Intent Assurance using LLMs guided by Intent Drift. *arXiv preprint arXiv:2402.00715* (2024).
[70] Lilian Edwards. 2021. The EU AI Act: a summary of its significance and scope. *Artificial Intelligence (the EU AI Act)* 1 (2021).
[71] Abul Ehtesham, Aditi Singh, Gaurav Kumar Gupta, and Saket Kumar. 2025. A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP). *arXiv preprint arXiv:2505.02279* (2025).
[72] EleutherAI. 2021. lm-evaluation-harness: A framework for evaluating language models on a wide range of tasks. https://github.com/EleutherAI/lm-evaluation-harness.
[73] Jonathan Evertz, Merlin Chlosta, Lea Schönherr, and Thorsten Eisenhofer. 2024. Whispers in the Machine: Confidentiality in LLM-integrated Systems. *arXiv preprint arXiv:2402.06922* (2024).
[74] Hugging Face. 2023. Open LLM Leaderboard: Track, rank, and evaluate open LLMs and chatbots. https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.
[75] Hugging Face. 2023. PEFT: Parameter-Efficient Fine-Tuning. https://github.com/huggingface/peft.
[76] Hugging Face. 2025. Hugging Face Datasets: A Community Library for Datasets. https://huggingface.co/datasets.
[77] Hugging Face. 2025. Inference Endpoints: Machine Learning At Your Service. https://endpoints.huggingface.co/.
[78] Angela Fan, Beliz Gokkaya, Mark Harman, Mitya Lyubarskiy, Shubho Sengupta, Shin Yoo, and Jie M Zhang. 2023. Large language models for software engineering: Survey and open problems. In *2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE)*. IEEE, 31–53.
[79] Alina Fastowski and Gjergji Kasneci. 2024. Understanding Knowledge Drift in LLMs through Misinformation. *arXiv preprint arXiv:2409.07085* (2024).
[80] Muhammad Fawi. 2024. Curlora: Stable llm continual fine-tuning and catastrophic forgetting mitigation. *arXiv preprint arXiv:2408.14572* (2024).
[81] Jia Feng, Jiachen Liu, Cuiyun Gao, Chun Yong Chong, Chaozheng Wang, Shan Gao, and Xin Xia. 2024. Complexcodeeval: A benchmark for evaluating large code models on more complex code. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*. 1895–1906.
[82] Shangbin Feng, Wenxuan Ding, Alisa Liu, Zifeng Wang, Weijia Shi, Yike Wang, Zejiang Shen, Xiaochuang Han, Hunter Lang, Chen-Yu Lee, et al. 2025. When One LLM Drools, Multi-LLM Collaboration Rules. *arXiv preprint arXiv:2502.04506* (2025).
[83] Shangbin Feng, Weijia Shi, Yike Wang, Wenxuan Ding, Vidhisha Balachandran, and Yulia Tsvetkov. 2024. Don't Hallucinate, Abstain: Identifying LLM Knowledge Gaps via Multi-LLM Collaboration. *arXiv preprint arXiv:2402.00367* (2024).
[84] Shangbin Feng, Taylor Sorensen, Yuhan Liu, Jillian Fisher, Chan Young Park, Yejin Choi, and Yulia Tsvetkov. 2024. Modular pluralism: Pluralistic alignment via multi-llm collaboration. *arXiv preprint arXiv:2406.15951* (2024).
[85] Tao Feng, Lizhen Qu, Niket Tandon, Zhuang Li, Xiaoxi Kang, and Gholamreza Haffari. 2024. From pre-training corpora to large language models: What factors influence llm performance in causal discovery tasks? *arXiv preprint arXiv:2407.19638* (2024).
[86] Maximilian T Fischer, Yannick Metz, Lucas Joos, Matthias Miller, and Daniel A Keim. 2024. MULTI-CASE: A Transformer-based Ethics-aware Multimodal Investigative Intelligence Framework. *arXiv preprint arXiv:2401.01955* (2024).
[87] Giorgio Franceschelli and Mirco Musolesi. 2024. On the creativity of large language models. *AI & SOCIETY* (2024), 1–11.
[88] Tingchen Fu, Mrinank Sharma, Philip Torr, Shay B Cohen, David Krueger, and Fazl Barez. 2024. PoisonBench: Assessing Large Language Model Vulnerability to Data Poisoning. *arXiv preprint arXiv:2410.08811* (2024).
[89] Yao Fu, Leyang Xue, Yeqi Huang, Andrei-Octavian Brabete, Dmitrii Ustiugov, Yuvraj Patel, and Luo Mai. 2024. Serverlessllm: Low-latency serverless inference for large language models. In *18th USENIX Symposium on Operating Systems Design and Implementation*. USENIX Association, 135–153.

[90] Gad Gad, Aya Farrag, Ahmed Aboulfotouh, Khaled Bedda, Zubair Md Fadlullah, and Mostafa M Fouda. 2024. Joint self-organizing maps and knowledge-distillation-based communication-efficient federated learning for resource-constrained UAV-IoT systems. *IEEE Internet of Things Journal* 11, 9 (2024), 15504–15522.
[91] Gad Gad, Eyad Gad, Zubair Md Fadlullah, Mostafa M Fouda, and Nei Kato. 2024. Communication-efficient and privacy-preserving federated learning via joint knowledge distillation and differential privacy in bandwidth-constrained networks. *IEEE Transactions on Vehicular Technology* (2024).
[92] Kanishk Gandhi, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah Goodman. 2024. Understanding social reasoning in language models with language models. *Advances in Neural Information Processing Systems* 36 (2024).
[93] Chujie Gao, Dongping Chen, Qihui Zhang, Yue Huang, Yao Wan, and Lichao Sun. 2024. Llm-as-a-coauthor: The challenges of detecting llm-human mixcase. *arXiv preprint arXiv:2401.05952* (2024).
[94] Shuzheng Gao, Wenxin Mao, Cuiyun Gao, Li Li, Xing Hu, Xin Xia, and Michael R Lyu. 2024. Learning in the wild: Towards leveraging unlabeled data for effectively tuning pre-trained code models. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 1–13.
[95] Senay A Gebreab, Khaled Salah, Raja Jayaraman, Muhammad Habib ur Rehman, and Samer Ellaham. 2024. LLM-Based Framework for Administrative Task Automation in Healthcare. In *2024 12th International Symposium on Digital Forensics and Security (ISDFS)*. IEEE, 1–7.
[96] Georgi Gerganov and contributors. 2023. llama.cpp: Inference of LLaMA models in pure C/C++. https://github.com/ggml-org/llama.cpp.
[97] Linyuan Gong, Sida Wang, Mostafa Elhoushi, and Alvin Cheung. 2024. Evaluation of llms on syntax-aware code fill-in-the-middle tasks. *arXiv preprint arXiv:2403.04814* (2024).
[98] Zhuocheng Gong, Jiahao Liu, Jingang Wang, Xunliang Cai, Dongyan Zhao, and Rui Yan. 2024. What makes quantization for large language model hard? an empirical study from the lens of perturbation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 18082–18089.
[99] Google. 2025. Gemini: Your Personal AI Assistant. https://gemini.google.com/.
[100] Sagar Goyal, Eti Rastogi, Sree Prasanna Rajagopal, Dong Yuan, Fen Zhao, Jai Chintagunta, Gautam Naik, and Jeff Ward. 2024. Healai: A healthcare llm for effective medical documentation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 1167–1168.
[101] Ryan Greenblatt, Carson Denison, Benjamin Wright, Fabien Roger, Monte MacDiarmid, Sam Marks, Johannes Treutlein, Tim Belonax, Jack Chen, David Duvenaud, et al. 2024. Alignment faking in large language models. *arXiv preprint arXiv:2412.14093* (2024).
[102] Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. 2023. Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*. 79–90.
[103] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594* (2024).
[104] Jia Gu, Liang Pang, Huawei Shen, and Xueqi Cheng. 2024. Do LLMs Play Dice? Exploring Probability Distribution Sampling in Large Language Models for Behavioral Simulation. *arXiv preprint arXiv:2404.09043* (2024).
[105] Sylvain Gugger, Thomas Wolf, Lysandre Debut, and contributors. 2020. Transformers: State-of-the-art Natural Language Processing. https://github.com/huggingface/transformers.
[106] Neel Guha, Julian Nyarko, Daniel Ho, Christopher Ré, Adam Chilton, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel Rockmore, Diego Zambrano, et al. 2024. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *Advances in Neural Information Processing Systems* 36 (2024).
[107] Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yu Wu, YK Li, et al. 2024. DeepSeek-Coder: When the Large Language Model Meets Programming–The Rise of Code Intelligence. *arXiv preprint arXiv:2401.14196* (2024).
[108] Junfeng Guo, Yiming Li, Ruibo Chen, Yihan Wu, Heng Huang, et al. 2025. ZeroMark: Towards Dataset Ownership Verification without Disclosing Watermark. *Advances in Neural Information Processing Systems* 37 (2025), 120468–120500.
[109] Yiduo Guo, Jie Fu, Huishuai Zhang, Dongyan Zhao, and Yikang Shen. 2024. Efficient continual pre-training by mitigating the stability gap. *arXiv preprint arXiv:2406.14833* (2024).
[110] Zishan Guo, Renren Jin, Chuang Liu, Yufei Huang, Dan Shi, Linhao Yu, Yan Liu, Jiaxuan Li, Bojian Xiong, Deyi Xiong, et al. 2023. Evaluating large language models: A comprehensive survey. *arXiv preprint arXiv:2310.19736* (2023).
[111] Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. 2024. Model editing at scale leads to gradual and catastrophic forgetting. *arXiv preprint arXiv:2401.07453* (2024).
[112] Muhammad Usman Hadi, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. 2023. A survey on large language models: Applications, challenges, limitations, and practical usage. *Authorea Preprints* 3 (2023).

[113] Songyue Han, Mingyu Wang, Jialong Zhang, Dongdong Li, and Junhong Duan. 2024. A Review of Large Language Models: Fundamental Architectures, Key Technological Evolutions, Interdisciplinary Technologies Integration, Optimization and Compression Techniques, Applications, and Challenges. *Electronics* 13, 24 (2024), 5040.

[114] Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608* (2024).

[115] Zixu Hao, Huiqiang Jiang, Shiqi Jiang, Ju Ren, and Ting Cao. 2024. Hybrid slm and llm for edge-cloud collaborative inference. In *Proceedings of the Workshop on Edge and Mobile Foundation Models*. 36–41.

[116] Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. 2024. LLM-rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. *arXiv preprint arXiv:2501.00274* (2024).

[117] Shabnam Hassani, Mehrdad Sabetzadeh, and Daniel Amyot. 2025. An empirical study on LLM-based classification of requirements-related provisions in food-safety regulations. *Empirical Software Engineering* 30, 3 (2025), 72.

[118] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. 2024. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354* (2024).

[119] Chaoyang He, Shen Li, Mahdi Soltanolkotabi, and Salman Avestimehr. 2021. PipeTransformer: Automated elastic pipelining for distributed training of large-scale models. In *International Conference on Machine Learning*. PMLR, 4150–4159.

[120] Jiaao He and Jidong Zhai. 2024. FastDecode: High-Throughput GPU-Efficient LLM Serving using Heterogeneous Pipelines. *arXiv preprint arXiv:2403.11421* (2024).

[121] Ying He, Jingcheng Fang, F Richard Yu, and Victor C Leung. 2024. Large language models (LLMs) inference offloading and resource allocation in cloud-edge computing: An active inference approach. *IEEE Transactions on Mobile Computing* (2024).

[122] Tharindu B Hewage, Shashikant Ilager, Maria Rodriguez Read, and Rajkumar Buyya. 2025. Aging-aware CPU Core Management for Embodied Carbon Amortization in Cloud LLM Inference. *arXiv preprint arXiv:2501.15829* (2025).

[123] Soka Hisaharo, Yuki Nishimura, and Aoi Takahashi. 2024. Optimizing llm inference clusters for enhanced performance and energy efficiency. *Authorea Preprints* (2024).

[124] Guiyang Hou, Yongliang Shen, and Weiming Lu. 2024. Progressive Tuning: Towards Generic Sentiment Abilities for Large Language Models. In *Findings of the Association for Computational Linguistics ACL 2024*. 14392–14402.

[125] Xinyi Hou, Jiahao Han, Yanjie Zhao, and Haoyu Wang. 2025. Unveiling the Landscape of LLM Deployment in the Wild: An Empirical Study. *arXiv preprint arXiv:2505.02502* (2025).

[126] Xinyi Hou, Yanjie Zhao, Yue Liu, Zhou Yang, Kailong Wang, Li Li, Xiapu Luo, David Lo, John Grundy, and Haoyu Wang. 2024. Large language models for software engineering: A systematic literature review. *ACM Transactions on Software Engineering and Methodology* 33, 8 (2024), 1–79.

[127] Xinyi Hou, Yanjie Zhao, Shenao Wang, and Haoyu Wang. 2025. Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions. *arXiv preprint arXiv:2503.23278* (2025).

[128] Chia-Yi Hsu, Yu-Lin Tsai, Chih-Hsun Lin, Pin-Yu Chen, Chia-Mu Yu, and Chun-Ying Huang. 2024. Safe LoRA: the Silver Lining of Reducing Safety Risks when Fine-tuning Large Language Models. *arXiv preprint arXiv:2405.16833* (2024).

[129] Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. 2024. MARS: A Benchmark for Multi-LLM Algorithmic Routing System. In *ICLR 2024 Workshop: How Far Are We From AGI*.

[130] 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).

[131] Chun-Yin Huang, Kartik Srinivas, Xin Zhang, and Xiaoxiao Li. 2024. Overcoming data and model heterogeneities in decentralized federated learning via synthetic anchors. *arXiv preprint arXiv:2405.11525* (2024).

[132] Dong Huang, Guangtao Zeng, Jianbo Dai, Meng Luo, Han Weng, Yuhao Qing, Heming Cui, Zhijiang Guo, and Jie M Zhang. 2024. Effi-Code: Unleashing Code Efficiency in Language Models. *arXiv preprint arXiv:2410.10209* (2024).

[133] Dong Huang, Jie M Zhang, Michael Luck, Qingwen Bu, Yuhao Qing, and Heming Cui. 2023. Agentcoder: Multi-agent-based code generation with iterative testing and optimisation. *arXiv preprint arXiv:2312.13010* (2023).

[134] Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. 2024. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. *arXiv preprint arXiv:2403.01244* (2024).

[135] Wei Huang, Xudong Ma, Haotong Qin, Xingyu Zheng, Chengtao Lv, Hong Chen, Jie Luo, Xiaojuan Qi, Xianglong Liu, and Michele Magno. 2024. How good are low-bit quantized llama3 models? an empirical study. *arXiv e-prints* (2024), arXiv–2404.

[136] Wei Huang, Xingyu Zheng, Xudong Ma, Haotong Qin, Chengtao Lv, Hong Chen, Jie Luo, Xiaojuan Qi, Xianglong Liu, and Michele Magno. 2024. An empirical study of llama3 quantization: From llms to mllms. *Visual Intelligence* 2, 1 (2024), 36.

[137] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186* (2024).

[138] Bo Hui, Haolin Yuan, Neil Gong, Philippe Burlina, and Yinzhi Cao. 2024. Pleak: Prompt leaking attacks against large language model applications. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 3600–3614.

[139] Fushuo Huo, Wenchao Xu, Jingcai Guo, Haozhao Wang, and Song Guo. 2024. C2KD: Bridging the Modality Gap for Cross-Modal Knowledge Distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16006–16015.

[140] Adam Ibrahim, Benjamin Thérien, Kshitij Gupta, Mats L Richter, Quentin Anthony, Timothée Lesort, Eugene Belilovsky, and Irina Rish. 2024. Simple and scalable strategies to continually pre-train large language models. *arXiv preprint arXiv:2403.08763* (2024).

[141] Intel. 2025. Intel® Software Guard Extensions (Intel® SGX). https://www.intel.com/content/www/us/en/products/docs/accelerator-engines/software-guard-extensions.html.

[142] Myeongjun Jang, Antonios Georgiadis, Yiyun Zhao, and Fran Silavong. 2024. DriftWatch: A Tool that Automatically Detects Data Drift and Extracts Representative Examples Affected by Drift. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*. 335–346.

[143] Hyesung Jeon, Yulhwa Kim, and Jae-joon Kim. 2024. L4q: Parameter efficient quantization-aware training on large language models via lora-wise lsq. *arXiv preprint arXiv:2402.04902* (2024).

[144] Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. 2024. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems* 36 (2024).

[145] Bowen Jiang, Yangxinyu Xie, Xiaomeng Wang, Weijie J Su, Camillo Jose Taylor, and Tanwi Mallick. 2024. Multi-modal and multi-agent systems meet rationality: A survey. In *ICML 2024 Workshop on LLMs and Cognition*.

[146] Gangwei Jiang, Caigao Jiang, Zhaoyi Li, Siqiao Xue, Jun Zhou, Linqi Song, Defu Lian, and Ying Wei. 2024. Interpretable catastrophic forgetting of large language model fine-tuning via instruction vector. *arXiv preprint arXiv:2406.12227* (2024).

[147] Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. 2024. A survey on large language models for code generation. *arXiv preprint arXiv:2406.00515* (2024).

[148] Shuli Jiang, Swanand Ravindra Kadhe, Yi Zhou, Farhan Ahmed, Ling Cai, and Nathalie Baracaldo. 2024. Turning Generative Models Degenerate: The Power of Data Poisoning Attacks. *arXiv preprint arXiv:2407.12281* (2024).

[149] Xiaotang Jiang, Huan Wang, Yiliu Chen, Ziqi Wu, Lichuan Wang, Bin Zou, Yafeng Yang, Zongyang Cui, Yu Cai, Tianhang Yu, et al. 2020. MNN: A universal and efficient inference engine. *Proceedings of Machine Learning and Systems* 2 (2020), 1–13.

[150] Junfeng Jiao, Saleh Afroogh, Yiming Xu, and Connor Phillips. 2024. Navigating llm ethics: Advancements, challenges, and future directions. *arXiv preprint arXiv:2406.18841* (2024).

[151] Hongpeng Jin and Yanzhao Wu. 2024. CE-CoLLM: Efficient and Adaptive Large Language Models Through Cloud-Edge Collaboration. *arXiv preprint arXiv:2411.02829* (2024).

[152] Lyudong Jin, Yanning Zhang, Yanhan Li, Shurong Wang, Howard H Yang, Jian Wu, and Meng Zhang. 2025. MoE^2: Optimizing Collaborative Inference for Edge Large Language Models. *arXiv preprint arXiv:2501.09410* (2025).

[153] Renren Jin, Jiangcun Du, Wuwei Huang, Wei Liu, Jian Luan, Bin Wang, and Deyi Xiong. 2024. A comprehensive evaluation of quantization strategies for large language models. In *Findings of the Association for Computational Linguistics ACL 2024*. 12186–12215.

[154] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and applications of large language models. *arXiv preprint arXiv:2307.10169* (2023).

[155] Damjan Kalajdzievski. 2024. Scaling laws for forgetting when fine-tuning large language models. *arXiv preprint arXiv:2401.05605* (2024).

[156] Junmo Kang, Leonid Karlinsky, Hongyin Luo, Zhen Wang, Jacob Hansen, James Glass, David Cox, Rameswar Panda, Rogerio Feris, and Alan Ritter. 2024. Self-moe: Towards compositional large language models with self-specialized experts. *arXiv preprint arXiv:2406.12034* (2024).

[157] Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. 2023. ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and individual differences* 103 (2023), 102274.

[158] Krishnaram Kenthapadi, Mehrnoosh Sameki, and Ankur Taly. 2024. Grounding and evaluation for large language models: Practical challenges and lessons learned (survey). In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 6523–6533.

[159] Mohammed Khan, Priyam Mehta, Ananth Sankar, Umashankar Kumaravelan, Sumanth Doddapaneni, Suriyaprasaad B, Varun G, Sparsh Jain, Anoop Kunchukuttan, Pratyush Kumar, Raj Dabre, and Mitesh Khapra. 2024. IndicLLMSuite: A Blueprint for Creating Pre-training and Fine-Tuning Datasets for Indian Languages. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 15831–15879. https://doi.org/10.18653/v1/2024.acl-long.843

[160] Mohammad Abdullah Matin Khan, M Saiful Bari, Xuan Long Do, Weishi Wang, Md Rizwan Parvez, and Shafiq Joty. 2023. xcodeeval: A large scale multilingual multitask benchmark for code understanding, generation, translation and retrieval. *arXiv preprint arXiv:2303.03004* (2023).

[161] Sanghyeon Kim, Hyunmo Yang, Yunghyun Kim, Youngjoon Hong, and Eunbyung Park. 2024. Hydra: Multi-head low-rank adaptation for parameter efficient fine-tuning. *Neural Networks* (2024), 106414.

[162] Siwon Kim, Sangdoo Yun, Hwaran Lee, Martin Gubri, Sungroh Yoon, and Seong Joon Oh. 2024. Propile: Probing privacy leakage in large language models. *Advances in Neural Information Processing Systems* 36 (2024).

[163] Young Jin Kim, Ammar Ahmad Awan, Alexandre Muzio, Andres Felipe Cruz Salinas, Liyang Lu, Amr Hendy, Samyam Rajbhandari, Yuxiong He, and Hany Hassan Awadalla. 2021. Scalable and efficient moe training for multitask multilingual models. *arXiv preprint arXiv:2109.10465* (2021).

[164] Jing Yu Koh, Daniel Fried, and Russ R Salakhutdinov. 2024. Generating images with multimodal language models. *Advances in Neural Information Processing Systems* 36 (2024).

[165] Patrick Tser Jern Kon, Jiachen Liu, Qiuyi Ding, Yiming Qiu, Zhenning Yang, Yibo Huang, Jayanth Srinivasa, Myungjin Lee, Mosharaf Chowdhury, and Ang Chen. 2025. Curie: Toward rigorous and automated scientific experimentation with ai agents. *arXiv preprint arXiv:2502.16069* (2025).

[166] Rui Kong, Qiyang Li, Xinyu Fang, Qingtian Feng, Qingfeng He, Yazhu Dong, Weijun Wang, Yuanchun Li, Linghe Kong, and Yunxin Liu. 2024. LoRA-Switch: Boosting the Efficiency of Dynamic LLM Adapters via System-Algorithm Co-design. *arXiv preprint arXiv:2405.17741* (2024).

[167] Weirui Kuang, Bingchen Qian, Zitao Li, Daoyuan Chen, Dawei Gao, Xuchen Pan, Yuexiang Xie, Yaliang Li, Bolin Ding, and Jingren Zhou. 2024. Federatedscope-llm: A comprehensive package for fine-tuning large language models in federated learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5260–5271.

[168] Sonu Kumar, Anubhav Girdhar, Ritesh Patil, and Divyansh Tripathi. 2025. MCP Guardian: A Security-First Layer for Safeguarding MCP-Based AI System. *arXiv preprint arXiv:2504.12757* (2025).

[169] Shachi H Kumar, Saurav Sahay, Sahisnu Mazumder, Eda Okur, Ramesh Manuvinakurike, Nicole Beckage, Hsuan Su, Hung-yi Lee, and Lama Nachman. 2024. Decoding biases: Automated methods and llm judges for gender bias detection in language models. *arXiv preprint arXiv:2408.03907* (2024).

[170] Alexey Kurakin, Natalia Ponomareva, Umar Syed, Liam MacDermed, and Andreas Terzis. 2023. Harnessing large-language models to generate private synthetic text. *arXiv preprint arXiv:2306.01684* (2023).

[171] Malgorzata Lazuka, Andreea Anghel, and Thomas Parnell. 2024. LLM-Pilot: Characterize and Optimize Performance of your LLM Inference Services. In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 1–18.

[172] Donghyun Lee and Mo Tiwari. 2024. Prompt infection: Llm-to-llm prompt injection within multi-agent systems. *arXiv preprint arXiv:2410.07283* (2024).

[173] Weixian Lei, Yixiao Ge, Kun Yi, Jianfeng Zhang, Difei Gao, Dylan Sun, Yuying Ge, Ying Shan, and Mike Zheng Shou. 2024. Vit-lens: Towards omni-modal representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 26647–26657.

[174] Bingdong Li, Zixiang Di, Yanting Yang, Hong Qian, Peng Yang, Hao Hao, Ke Tang, and Aimin Zhou. 2024. It's Morphing Time: Unleashing the Potential of Multiple LLMs via Multi-objective Optimization. *arXiv preprint arXiv:2407.00487* (2024).

[175] Bohao Li, Yuying Ge, Yixiao Ge, Guangzhi Wang, Rui Wang, Ruimao Zhang, and Ying Shan. 2024. SEED-Bench: Benchmarking Multimodal Large Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13299–13308.

[176] Chen-An Li and Hung-Yi Lee. 2024. Examining forgetting in continual pre-training of aligned large language models. *arXiv preprint arXiv:2401.03129* (2024).

[177] Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, et al. 2024. From generation to judgment: Opportunities and challenges of llm-as-a-judge. *arXiv preprint arXiv:2411.16594* (2024).

[178] Dong Li, Meng Yan, Yaosheng Zhang, Zhongxin Liu, Chao Liu, Xiaohong Zhang, Ting Chen, and David Lo. 2024. CoSec: On-the-Fly Security Hardening of Code LLMs via Supervised Co-decoding. In *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*. 1428–1439.

[179] Hongyu Li, Liang Ding, Meng Fang, and Dacheng Tao. 2024. Revisiting Catastrophic Forgetting in Large Language Model Tuning. *arXiv preprint arXiv:2406.04836* (2024).

[180] Haitao Li, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu. 2024. Llms-as-judges: a comprehensive survey on llm-based evaluation methods. *arXiv preprint arXiv:2412.05579* (2024).

[181] Haoling Li, Xin Zhang, Xiao Liu, Yeyun Gong, Yifan Wang, Yujiu Yang, Qi Chen, and Peng Cheng. 2024. Gradient-Mask Tuning Elevates the Upper Limits of LLM Performance. *arXiv preprint arXiv:2406.15330* (2024).

[182] Jia Li, Ge Li, Xuanming Zhang, Yihong Dong, and Zhi Jin. 2024. EvoCodeBench: An Evolving Code Generation Benchmark Aligned with Real-World Code Repositories. *arXiv preprint arXiv:2404.00599* (2024).

[183] Jing Li, Zhijie Sun, Xuan He, Li Zeng, Yi Lin, Entong Li, Binfan Zheng, Rongqian Zhao, and Xin Chen. 2024. Locmoe: A low-overhead moe for large language model training. *arXiv preprint arXiv:2401.13920* (2024).

[184] Luchang Li, Sheng Qian, Jie Lu, Lunxi Yuan, Rui Wang, and Qin Xie. 2024. Transformer-lite: High-efficiency deployment of large language models on mobile phone gpus. *arXiv preprint arXiv:2403.20041* (2024).

[185] Miaomiao Li, Hao Chen, Yang Wang, Tingyuan Zhu, Weijia Zhang, Kaijie Zhu, Kam-Fai Wong, and Jindong Wang. 2025. Understanding and Mitigating the Bias Inheritance in LLM-based Data Augmentation on Downstream Tasks. *arXiv preprint arXiv:2502.04419* (2025).

[186] Ming Li, Jiuhai Chen, Lichang Chen, and Tianyi Zhou. 2024. Can llms speak for diverse people? tuning llms via debate to generate controllable controversial statements. *arXiv preprint arXiv:2402.10614* (2024).

[187] Miaoge Li, Jingcai Guo, Richard Yi Da Xu, Dongsheng Wang, Xiaofeng Cao, and Song Guo. 2024. TsCA: On the Semantic Consistency Alignment via Conditional Transport for Compositional Zero-Shot Learning. *arXiv preprint arXiv:2408.08703* (2024).

[188] Qinfeng Li, Zhiqiang Shen, Zhenghan Qin, Yangfan Xie, Xuhong Zhang, Tianyu Du, Sheng Cheng, Xun Wang, and Jianwei Yin. 2024. TransLinkGuard: Safeguarding Transformer Models Against Model Stealing in Edge Deployment. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 3479–3488.

[189] Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. 2023. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161* (2023).

[190] Tianhao Li, Shangjie Li, Binbin Xie, Deyi Xiong, and Baosong Yang. 2024. MoE-CT: a novel approach for large language models training with resistance to catastrophic forgetting. *arXiv preprint arXiv:2407.00875* (2024).

[191] Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190* (2021).

[192] Yichen Li, Yun Peng, Yintong Huo, and Michael R Lyu. 2024. Enhancing llm-based coding tools through native integration of ide-derived static context. In *Proceedings of the 1st International Workshop on Large Language Models for Code*. 70–74.

[193] Yannan Li, Yong Yu, Willy Susilo, Zhiyong Hong, and Mohsen Guizani. 2021. Security and privacy for edge intelligence in 5G and beyond networks: Challenges and solutions. *IEEE Wireless Communications* 28, 2 (2021), 63–69.

[194] Zhaowei Li, Wei Wang, YiQing Cai, Xu Qi, Pengyu Wang, Dong Zhang, Hang Song, Botian Jiang, Zhida Huang, and Tao Wang. 2024. Unifiedmllm: Enabling unified representation for multi-modal multi-tasks with large language model. *arXiv preprint arXiv:2408.02503* (2024).

[195] Zhaowei Li, Qi Xu, Dong Zhang, Hang Song, Yiqing Cai, Qi Qi, Ran Zhou, Junting Pan, Zefeng Li, Vu Tu, et al. 2024. Groundinggpt: Language enhanced multi-modal grounding model. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 6657–6678.

[196] Zhang Li, Biao Yang, Qiang Liu, Zhiyin Ma, Shuo Zhang, Jingxu Yang, Yabo Sun, Yuliang Liu, and Xiang Bai. 2024. Monkey: Image resolution and text label are important things for large multi-modal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 26763–26773.

[197] Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. 2024. Vila: On pre-training for visual language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 26689–26699.

[198] Luyang Lin, Lingzhi Wang, Jinsong Guo, and Kam-Fai Wong. 2024. Investigating bias in llm-based bias detection: Disparities between llms and human perception. *arXiv preprint arXiv:2403.14896* (2024).

[199] Sam Lin, Wenyue Hua, Zhenting Wang, Mingyu Jin, Lizhou Fan, and Yongfeng Zhang. 2025. EmojiPrompt: Generative Prompt Obfuscation for Privacy-Preserving Communication with Cloud-based LLMs. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 12342–12361.

[200] Yang Lin, Xinyu Ma, Xu Chu, Yujie Jin, Zhibang Yang, Yasha Wang, and Hong Mei. 2024. Lora dropout as a sparsity regularizer for overfitting control. *arXiv preprint arXiv:2404.09610* (2024).
[201] Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. 2024. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. *arXiv preprint arXiv:2405.04532* (2024).
[202] Yen-Ting Lin and Yun-Nung Chen. 2023. Llm-eval: Unified multi-dimensional automatic evaluation for open-domain conversations with large language models. *arXiv preprint arXiv:2305.13711* (2023).
[203] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. DeepSeek-V3 Technical Report. *arXiv preprint arXiv:2412.19437* (2024).
[204] Bingchang Liu, Chaoyu Chen, Zi Gong, Cong Liao, Huan Wang, Zhichao Lei, Ming Liang, Dajun Chen, Min Shen, Hailian Zhou, et al. 2024. Mftcoder: Boosting code llms with multitask fine-tuning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5430–5441.
[205] Chengyuan Liu, Yangyang Kang, Shihang Wang, Lizhi Qing, Fubang Zhao, Changlong Sun, Kun Kuang, and Fei Wu. 2024. More than catastrophic forgetting: Integrating general capabilities for domain-specific llms. *arXiv preprint arXiv:2405.17830* (2024).
[206] Dong Liu. 2024. Contemporary Model Compression on Large Language Models Inference. *arXiv preprint arXiv:2409.01990* (2024).
[207] Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. 2023. Mitigating hallucination in large multi-modal models via robust instruction tuning. In *The Twelfth International Conference on Learning Representations*.
[208] Hongyi Liu, Zirui Liu, Ruixiang Tang, Jiayi Yuan, Shaochen Zhong, Yu-Neng Chuang, Li Li, Rui Chen, and Xia Hu. 2024. LoRA-as-an-Attack! Piercing LLM Safety Under The Share-and-Play Scenario. *arXiv preprint arXiv:2403.00108* (2024).
[209] Jingyu Liu, Jiaen Lin, and Yong Liu. 2024. How much can rag help the reasoning of llm? *arXiv preprint arXiv:2410.02338* (2024).
[210] Junwei Liu, Kaixin Wang, Yixuan Chen, Xin Peng, Zhenpeng Chen, Lingming Zhang, and Yiling Lou. 2024. Large language model-based agents for software engineering: A survey. *arXiv preprint arXiv:2409.02977* (2024).
[211] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2024. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems* 36 (2024).
[212] Lian Liu, Haimeng Ren, Long Cheng, Zhaohui Xu, Yudong Pan, Mengdi Wang, Xiaowei Li, Yinhe Han, and Ying Wang. 2024. COMET: Towards Partical W4A4KV4 LLMs Serving. *arXiv preprint arXiv:2410.12168* (2024).
[213] Liang Liu, Dong Zhang, Shoushan Li, Guodong Zhou, and Erik Cambria. 2024. Two Heads are Better than One: Zero-shot Cognitive Reasoning via Multi-LLM Knowledge Fusion. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 1462–1472.
[214] Minghao Liu, Zonglin Di, Jiaheng Wei, Zhongruo Wang, Hengxiang Zhang, Ruixuan Xiao, Haoyu Wang, Jinlong Pang, Hao Chen, Ankit Shah, et al. 2024. Automatic dataset construction (adc): Sample collection, data curation, and beyond. *arXiv preprint arXiv:2408.11338* (2024).
[215] Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. 2024. When moe meets llms: Parameter efficient fine-tuning for multi-task medical applications. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1104–1114.
[216] Shigang Liu, Di Cao, Junae Kim, Tamas Abraham, Paul Montague, Seyit Camtepe, Jun Zhang, and Yang Xiang. 2024. {EaTVul}:{ChatGPT-based} Evasion Attack Against Software Vulnerability Detection. In *33rd USENIX Security Symposium (USENIX Security 24)*. 7357–7374.
[217] Tong Liu, Zizhuang Deng, Guozhu Meng, Yuekang Li, and Kai Chen. 2024. Demystifying rce vulnerabilities in llm-integrated apps. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 1716–1730.
[218] Tianyang Liu, Canwen Xu, and Julian McAuley. 2023. Repobench: Benchmarking repository-level code auto-completion systems. *arXiv preprint arXiv:2306.03091* (2023).
[219] Xin Liu, Yichen Zhu, Jindong Gu, Yunshi Lan, Chao Yang, and Yu Qiao. 2025. Mm-safetybench: A benchmark for safety evaluation of multimodal large language models. In *European Conference on Computer Vision*. Springer, 386–403.
[220] Yang Liu, Jiahuan Cao, Chongyu Liu, Kai Ding, and Lianwen Jin. 2024. Datasets for large language models: A comprehensive survey. *arXiv preprint arXiv:2402.18041* (2024).
[221] Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, et al. 2023. Prompt Injection attack against LLM-integrated Applications. *arXiv preprint arXiv:2306.05499* (2023).
[222] 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.

[223] Yilun Liu, Shimin Tao, Xiaofeng Zhao, Ming Zhu, Wenbing Ma, Junhao Zhu, Chang Su, Yutai Hou, Miao Zhang, Min Zhang, et al. 2024. Coachlm: Automatic instruction revisions improve the data quality in llm instruction tuning. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 5184–5197.

[224] Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2023. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888* (2023).

[225] Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. 2024. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750* (2024).

[226] Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. *arXiv preprint arXiv:2406.15126* (2024).

[227] Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, et al. 2023. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity. *arXiv preprint arXiv:2305.13169* (2023).

[228] Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, et al. 2024. Starcoder 2 and the stack v2: The next generation. *arXiv preprint arXiv:2402.19173* (2024).

[229] Daqin Luo, Chengjian Feng, Yuxuan Nong, and Yiqing Shen. 2024. Autom3l: An automated multimodal machine learning framework with large language models. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 8586–8594.

[230] Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, et al. 2025. Large Language Model Agent: A Survey on Methodology, Applications and Challenges. *arXiv preprint arXiv:2503.21460* (2025).

[231] Xiang Luo, Zhiwen Tang, Jin Wang, and Xuejie Zhang. 2024. Zero-Shot Cross-Domain Dialogue State Tracking via Dual Low-Rank Adaptation. *arXiv preprint arXiv:2407.21633* (2024).

[232] Alisia Lupidi, Carlos Gemmell, Nicola Cancedda, Jane Dwivedi-Yu, Jason Weston, Jakob Foerster, Roberta Raileanu, and Maria Lomeli. 2024. Source2synth: Synthetic data generation and curation grounded in real data sources. *arXiv preprint arXiv:2409.08239* (2024).

[233] Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Lifeng Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The era of 1-bit llms: All large language models are in 1.58 bits. *arXiv preprint arXiv:2402.17764* 1 (2024).

[234] Wanqin Ma, Chenyang Yang, and Christian Kästner. 2024. (Why) Is My Prompt Getting Worse? Rethinking Regression Testing for Evolving LLM APIs. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*. 166–171.

[235] Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems* 36 (2023), 21702–21720.

[236] Zilin Ma, Yiyang Mei, Krzysztof Z Gajos, and Ian Arawjo. 2024. Schrödinger's Update: User Perceptions of Uncertainties in Proprietary Large Language Model Updates. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. 1–9.

[237] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems* 36 (2023), 46534–46594.

[238] Pratyush Maini, Hengrui Jia, Nicolas Papernot, and Adam Dziedzic. 2024. LLM Dataset Inference: Did you train on my dataset? *arXiv preprint arXiv:2406.06443* (2024).

[239] Pratyush Maini, Hengrui Jia, Nicolas Papernot, and Adam Dziedzic. 2025. LLM Dataset Inference: Did you train on my dataset? *Advances in Neural Information Processing Systems* 37 (2025), 124069–124092.

[240] Paul Joe Maliakel, Shashikant Ilager, and Ivona Brandic. 2025. Investigating Energy Efficiency and Performance Trade-offs in LLM Inference Across Tasks and DVFS Settings. *arXiv preprint arXiv:2501.08219* (2025).

[241] Mihai Masala, Denis C Ilie-Ablachim, Alexandru Dima, Dragos Corlatescu, Miruna Zavelca, Ovio Olaru, Simina Terian, Andrei Terian, Marius Leordeanu, Horia Velicu, et al. 2024. " Vorbe\c {s} ti Rom\ˆ ane\c {s} te?" A Recipe to Train Powerful Romanian LLMs with English Instructions. *arXiv preprint arXiv:2406.18266* (2024).

[242] Arsalan Masoudifard, Mohammad Mowlavi Sorond, Moein Madadi, Mohammad Sabokrou, and Elahe Habibi. 2024. Leveraging Graph-RAG and Prompt Engineering to Enhance LLM-Based Automated Requirement Traceability and Compliance Checks. *arXiv preprint arXiv:2412.08593* (2024).

[243] Timothy R McIntosh, Teo Susnjak, Nalin Arachchilage, Tong Liu, Paul Watters, and Malka N Halgamuge. 2024. Inadequacies of large language model benchmarks in the era of generative artificial intelligence. *arXiv preprint arXiv:2402.09880* (2024).

[244] Matthieu Meeus, Shubham Jain, Marek Rei, and Yves-Alexandre de Montjoye. 2024. Did the neurons read your book? document-level membership inference for large language models. In *33rd USENIX Security Symposium (USENIX Security 24)*. 2369–2385.
[245] Ahmed Menshawy, Zeeshan Nawaz, and Mahmoud Fahmy. 2024. Navigating Challenges and Technical Debt in Large Language Models Deployment. In *Proceedings of the 4th Workshop on Machine Learning and Systems*. 192–199.
[246] Microsoft. 2025. Azure OpenAI Service. https://azure.microsoft.com/en-us/products/ai-services/openai-service.
[247] Asit Mishra, Jorge Albericio Latorre, Jeff Pool, Darko Stosic, Dusan Stosic, Ganesh Venkatesh, Chong Yu, and Paulius Micikevicius. 2021. Accelerating sparse deep neural networks. *arXiv preprint arXiv:2104.08378* (2021).
[248] Yuhong Mo, Hao Qin, Yushan Dong, Ziyi Zhu, and Zhenglin Li. 2024. Large language model (llm) ai text generation detection based on transformer deep learning algorithm. *arXiv preprint arXiv:2405.06652* (2024).
[249] Apoorve Mohan, Mengmei Ye, Hubertus Franke, Mudhakar Srivatsa, Zhuoran Liu, and Nelson Mimura Gonzalez. 2024. Securing AI Inference in the Cloud: Is CPU-GPU Confidential Computing Ready?. In *2024 IEEE 17th International Conference on Cloud Computing (CLOUD)*. IEEE, 164–175.
[250] Ahmad Mohsin, Helge Janicke, Adrian Wood, Iqbal H Sarker, Leandros Maglaras, and Naeem Janjua. 2024. Can we trust large language models generated code? a framework for in-context learning, security patterns, and code evaluations across diverse llms. *arXiv preprint arXiv:2406.12513* (2024).
[251] Ida Momennejad, Hosein Hasanbeig, Felipe Vieira Frujeri, Hiteshi Sharma, Nebojsa Jojic, Hamid Palangi, Robert Ness, and Jonathan Larson. 2024. Evaluating cognitive maps and planning in large language models with CogEval. *Advances in Neural Information Processing Systems* 36 (2024).
[252] Mahdi Morafah, Vyacheslav Kungurtsev, Hojin Chang, Chen Chen, and Bill Lin. 2024. Towards Diverse Device Heterogeneous Federated Learning via Task Arithmetic Knowledge Integration. *arXiv preprint arXiv:2409.18461* (2024).
[253] Alhassan Mumuni and Fuseini Mumuni. 2025. Large language models for artificial general intelligence (AGI): A survey of foundational principles and approaches. *arXiv preprint arXiv:2501.03151* (2025).
[254] Vineeth Sai Narajala and Idan Habler. 2025. Enterprise-Grade Security for the Model Context Protocol (MCP): Frameworks and Mitigation Strategies. *arXiv preprint arXiv:2504.08623* (2025).
[255] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, et al. 2021. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the international conference for high performance computing, networking, storage and analysis*. 1–15.
[256] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2023. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435* (2023).
[257] Nihal V Nayak, Yiyang Nan, Avi Trost, and Stephen H Bach. 2024. Learning to generate instruction tuning datasets for zero-shot task adaptation. *arXiv preprint arXiv:2402.18334* (2024).
[258] Mahmoud Nazzal, Issa Khalil, Abdallah Khreishah, and NhatHai Phan. 2024. PromSec: Prompt Optimization for Secure Generation of Functional Source Code with Large Language Models (LLMs). In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 2266–2280.
[259] Shiwen Ni, Dingwei Chen, Chengming Li, Xiping Hu, Ruifeng Xu, and Min Yang. 2023. Forgetting before learning: Utilizing parametric arithmetic for knowledge updating in large language models. *arXiv preprint arXiv:2311.08011* (2023).
[260] Xuefei Ning, Zifu Wang, Shiyao Li, Zinan Lin, Peiran Yao, Tianyu Fu, Matthew B. Blaschko, Guohao Dai, Huazhong Yang, and Yu Wang. 2024. Can LLMs Learn by Teaching for Better Reasoning? A Preliminary Study. arXiv:2406.14629 [cs.CL] https://arxiv.org/abs/2406.14629
[261] Kosuke Nishida, Kyosuke Nishida, and Kuniko Saito. 2024. Initialization of large language models via reparameterization to mitigate loss spikes. *arXiv preprint arXiv:2410.05052* (2024).
[262] Dominic Novado, Eliyah Cohen, and Jacob Foster. 2024. Multi-tier privacy protection for large language models using differential privacy. *Authorea Preprints* (2024).
[263] NVIDIA. 2025. TensorRT: High-Performance Deep Learning Inference Library. https://github.com/NVIDIA/TensorRT.
[264] Hyungjun Oh, Kihong Kim, Jaemin Kim, Sungkyun Kim, Junyeol Lee, Du-seong Chang, and Jiwon Seo. 2024. Exegpt: Constraint-aware resource scheduling for llm inference. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*. 369–384.
[265] Ollama. 2025. Ollama Official Website. https://ollama.com/. Accessed: 2025-05-15.
[266] OpenAI. 2023. OpenAI Platform API Reference. https://platform.openai.com/docs/api-reference/introduction.
[267] OpenAI. 2025. Introducing GPT-4.5. https://openai.com/index/introducing-gpt-4-5/ Accessed: 2025-04-20.
[268] OpenAI. 2025. OpenAI API Platform. https://openai.com/api/.

[269] Malte Ostendorff, Pedro Ortiz Suarez, Lucas Fonseca Lage, and Georg Rehm. 2024. LLM-Datasets: An Open Framework for Pretraining Datasets of Large Language Models. In *First Conference on Language Modeling*.

[270] Tiago P Pagano, Rafael B Loureiro, Fernanda VN Lisboa, Rodrigo M Peixoto, Guilherme AS Guimarães, Gustavo OR Cruz, Maira M Araujo, Lucas L Santos, Marco AS Cruz, Ewerton LS Oliveira, et al. 2023. Bias and unfairness in machine learning models: a systematic review on datasets, tools, fairness metrics, and identification and mitigation methods. *Big data and cognitive computing* 7, 1 (2023), 15.

[271] Artidoro Pagnoni, Ram Pasunuru, Pedro Rodriguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason Weston, Luke Zettlemoyer, et al. 2024. Byte Latent Transformer: Patches Scale Better Than Tokens. *arXiv preprint arXiv:2412.09871* (2024).

[272] Leyi Pan, Aiwei Liu, Shiyu Huang, Yijian Lu, Xuming Hu, Lijie Wen, Irwin King, and Philip S Yu. 2025. Can LLM Watermarks Robustly Prevent Unauthorized Knowledge Distillation? *arXiv preprint arXiv:2502.11598* (2025).

[273] Jinlong Pang, Jiaheng Wei, Ankit Parag Shah, Zhaowei Zhu, Yaxuan Wang, Chen Qian, Yang Liu, Yujia Bao, and Wei Wei. 2024. Improving Data Efficiency via Curating LLM-Driven Rating Systems. *arXiv preprint arXiv:2410.10877* (2024).

[274] Qi Pang, Shengyuan Hu, Wenting Zheng, and Virginia Smith. 2024. Attacking llm watermarks by exploiting their strengths. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*.

[275] Xiaoyi Pang, Jiahui Hu, Peng Sun, Ju Ren, and Zhibo Wang. 2024. When Federated Learning Meets Knowledge Distillation. *IEEE Wireless Communications* 31, 5 (2024), 208–214.

[276] Samuel Panterino and Matthew Fellington. 2024. Dynamic moving target defense for mitigating targeted llm prompt injection. *Authorea Preprints* (2024).

[277] David Pape, Sina Mavali, Thorsten Eisenhofer, and Lea Schönherr. 2024. Prompt obfuscation for large language models. *arXiv preprint arXiv:2409.11026* (2024).

[278] Bhrij Patel, Vishnu Sashank Dorbala, and Amrit Singh Bedi. 2024. Embodied Question Answering via Multi-LLM Systems. *arXiv preprint arXiv:2406.10918* (2024).

[279] Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. 2024. Splitwise: Efficient generative llm inference using phase splitting. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 118–132.

[280] Pankayaraj Pathmanathan, Souradip Chakraborty, Xiangyu Liu, Yongyuan Liang, and Furong Huang. 2024. Is poisoning a real threat to LLM alignment? Maybe more so than you think. *arXiv preprint arXiv:2406.12091* (2024).

[281] Rodrigo Pedro, Miguel E Coimbra, Daniel Castro, Paulo Carreira, and Nuno Santos. 2024. Prompt-to-SQL Injections in LLM-Integrated Web Applications: Risks and Defenses. In *2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE)*. IEEE Computer Society, 76–88.

[282] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116* (2023).

[283] Letian Peng, Zilong Wang, Feng Yao, Zihan Wang, and Jingbo Shang. 2024. Metaie: Distilling a meta model from llm for all kinds of information extraction tasks. *arXiv preprint arXiv:2404.00457* (2024).

[284] Nikhil Pesati. 2024. Security Considerations for Large Language Model Use: Implementation Research in Securing LLM-Integrated Applications. *Available at SSRN 4962370* (2024).

[285] Matthew E Peters, Sebastian Ruder, and Noah A Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. *arXiv preprint arXiv:1903.05987* (2019).

[286] Julien Piet, Maha Alrashed, Chawin Sitawarin, Sizhe Chen, Zeming Wei, Elizabeth Sun, Basel Alomair, and David Wagner. 2024. Jatmo: Prompt injection defense by task-specific finetuning. In *European Symposium on Research in Computer Security*. Springer, 105–124.

[287] Max Ploner and Alan Akbik. 2024. Parameter-Efficient Fine-Tuning: Is There An Optimal Subset of Parameters to Tune?. In *Findings of the Association for Computational Linguistics: EACL 2024*. 1743–1759.

[288] PyTorch Team. 2023. Executorch Overview. https://pytorch.org/executorch-overview.

[289] Shuang Qiao, Haiyang Xu, Chenhong Cao, Wei Gong, Si Chen, and Jiangchuan Liu. 2025. PrismPrompt: Layering prompt-enhanced cloud-edge collaborative language model towards healthcare. *IEEE Network* (2025).

[290] Laiqiao Qin, Tianqing Zhu, Wanlei Zhou, and Philip S Yu. 2024. Knowledge distillation in federated learning: A survey on long lasting challenges and new solutions. *arXiv preprint arXiv:2406.10861* (2024).

[291] Guanqiao Qu, Qiyuan Chen, Wei Wei, Zheng Lin, Xianhao Chen, and Kaibin Huang. 2025. Mobile edge intelligence for large language models: A contemporary survey. *IEEE Communications Surveys & Tutorials* (2025).

[292] Brandon Radosevich and John Halloran. 2025. MCP Safety Audit: LLMs with the Model Context Protocol Allow Major Security Exploits. *arXiv preprint arXiv:2504.03767* (2025).

[293] Nikitha Rao, Kush Jain, Uri Alon, Claire Le Goues, and Vincent J Hellendoorn. 2023. CAT-LM training language models on aligned code and tests. In *2023 38th IEEE/ACM International Conference on Automated Software Engineering*

(ASE). IEEE, 409–420.

[294] Hanoona Rasheed, Muhammad Maaz, Sahal Shaji, Abdelrahman Shaker, Salman Khan, Hisham Cholakkal, Rao M Anwer, Eric Xing, Ming-Hsuan Yang, and Fahad S Khan. 2024. Glamm: Pixel grounding large multimodal model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13009–13018.

[295] Deevashwer Rathee, Dacheng Li, Ion Stoica, Hao Zhang, and Raluca Popa. 2024. Mpc-minimized secure llm inference. *arXiv preprint arXiv:2408.03561* (2024).

[296] Partha Pratim Ray and Mohan Pratap Pradhan. 2024. LLMEdge: A Novel Framework for Localized LLM Inferencing at Resource Constrained Edge. In *2024 International Conference on IoT Based Control Networks and Intelligent Systems (ICICNIS)*. IEEE, 1–8.

[297] Weijieying Ren, Xinlong Li, Lei Wang, Tianxiang Zhao, and Wei Qin. 2024. Analyzing and Reducing Catastrophic Forgetting in Parameter Efficient Tuning. *arXiv preprint arXiv:2402.18865* (2024).

[298] Martin Riddell, Ansong Ni, and Arman Cohan. 2024. Quantifying contamination in evaluating code generation capabilities of language models. *arXiv preprint arXiv:2403.04811* (2024).

[299] Adam Roe, Samuel Richardson, Joseph Schneider, Anthony Cummings, Nicholas Forsberg, and Jonathan Klein. 2024. Semantic drift mitigation in large language model knowledge retention using the residual knowledge stability concept. *Authorea Preprints* (2024).

[300] T-YLPG Ross and GKHP Dollár. 2017. Focal loss for dense object detection. In *proceedings of the IEEE conference on computer vision and pattern recognition*. 2980–2988.

[301] Paul Röttger, Valentin Hofmann, Valentina Pyatkin, Musashi Hinck, Hannah Rose Kirk, Hinrich Schütze, and Dirk Hovy. 2024. Political compass or spinning arrow? towards more meaningful evaluations for values and opinions in large language models. *arXiv preprint arXiv:2402.16786* (2024).

[302] Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2020. Adapterdrop: On the efficiency of adapters in transformers. *arXiv preprint arXiv:2010.11918* (2020).

[303] Lorenzo Sani, Alex Iacob, Zeyu Cao, Bill Marino, Yan Gao, Tomas Paulik, Wanru Zhao, William F Shen, Preslav Aleksandrov, Xinchi Qiu, et al. 2024. The future of large language model pre-training is federated. *arXiv preprint arXiv:2405.10853* (2024).

[304] Helen Santos, Anthony Schmidt, Caspian Dimitrov, Christopher Antonucci, and Dorian Kuznetsov. 2024. Adaptive contextualization in large language models using dynamic semantic drift encoding. *Authorea Preprints* (2024).

[305] Pritam Sarkar and Ali Etemad. 2024. Xkd: Cross-modal knowledge distillation with domain alignment for video representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 14875–14885.

[306] Robert Schmirler, Michael Heinzinger, and Burkhard Rost. 2024. Fine-tuning protein language models boosts predictions across diverse tasks. *Nature Communications* 15, 1 (2024), 7407.

[307] Philipp Schoenegger, Indre Tuminauskaite, Peter S Park, Rafael Valdece Sousa Bastos, and Philip E Tetlock. 2024. Wisdom of the silicon crowd: LLM ensemble prediction capabilities rival human crowd accuracy. *Science Advances* 10, 45 (2024), eadp1528.

[308] Leo Schwinn, David Dobre, Sophie Xhonneux, Gauthier Gidel, and Stephan Günnemann. 2024. Soft prompt threats: Attacking safety alignment and unlearning in open-source llms through the embedding space. *Advances in Neural Information Processing Systems* 37 (2024), 9086–9116.

[309] Weizhou Shen, Chenliang Li, Hongzhan Chen, Ming Yan, Xiaojun Quan, Hehong Chen, Ji Zhang, and Fei Huang. 2024. Small llms are weak tool learners: A multi-llm agent. *arXiv preprint arXiv:2401.07324* (2024).

[310] Xuan Shen, Pu Zhao, Yifan Gong, Zhenglun Kong, Zheng Zhan, Yushu Wu, Ming Lin, Chao Wu, Xue Lin, and Yanzhi Wang. 2025. Search for efficient large language models. *Advances in Neural Information Processing Systems* 37 (2025), 139294–139315.

[311] Jieke Shi, Zhou Yang, and David Lo. 2024. Efficient and green large language models for software engineering: Vision and the road ahead. *ACM Transactions on Software Engineering and Methodology* (2024).

[312] Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2023. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789* (2023).

[313] Weiyan Shi, Ryan Shea, Si Chen, Chiyuan Zhang, Ruoxi Jia, and Zhou Yu. 2022. Just fine-tune twice: Selective differential privacy for large language models. *arXiv preprint arXiv:2204.07667* (2022).

[314] Anup Shirgaonkar, Nikhil Pandey, Nazmiye Ceren Abay, Tolga Aktas, and Vijay Aski. 2024. Knowledge Distillation Using Frontier Open-source LLMs: Generalizability and the Role of Synthetic Data. *arXiv preprint arXiv:2410.18588* (2024).

[315] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. 2023. The curse of recursion: Training on generated data makes models forget. *arXiv preprint arXiv:2305.17493* (2023).

[316] Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2024. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers. *arXiv preprint arXiv:2409.04109* (2024).

[317] Aditi Singh, Abul Ehtesham, Saket Kumar, and Tala Talaei Khoei. 2025. A Survey of the Model Context Protocol (MCP): Standardizing Context to Enhance Large Language Models (LLMs). (2025).
[318] Neha Singh, Jatin Rupchandani, and Mainak Adhikari. 2023. Personalized federated learning for heterogeneous edge device: Self-knowledge distillation approach. *IEEE Transactions on Consumer Electronics* 70, 1 (2023), 4625–4632.
[319] Shivalika Singh, Freddie Vargus, Daniel Dsouza, Börje F Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura OMahony, et al. 2024. Aya dataset: An open-access collection for multilingual instruction tuning. *arXiv preprint arXiv:2402.06619* (2024).
[320] Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. 2022. Large language models encode clinical knowledge. *arXiv preprint arXiv:2212.13138* (2022).
[321] Jonas Sjöström and Stefan Cronholm. 2024. Meta-requirements for LLM-Based Knowledge Exploration Tools in Information Systems Research. In *International Conference on Design Science Research in Information Systems and Technology*. Springer, 424–439.
[322] Mahdi Soleimani, Grace Jia, In Gim, Seung-seob Lee, and Anurag Khandelwal. 2025. Wiretapping LLMs: Network Side-Channel Attacks on Interactive LLM Services. *Cryptology ePrint Archive* (2025).
[323] Adir Solomon, Meira Levy, Dikla Agur-Cohen, Malak Younis, and Efrat Moshe. 2024. Requirements Engineering for LLM: The Case of Digital Inquiries Application. In *2024 IEEE 32nd International Requirements Engineering Conference Workshops (REW)*. IEEE, 370–375.
[324] Yifan Song, Guoyin Wang, Sujian Li, and Bill Yuchen Lin. 2024. The good, the bad, and the greedy: Evaluation of llms should not ignore non-determinism. *arXiv preprint arXiv:2407.10457* (2024).
[325] Shikhar Srivastava, Md Yousuf Harun, Robik Shrestha, and Christopher Kanan. 2024. Improving multimodal large language models using continual learning. *arXiv preprint arXiv:2410.19925* (2024).
[326] Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. 2024. On the Self-Verification Limitations of Large Language Models on Reasoning and Planning Tasks. arXiv:2402.08115 [cs.AI] https://arxiv.org/abs/2402.08115
[327] Jovan Stojkovic, Esha Choukse, Chaojie Zhang, Inigo Goiri, and Josep Torrellas. 2024. Towards Greener LLMs: Bringing Energy-Efficiency to the Forefront of LLM Inference. *arXiv preprint arXiv:2403.20306* (2024).
[328] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Josep Torrellas, and Esha Choukse. 2024. Dynamollm: Designing llm inference clusters for performance and energy efficiency. *arXiv preprint arXiv:2408.00741* (2024).
[329] Winnie Street, John Oliver Siy, Geoff Keeling, Adrien Baranes, Benjamin Barnett, Michael McKibben, Tatenda Kanyere, Alison Lentz, Robin IM Dunbar, et al. 2024. LLMs achieve adult human performance on higher-order theory of mind tasks. *arXiv preprint arXiv:2405.18870* (2024).
[330] Dimitris Stripelis, Zijian Hu, Jipeng Zhang, Zhaozhuo Xu, Alay Dilipbhai Shah, Han Jin, Yuhang Yao, Salman Avestimehr, and Chaoyang He. 2024. TensorOpera Router: A Multi-Model Router for Efficient LLM Inference. *arXiv preprint arXiv:2408.12320* (2024).
[331] Haochen Sun, Jason Li, and Hongyang Zhang. 2024. zkllm: Zero knowledge proofs for large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 4405–4419.
[332] Liangtai Sun, Yang Han, Zihan Zhao, Da Ma, Zhennan Shen, Baocai Chen, Lu Chen, and Kai Yu. 2024. Scieval: A multi-level large language model evaluation benchmark for scientific research. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 19053–19061.
[333] Quan Sun, Yufeng Cui, Xiaosong Zhang, Fan Zhang, Qiying Yu, Yueze Wang, Yongming Rao, Jingjing Liu, Tiejun Huang, and Xinlong Wang. 2024. Generative multimodal models are in-context learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14398–14409.
[334] Xuchen Suo. 2024. Signed-Prompt: A new approach to prevent prompt injection attacks against LLM-integrated applications. In *AIP Conference Proceedings*, Vol. 3194. AIP Publishing.
[335] Rao Surapaneni, Miku Jha, Michael Vakoc, and Todd Segal. 2025. Announcing the Agent2Agent Protocol (A2A). https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/ Accessed: 2025-04-25.
[336] Zhendong Tan, Xingjun Zhang, and Zheng Wei. 2024. WRP: Weight Recover Prune for Structured Sparsity. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 6433–6443.
[337] Jiaxi Tang, Yoel Drori, Daryl Chang, Maheswaran Sathiamoorthy, Justin Gilmer, Li Wei, Xinyang Yi, Lichan Hong, and Ed H Chi. 2023. Improving training stability for multitask ranking models in recommender systems. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4882–4893.
[338] DeepSpeed Team. 2020. DeepSpeed: Accelerating Deep Learning Training and Inference. https://github.com/deepspeedai/DeepSpeed.
[339] TensorFlow. 2025. TensorFlow Serving Guide. https://www.tensorflow.org/tfx/guide/serving Accessed: 2025-04-26.
[340] Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. 2023. Large language models in medicine. *Nature medicine* 29, 8 (2023), 1930–1940.

[341] Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Cheng-Zhong Xu. 2025. Hydralora: An asymmetric lora architecture for efficient fine-tuning. *Advances in Neural Information Processing Systems* 37 (2025), 9565–9584.
[342] Kushal Tirumala, Daniel Simig, Armen Aghajanyan, and Ari Morcos. 2023. D4: Improving llm pretraining via document de-duplication and diversification. *Advances in Neural Information Processing Systems* 36 (2023), 53983–53995.
[343] Meng Tong, Kejiang Chen, Jie Zhang, Yuang Qi, Weiming Zhang, Nenghai Yu, Tianwei Zhang, and Zhikun Zhang. 2023. InferDPT: Privacy-preserving inference for black-box large language model. *arXiv preprint arXiv:2310.12214* (2023).
[344] E Paul Torrance. 1966. Torrance tests of creative thinking. *Educational and psychological measurement* (1966).
[345] Christoph Treude and Hideaki Hata. 2023. She elicits requirements and he tests: Software engineering gender bias in large language models. In *2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR)*. IEEE, 624–629.
[346] Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. 2022. Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. *arXiv preprint arXiv:2210.07558* (2022).
[347] Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2024. Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change. *Advances in Neural Information Processing Systems* 36 (2024).
[348] Tempest A van Schaik and Brittany Pugh. 2024. A field guide to automatic evaluation of llm-generated summaries. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2832–2836.
[349] vLLM Project. 2025. vLLM: A high-throughput and memory-efficient inference and serving engine for LLMs. https://github.com/vllm-project/vllm.
[350] Gianmario Voria, Gemma Catolino, and Fabio Palomba. 2024. Is Attention All You Need? Toward a Conceptual Model for Social Awareness in Large Language Models. In *Proceedings of the 2024 IEEE/ACM First International Conference on AI Foundation Models and Software Engineering* (Lisbon, Portugal) *(FORGE '24)*. Association for Computing Machinery, New York, NY, USA, 69–73. https://doi.org/10.1145/3650105.3652294
[351] Ivan Vykopal, Simon Ostermann, and Marián Šimko. 2024. Soft Language Prompts for Language Transfer. *arXiv preprint arXiv:2407.02317* (2024).
[352] Angelina Wang, Alexander Liu, Ryan Zhang, Anat Kleiman, Leslie Kim, Dora Zhao, Iroha Shirai, Arvind Narayanan, and Olga Russakovsky. 2022. REVISE: A tool for measuring and mitigating bias in visual datasets. *International Journal of Computer Vision* 130, 7 (2022), 1790–1810.
[353] Cunxiang Wang, Sirui Cheng, Qipeng Guo, Yuanhao Yue, Bowen Ding, Zhikun Xu, Yidong Wang, Xiangkun Hu, Zheng Zhang, and Yue Zhang. 2023. Evaluating open-qa evaluation. *Advances in Neural Information Processing Systems* 36 (2023), 77013–77042.
[354] Cangqing Wang, Yutian Yang, Ruisi Li, Dan Sun, Ruicong Cai, Yuzhu Zhang, Chengqian Fu, and Lillian Floyd. 2024. Adapting llms for efficient context processing through soft prompt compression. *arXiv preprint arXiv:2404.04997* (2024).
[355] Haoxiang Wang, Yong Lin, Wei Xiong, Rui Yang, Shizhe Diao, Shuang Qiu, Han Zhao, and Tong Zhang. 2024. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. *arXiv preprint arXiv:2402.18571* (2024).
[356] Hanbin Wang, Zhenghao Liu, Shuo Wang, Ganqu Cui, Ning Ding, Zhiyuan Liu, and Ge Yu. 2023. Intervenor: Prompting the coding ability of large language models with the interactive chain of repair. *arXiv preprint arXiv:2311.09868* (2023).
[357] Jianxun Wang and Yixiang Chen. 2023. A review on code generation with llms: Application and evaluation. In *2023 IEEE International Conference on Medical Artificial Intelligence (MedAI)*. IEEE, 284–289.
[358] Ruiqi Wang, Jiyu Guo, Cuiyun Gao, Guodong Fan, Chun Yong Chong, and Xin Xia. 2025. Can LLMs Replace Human Evaluators? An Empirical Study of LLM-as-a-Judge in Software Engineering. *arXiv preprint arXiv:2502.06193* (2025).
[359] Sheng Wang, Liheng Chen, Jiyue Jiang, Boyang Xue, Lingpeng Kong, and Chuan Wu. 2024. LoRA Meets Dropout under a Unified Framework. *arXiv preprint arXiv:2403.00812* (2024).
[360] Shijian Wang, Linxin Song, Jieyu Zhang, Ryotaro Shimizu, Ao Luo, Li Yao, Cunjian Chen, Julian McAuley, and Hanqian Wu. 2024. Template Matters: Understanding the Role of Instruction Templates in Multimodal Language Model Evaluation and Training. arXiv:2412.08307 [cs.CV] https://arxiv.org/abs/2412.08307
[361] Shang Wang, Tianqing Zhu, Bo Liu, Ming Ding, Xu Guo, Dayong Ye, Wanlei Zhou, and Philip S Yu. 2024. Unique security and privacy threats of large language model: A comprehensive survey. *arXiv preprint arXiv:2406.07973* (2024).
[362] Tianduo Wang, Shichen Li, and Wei Lu. 2024. Self-Training with Direct Preference Optimization Improves Chain-of-Thought Reasoning. arXiv:2407.18248 [cs.CL] https://arxiv.org/abs/2407.18248
[363] Xiaofei Wang, Xiaoxu Ren, Chao Qiu, Zehui Xiong, Haipeng Yao, and Victor CM Leung. 2022. Integrating edge intelligence and blockchain: What, why, and how. *IEEE Communications Surveys & Tutorials* 24, 4 (2022), 2193–2229.

[364] Xujia Wang, Haiyan Zhao, Shuo Wang, Hanqing Wang, and Zhiyuan Liu. 2024. MALoRA: Mixture of Asymmetric Low-Rank Adaptation for Enhanced Multi-Task Learning. *arXiv preprint arXiv:2410.22782* (2024).
[365] Yihan Wang, Jatin Chauhan, Wei Wang, and Cho-Jui Hsieh. 2024. Universality and limitations of prompt tuning. *Advances in Neural Information Processing Systems* 36 (2024).
[366] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560* (2022).
[367] Yuanyi Wang, Haifeng Sun, Jiabo Wang, Jingyu Wang, Wei Tang, Qi Qi, Shaoling Sun, and Jianxin Liao. 2024. Towards semantic consistency: Dirichlet energy driven robust multi-modal entity alignment. *arXiv preprint arXiv:2401.17859* (2024).
[368] Ya Wang, Zhijian Zhuo, Yutao Zeng, Xun Zhou, Jian Yang, and Xiaoqing Li. 2025. Scale-Distribution Decoupling: Enabling Stable and Effective Training of Large Language Models. *arXiv preprint arXiv:2502.15499* (2025).
[369] Zeyu Wang. 2024. CausalBench: A Comprehensive Benchmark for Evaluating Causal Reasoning Capabilities of Large Language Models. In *Proceedings of the 10th SIGHAN Workshop on Chinese Language Processing (SIGHAN-10)*. 143–151.
[370] Zilan Wang, Junfeng Guo, Jiacheng Zhu, Yiming Li, Heng Huang, Muhao Chen, and Zhengzhong Tu. 2025. Sleepermark: Towards robust watermark against fine-tuning text-to-image diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 8213–8224.
[371] Noa Wegerhoff, Avishag Shapira, Yuval Elovici, and Asaf Shabtai. 2024. DataDetective: Dataset Watermarking for Leaker Identification in ML Training. In *ECAI 2024*. IOS Press, 2442–2451.
[372] Tianjun Wei, Wei Wen, Ruizhi Qiao, Xing Sun, and Jianghong Ma. [n. d.]. RocketEval: Efficient automated LLM evaluation via grading checklist. In *The Thirteenth International Conference on Learning Representations*.
[373] Weights & Biases. 2025. Weights & Biases: The AI Developer Platform. https://wandb.ai/site.
[374] Chenxi Whitehouse, Monojit Choudhury, and Alham Fikri Aji. 2023. LLM-powered data augmentation for enhanced cross-lingual performance. *arXiv preprint arXiv:2305.14288* (2023).
[375] Jan Wichelmann, Anna Pätschke, Luca Wilke, and Thomas Eisenbarth. 2023. Cipherfix: Mitigating ciphertext {Side-Channel} attacks in software. In *32nd USENIX Security Symposium (USENIX Security 23)*. 6789–6806.
[376] Grant Wilkins, Srinivasan Keshav, and Richard Mortier. 2024. Hybrid Heterogeneous Clusters Can Lower the Energy Consumption of LLM Inference Workloads. In *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*. 506–513.
[377] Sunghyeon Woo, Baeseong Park, Byeongwook Kim, Minjung Jo, Se Jung Kwon, Dongsuk Jeon, and Dongsoo Lee. 2025. DropBP: accelerating fine-tuning of large language models by dropping backward propagation. *Advances in Neural Information Processing Systems* 37 (2025), 20170–20197.
[378] Mitchell Wortsman, Tim Dettmers, Luke Zettlemoyer, Ari Morcos, Ali Farhadi, and Ludwig Schmidt. 2023. Stable and low-precision training for large-scale vision-language models. *Advances in Neural Information Processing Systems* 36 (2023), 10271–10298.
[379] Bingyang Wu, Ruidong Zhu, Zili Zhang, Peng Sun, Xuanzhe Liu, and Xin Jin. 2024. {dLoRA}: Dynamically Orchestrating Requests and Adapters for {LoRA}{LLM} Serving. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*. 911–927.
[380] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2024. A survey on large language models for recommendation. *World Wide Web* 27, 5 (2024), 60.
[381] Qinyun Wu, Chao Peng, Pengfei Gao, Ruida Hu, Haoyu Gan, Bo Jiang, Jinhe Tang, Zhiwen Deng, Zhanming Guan, Cuiyun Gao, et al. 2024. Repomastereval: Evaluating code completion via real-world repositories. *arXiv preprint arXiv:2408.03519* (2024).
[382] Xingyu Wu, Sheng-hao Wu, Jibin Wu, Liang Feng, and Kay Chen Tan. 2024. Evolutionary computation in the era of large language model: Survey and roadmap. *IEEE Transactions on Evolutionary Computation* (2024).
[383] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences* 68, 2 (2025), 121101.
[384] Chunqiu Steven Xia, Yinlin Deng, and Lingming Zhang. 2024. Top Leaderboard Ranking= Top Coding Proficiency, Always? EvoEval: Evolving Coding Benchmarks via LLM. *arXiv preprint arXiv:2403.19114* (2024).
[385] Yifei Xia, Fangcheng Fu, Wentao Zhang, Jiawei Jiang, and Bin Cui. 2025. Efficient Multi-task LLM Quantization and Serving for Multiple LoRA Adapters. *Advances in Neural Information Processing Systems* 37 (2025), 63686–63714.
[386] Qianqian Xie, Weiguang Han, Xiao Zhang, Yanzhao Lai, Min Peng, Alejandro Lopez-Lira, and Jimin Huang. 2024. Pixiu: A comprehensive benchmark, instruction dataset and large language model for finance. *Advances in Neural Information Processing Systems* 36 (2024).
[387] Xingyu Xie, Shuicheng Yan, Kim-Chuan Toh, and Tianwen Wei. 2024. Optimization hyper-parameter laws for large language models. *arXiv preprint arXiv:2409.04777* (2024).

[388] Chunlei Xin, Yaojie Lu, Hongyu Lin, Shuheng Zhou, Huijia Zhu, Weiqiang Wang, Zhongyi Liu, Xianpei Han, and Le Sun. 2024. Beyond full fine-tuning: Harnessing the power of LoRA for multi-task instruction tuning. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. 2307–2317.

[389] Yi Xin, Siqi Luo, Haodi Zhou, Junlong Du, Xiaohong Liu, Yue Fan, Qing Li, and Yuntao Du. 2024. Parameter-efficient fine-tuning for pre-trained vision models: A survey. *arXiv preprint arXiv:2402.02242* (2024).

[390] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*. PMLR, 10524–10533.

[391] Cheng Xu, Shuhao Guan, Derek Greene, M Kechadi, et al. 2024. Benchmark Data Contamination of Large Language Models: A Survey. *arXiv preprint arXiv:2406.04244* (2024).

[392] Frank F Xu, Uri Alon, Graham Neubig, and Vincent Josua Hellendoorn. 2022. A systematic evaluation of large language models of code. In *Proceedings of the 6th ACM SIGPLAN international symposium on machine programming*. 1–10.

[393] Hongling Xu, Yice Zhang, Qianlong Wang, and Ruifeng Xu. 2024. DS^2-ABSA: Dual-Stream Data Synthesis with Label Refinement for Few-Shot Aspect-Based Sentiment Analysis. arXiv:2412.14849 [cs.CL] https://arxiv.org/abs/2412.14849

[394] Mengwei Xu, Dongqi Cai, Wangsong Yin, Shangguang Wang, Xin Jin, and Xuanzhe Liu. 2025. Resource-efficient algorithms and systems of foundation models: A survey. *Comput. Surveys* 57, 5 (2025), 1–39.

[395] 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).

[396] Shaoyang Xu, Yongqi Leng, Linhao Yu, and Deyi Xiong. 2024. Self-Pluralising Culture Alignment for Large Language Models. *arXiv preprint arXiv:2410.12971* (2024).

[397] Wenda Xu, Guanglei Zhu, Xuandong Zhao, Liangming Pan, Lei Li, and William Wang. 2024. Pride and prejudice: LLM amplifies self-bias in self-refinement. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 15474–15492.

[398] Yuancheng Xu, Jiarui Yao, Manli Shu, Yanchao Sun, Zichu Wu, Ning Yu, Tom Goldstein, and Furong Huang. 2025. Shadowcast: Stealthy data poisoning attacks against vision-language models. *Advances in Neural Information Processing Systems* 37 (2025), 57733–57764.

[399] Zhentao Xu, Mark Jerome Cruz, Matthew Guevara, Tie Wang, Manasi Deshpande, Xiaofeng Wang, and Zheng Li. 2024. Retrieval-augmented generation with knowledge graphs for customer service question answering. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2905–2909.

[400] Zhaozhuo Xu, Zirui Liu, Beidi Chen, Shaochen Zhong, Yuxin Tang, WANG Jue, Kaixiong Zhou, Xia Hu, and Anshumali Shrivastava. [n. d.]. Soft Prompt Recovers Compressed LLMs, Transferably. In *Forty-first International Conference on Machine Learning*.

[401] Di Xue, Gang Zhao, Zhongqi Fan, Wei Li, Yahong Xu, Zhen Liu, Yin Liu, and Zhongliang Yuan. 2024. Poster: An Exploration of Large Language Models in Malicious Source Code Detection. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 4940–4942.

[402] Fuzhao Xue, Kabir Jain, Mahir Hitesh Shah, Zangwei Zheng, and Yang You. 2023. Instruction in the wild: A user-based instruction dataset.

[403] Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. 2024. Openmoe: An early effort on open mixture-of-experts language models. *arXiv preprint arXiv:2402.01739* (2024).

[404] Lixiang Yan, Lele Sha, Linxuan Zhao, Yuheng Li, Roberto Martinez-Maldonado, Guanliang Chen, Xinyu Li, Yueqiao Jin, and Dragan Gašević. 2024. Practical and ethical challenges of large language models in education: A systematic scoping review. *British Journal of Educational Technology* 55, 1 (2024), 90–112.

[405] Shenao Yan, Shen Wang, Yue Duan, Hanbin Hong, Kiho Lee, Doowon Kim, and Yuan Hong. 2024. An {LLM-Assisted}{Easy-to-Trigger} Backdoor Attack on Code Completion Models: Injecting Disguised Vulnerabilities against Strong Detection. In *33rd USENIX Security Symposium (USENIX Security 24)*. 1795–1812.

[406] Diji Yang, Linda Zeng, Kezhen Chen, and Yi Zhang. 2024. Reinforcing Thinking through Reasoning-Enhanced Reward Models. arXiv:2501.01457 [cs.LG] https://arxiv.org/abs/2501.01457

[407] Fan Yang, Zehao Wang, Haoyu Zhang, Zhenhua Zhu, Xinhao Yang, Guohao Dai, and Yu Wang. 2024. Efficient Deployment of Large Language Model across Cloud-Device Systems. In *2024 IEEE 37th International System-on-Chip Conference (SOCC)*. IEEE, 1–6.

[408] Huan Yang, Deyu Zhang, Yudong Zhao, Yuanchun Li, and Yunxin Liu. 2024. A First Look At Efficient And Secure On-Device LLM Inference Against KV Leakage. In *Proceedings of the 19th Workshop on Mobility in the Evolving Internet Architecture*. 13–18.

[409] Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. 2024. RLCD: Reinforcement Learning from Contrastive Distillation for Language Model Alignment. arXiv:2307.12950 [cs.CL] https://arxiv.org/abs/2307.12950

[410] Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E Gonzalez, and Ion Stoica. 2023. Rethinking benchmark and contamination for language models with rephrased samples. *arXiv preprint arXiv:2311.04850* (2023).

[411] Xinyu Yang, Jixuan Leng, Geyang Guo, Jiawei Zhao, Ryumei Nakada, Linjun Zhang, Huaxiu Yao, and Beidi Chen. 2024. S^2FT: Efficient, Scalable and Generalizable LLM Fine-tuning by Structured Sparsity. *arXiv preprint arXiv:2412.06289* (2024).

[412] Xiaoyu Yang, Jie Lu, and En Yu. 2024. Adapting Multi-modal Large Language Model to Concept Drift in the Long-tailed Open World. *arXiv preprint arXiv:2405.13459* (2024).

[413] Yijun Yang, Ruiyuan Gao, Xiao Yang, Jianyuan Zhong, and Qiang Xu. 2025. Guardt2i: Defending text-to-image models from adversarial prompts. *Advances in Neural Information Processing Systems* 37 (2025), 76380–76403.

[414] Yuqi Yang, Peng-Tao Jiang, Qibin Hou, Hao Zhang, Jinwei Chen, and Bo Li. 2024. Multi-task dense prediction via mixture of low-rank experts. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 27927–27937.

[415] Ziqing Yang, Michael Backes, Yang Zhang, and Ahmed Salem. 2024. Sos! soft prompt attack against open-source large language models. *arXiv preprint arXiv:2407.03160* (2024).

[416] Ziqi Yang, Xuhai Xu, Bingsheng Yao, Ethan Rogers, Shao Zhang, Stephen Intille, Nawar Shara, Guodong Gordon Gao, and Dakuo Wang. 2024. Talk2Care: An LLM-based Voice Assistant for Communication between Healthcare Providers and Older Adults. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 2 (2024), 1–35.

[417] Zheming Yang, Yuanhao Yang, Chang Zhao, Qi Guo, Wenkai He, and Wen Ji. 2024. Perllm: Personalized inference scheduling with edge-cloud collaboration for diverse llm services. *arXiv preprint arXiv:2405.14636* (2024).

[418] Zhiqin Yang, Yonggang Zhang, Yu Zheng, Xinmei Tian, Hao Peng, Tongliang Liu, and Bo Han. 2023. Fedfed: Feature distillation against data heterogeneity in federated learning. *Advances in Neural Information Processing Systems* 36 (2023), 60397–60428.

[419] Hongwei Yao, Jian Lou, and Zhan Qin. 2024. Poisonprompt: Backdoor attack on prompt-based large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7745–7749.

[420] Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. 2024. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing* (2024), 100211.

[421] Zhi Yao, Zhiqing Tang, Jiong Lou, Ping Shen, and Weijia Jia. 2024. Velo: A vector database-assisted cloud-edge collaborative llm qos optimization framework. In *2024 IEEE International Conference on Web Services (ICWS)*. IEEE, 865–876.

[422] Zhewei Yao, Xiaoxia Wu, Cheng Li, Stephen Youn, and Yuxiong He. 2024. Exploring post-training quantization in llms from comprehensive study to low rank compensation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 19377–19385.

[423] Zhiwei Yao, Yang Xu, Hongli Xu, Yunming Liao, and Zuan Xie. 2025. Efficient Deployment of Large Language Models on Resource-constrained Devices. *arXiv preprint arXiv:2501.02438* (2025).

[424] Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, et al. 2024. Justice or prejudice? quantifying biases in llm-as-a-judge. *arXiv preprint arXiv:2410.02736* (2024).

[425] Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. 2024. Openfedllm: Training large language models on decentralized private data via federated learning. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*. 6137–6147.

[426] Dongshuo Yin, Xueting Han, Bin Li, Hao Feng, and Jing Bai. 2024. Parameter-efficient is not sufficient: Exploring parameter, memory, and time efficient adapter tuning for dense predictions. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 1398–1406.

[427] Wangsong Yin, Mengwei Xu, Yuanchun Li, and Xuanzhe Liu. 2024. Llm as a system service on mobile devices. *arXiv preprint arXiv:2403.11805* (2024).

[428] Luke Yoffe, Alfonso Amayuelas, and William Yang Wang. 2024. DebUnc: mitigating hallucinations in large language model agent communication with uncertainty estimations. *arXiv preprint arXiv:2407.06426* (2024).

[429] Runyi Yu, Zhennan Wang, Yinhuai Wang, Kehan Li, Chang Liu, Haoyi Duan, Xiangyang Ji, and Jie Chen. 2023. Lape: Layer-adaptive position embedding for vision transformers with independent layer normalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 5886–5896.

[430] Yangyang Yu, Zhiyuan Yao, Haohang Li, Zhiyang Deng, Yuechen Jiang, Yupeng Cao, Zhi Chen, Jordan Suchow, Zhenyu Cui, Rong Liu, et al. 2025. Fincon: A synthesized llm multi-agent system with conceptual verbal reinforcement for enhanced financial decision making. *Advances in Neural Information Processing Systems* 37 (2025), 137010–137045.

[431] Yue Yu, Yuchen Zhuang, Jieyu Zhang, Yu Meng, Alexander J Ratner, Ranjay Krishna, Jiaming Shen, and Chao Zhang. 2024. Large language model as attributed training data generator: A tale of diversity and bias. *Advances in Neural Information Processing Systems* 36 (2024).

[432] Zhaojian Yu, Xin Zhang, Ning Shang, Yangyu Huang, Can Xu, Yishujie Zhao, Wenxiang Hu, and Qiufeng Yin. 2024. Wavecoder: Widespread and versatile enhancement for code large language models by instruction tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 5140–5153.

[433] Ann Yuan, Daphne Ippolito, Vitaly Nikolaev, Chris Callison-Burch, Andy Coenen, and Sebastian Gehrmann. 2021. Synthbio: A case study in human-ai collaborative curation of text datasets. *arXiv preprint arXiv:2111.06467* (2021).

[434] Siyu Yuan, Kaitao Song, Jiangjie Chen, Xu Tan, Yongliang Shen, Ren Kan, Dongsheng Li, and Deqing Yang. 2024. Easytool: Enhancing llm-based agents with concise tool instruction. *arXiv preprint arXiv:2401.06201* (2024).

[435] Daoguang Zan, Zhirong Huang, Ailun Yu, Shaoxin Lin, Yifan Shi, Wei Liu, Dong Chen, Zongshuai Qi, Hao Yu, Lei Yu, et al. 2024. Swe-bench-java: A github issue resolving benchmark for java. *arXiv preprint arXiv:2408.14354* (2024).

[436] Zhengran Zeng, Yidong Wang, Rui Xie, Wei Ye, and Shikun Zhang. 2024. Coderujb: An executable and unified java benchmark for practical programming scenarios. In *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*. 124–136.

[437] Yuexiang Zhai, Shengbang Tong, Xiao Li, Mu Cai, Qing Qu, Yong Jae Lee, and Yi Ma. 2024. Investigating the catastrophic forgetting in multimodal large language model fine-tuning. In *Conference on Parsimony and Learning*. PMLR, 202–227.

[438] Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. 2024. Injecagent: Benchmarking indirect prompt injections in tool-integrated large language model agents. *arXiv preprint arXiv:2403.02691* (2024).

[439] Biao Zhang, Zhongtao Liu, Colin Cherry, and Orhan Firat. 2024. When scaling meets llm finetuning: The effect of data, model and finetuning method. *arXiv preprint arXiv:2402.17193* (2024).

[440] Duzhen Zhang, Yahan Yu, Jiahua Dong, Chenxing Li, Dan Su, Chenhui Chu, and Dong Yu. 2024. Mm-llms: Recent advances in multimodal large language models. *arXiv preprint arXiv:2401.13601* (2024).

[441] Jiaming Zhang, Xingjun Ma, Xin Wang, Lingyu Qiu, Jiaqi Wang, Yu-Gang Jiang, and Jitao Sang. 2024. Adversarial prompt tuning for vision-language models. In *European Conference on Computer Vision*. Springer, 56–72.

[442] Lin Zhang, Li Shen, Liang Ding, Dacheng Tao, and Ling-Yu Duan. 2022. Fine-tuning global model via data-free knowledge distillation for non-iid federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10174–10183.

[443] Mingjin Zhang, Xiaoming Shen, Jiannong Cao, Zeyang Cui, and Shan Jiang. 2024. Edgeshard: Efficient llm inference via collaborative edge computing. *IEEE Internet of Things Journal* (2024).

[444] Nan Zhang, Yanchi Liu, Xujiang Zhao, Wei Cheng, Runxue Bao, Rui Zhang, Prasenjit Mitra, and Haifeng Chen. 2024. Pruning as a Domain-specific LLM Extractor. *arXiv preprint arXiv:2405.06275* (2024).

[445] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512* (2023).

[446] Quanjun Zhang, Chunrong Fang, Yang Xie, YuXiang Ma, Weisong Sun, Yun Yang, and Zhenyu Chen. 2024. A systematic literature review on large language models for automated program repair. *arXiv preprint arXiv:2405.01466* (2024).

[447] Xiaojin Zhang, Yahao Pang, Yan Kang, Wei Chen, Lixin Fan, Hai Jin, and Qiang Yang. 2025. No free lunch theorem for privacy-preserving llm inference. *Artificial Intelligence* (2025), 104293.

[448] Bowen Zhao, Hannaneh Hajishirzi, and Qingqing Cao. 2024. Apt: Adaptive pruning and tuning pretrained language models for efficient training and inference. *arXiv preprint arXiv:2401.12200* (2024).

[449] Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology* 15, 2 (2024), 1–38.

[450] Huaqin Zhao, Zhengliang Liu, Zihao Wu, Yiwei Li, Tianze Yang, Peng Shu, Shaochen Xu, Haixing Dai, Lin Zhao, Gengchen Mai, et al. 2024. Revolutionizing finance with llms: An overview of applications and insights. *arXiv preprint arXiv:2401.11641* (2024).

[451] Juntao Zhao, Borui Wan, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. LLM-PQ: Serving LLM on Heterogeneous Clusters with Phase-Aware Partition and Adaptive Quantization. *arXiv preprint arXiv:2403.01136* (2024).

[452] Jiachen Zhao, Wenlong Zhao, Andrew Drozdov, Benjamin Rozonoyer, Md Arafat Sultan, Jay Yoon Lee, Mohit Iyyer, and Andrew McCallum. 2024. Multistage collaborative knowledge distillation from a large language model for semi-supervised sequence generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 14201–14214.

[453] Weilin Zhao, Yuxiang Huang, Xu Han, Zhiyuan Liu, Zhengyan Zhang, and Maosong Sun. 2023. CPET: Effective parameter-efficient tuning for compressed large language models. *arXiv preprint arXiv:2307.07705* (2023).
[454] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* 1, 2 (2023).
[455] Zesen Zhao, Shuowei Jin, and Z Morley Mao. 2024. Eagle: Efficient training-free router for multi-llm inference. *arXiv preprint arXiv:2409.15518* (2024).
[456] Junhao Zheng, Xidi Cai, Shengjie Qiu, and Qianli Ma. 2025. Spurious Forgetting in Continual Learning of Language Models. *arXiv preprint arXiv:2501.13453* (2025).
[457] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems* 36 (2024).
[458] Peter Yong Zhong, Siyuan Chen, Ruiqi Wang, McKenna McCall, Ben L Titzer, and Heather Miller. 2025. RTBAS: Defending LLM Agents Against Prompt Injection and Privacy Leakage. *arXiv preprint arXiv:2502.08966* (2025).
[459] Andy Zhou, Bo Li, and Haohan Wang. 2025. Robust prompt optimization for defending language models against jailbreaking attacks. *Advances in Neural Information Processing Systems* 37 (2025), 40184–40211.
[460] Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. 2024. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039* (2024).
[461] Hao Zhou, Chengming Hu, Ye Yuan, Yufei Cui, Yili Jin, Can Chen, Haolun Wu, Dun Yuan, Li Jiang, Di Wu, et al. 2024. Large language model (llm) for telecommunications: A comprehensive survey on principles, key techniques, and opportunities. *IEEE Communications Surveys & Tutorials* (2024).
[462] Han Zhou, Xingchen Wan, Ivan Vulić, and Anna Korhonen. 2024. Autopeft: Automatic configuration search for parameter-efficient fine-tuning. *Transactions of the Association for Computational Linguistics* 12 (2024), 525–542.
[463] Kun Zhou, Yutao Zhu, Zhipeng Chen, Wentong Chen, Wayne Xin Zhao, Xu Chen, Yankai Lin, Ji-Rong Wen, and Jiawei Han. 2023. Don't make your llm an evaluation benchmark cheater. *arXiv preprint arXiv:2311.01964* (2023).
[464] Xin Zhou, Kisub Kim, Bowen Xu, Jiakun Liu, DongGyun Han, and David Lo. 2023. The devil is in the tails: How long-tailed code distributions impact large language models. *arXiv preprint arXiv:2309.03567* (2023).
[465] Yuhang Zhou and Wei Ai. 2024. Teaching-Assistant-in-the-Loop: Improving Knowledge Distillation from Imperfect Teacher Models in Low-Budget Scenarios. *arXiv preprint arXiv:2406.05322* (2024).
[466] Kaijie Zhu, Jindong Wang, Qinlin Zhao, Ruochen Xu, and Xing Xie. 2024. Dynamic Evaluation of Large Language Models by Meta Probing Agents. In *Forty-first International Conference on Machine Learning*. https://openreview.net/forum?id=DwTgy1hXXo
[467] Pengfei Zhu, Yang Sun, Bing Cao, and Qinghua Hu. 2024. Task-customized mixture of adapters for general image fusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 7099–7108.
[468] Qihao Zhu, Daya Guo, Zhihong Shao, Dejian Yang, Peiyi Wang, Runxin Xu, Y Wu, Yukun Li, Huazuo Gao, Shirong Ma, et al. 2024. DeepSeek-Coder-V2: Breaking the Barrier of Closed-Source Models in Code Intelligence. *arXiv preprint arXiv:2406.11931* (2024).
[469] Xun Zhu, Ying Hu, Fanbin Mo, Miao Li, and Ji Wu. 2024. Uni-Med: A Unified Medical Generalist Foundation Model For Multi-Task Learning Via Connector-MoE. *arXiv preprint arXiv:2409.17508* (2024).
[470] Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2024. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics* 12 (2024), 1556–1577.
[471] Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107* (2023).
[472] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. 2021. Data-free knowledge distillation for heterogeneous federated learning. In *International conference on machine learning*. PMLR, 12878–12889.
[473] KOU Zili, Sharad Sinha, HE Wenjian, and Wei Zhang. 2023. Cache side-channel attacks and defenses of the sliding window algorithm in TEEs. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 1–6.
[474] Itamar Zimerman, Allon Adir, Ehud Aharoni, Matan Avitan, Moran Baruch, Nir Drucker, Jenny Lerner, Ramy Masalha, Reut Meiri, and Omri Soceanu. 2024. Power-softmax: Towards secure llm inference over encrypted data. *arXiv preprint arXiv:2410.09457* (2024).
[475] SM Zobaed and Mohsen Amini Salehi. 2025. Confidential Computing Across Edge-To-Cloud for Machine Learning: A Survey Study. *Software: Practice and Experience* (2025).
[476] Nicolas Zucchet and Antonio Orvieto. 2025. Recurrent neural networks: vanishing and exploding gradients are not the end of the story. *Advances in Neural Information Processing Systems* 37 (2025), 139402–139443.