Autonomous Intelligent Systems

REVIEW Open Access

Large language models for PHM: a review of optimization techniques and applications

Tingyi Yu[1], Junya Tang[1], Qingyun Yu[1*], Li Li[1], Ying Liu[2] and Raul Poler[3]

Abstract

The rapid advancement of Large Language Models (LLMs) has created unprecedented opportunities for industrial automation, process optimization, and decision support systems. As industries seek to leverage LLMs for industrial tasks, understanding their architecture, deployment strategies, and fine-tuning methods becomes critical. In this review, we aim to summarize the challenges, key technologies, current status, and future directions of LLM in Prognostics and Health Management(PHM). First, this review introduces deep learning for PHM. We begin by analyzing the architectural considerations and deployment strategies for industrial environments, including acceleration techniques and quantization methods that enable efficient operation on resource-constrained industrial hardware. Second, we investigate Parameter Efficient Fine-Tuning (PEFT) techniques that allow industry-specific adaptation without prohibitive computational costs. Multi-modal capabilities extending LLMs beyond text to process sensor data, images, and time-series information are also discussed. Finally, we explore emerging PHM including anomaly detection systems that identify equipment malfunctions, fault diagnosis frameworks that determine root causes, and specialized question-answering systems that empower workers with instant domain expertise. We conclude by identifying key challenges and future research directions for LLM deployment in PHM. This review provides a timely resource for researchers, engineers, and decision-makers navigating the transformative potential of language models in industry 4.0 environments.

Keywords: LLM for PHM, Parameter Efficient Fine-Tuning, Quantization

1 Introduction

In recent years, machine learning, particularly deep learning, has revolutionized Prognostics and Health Management (PHM) data analysis, establishing itself as a mainstream approach. This transformation primarily involves two key components: feature engineering [1] and machine learning [2]. Feature engineering leverages statistical and signal analysis techniques to extract meaningful insights from PHM monitoring data, while machine learning models, such as Support Vector Machines (SVM) [3, 4] and K-Nearest Neighbors (KNN) [5, 6], facilitate predictive analytics and intelligent decision-making. Despite delivering promising results and enabling a degree of automation, traditional machine learning still relies heavily on manual feature engineering. This dependency not only limits its scalability in handling large volumes of industrial data but also hinders its ability to generalize across diverse applications.

Deep learning [7], which employs multi-layer artificial neural networks for data processing, has become a cornerstone of PHM data analysis due to its ability to automatically handle large volumes of high-dimensional data with strong generalization capabilities. Various deep learning models cater to different PHM tasks, including autoencoders (AE) [8], convolutional neural networks (CNNs) [9], and recurrent neural networks (RNNs) [10]. AE facilitate unsupervised representation learning through data reconstruction, making them highly effective for tasks such as noise reduction, dimension reduction, and

*Correspondence: qingyunyu@tongji.edu.cn
[1]College of Electronic and Information Engineering, Tongji University, Shanghai, 201804, China
Full list of author information is available at the end of the article

anomaly detection. CNNs, leveraging convolution theory, excel in extracting hierarchical features from data, making them particularly suitable for industrial image analysis and periodic time-series processing [9]. Meanwhile, RNNs are designed to capture long-term dependencies in sequential data, making them ideal for analyzing and processing extended time-series signals [11]. By enabling end-to-end learning, deep learning minimizes the need for manual intervention in PHM. However, these models still face challenges related to multitasking, generalization, and cognitive reasoning, particularly in the context of human-computer interaction.

The emergence of ChatGPT [12] marks a new era in AI technology, driven by LLMs that overcome the limitations of traditional AI. These models offer significant advancements in understanding, reasoning, and reuse efficiency, laying a strong foundation for AI applications across various fields. Generalization and reusability are the core strengths of LLMs, enabling their widespread application in natural language processing, computer vision, speech recognition, and beyond. By serving as a common infrastructure for diverse applications, these models allow developers to leverage fine-tuning, zero-shot learning, and other techniques to adapt them for different industries and scenarios.

Despite their potential, the application of LLMs in PHM domains remains in an exploratory phase. There is still no definitive approach to integrating them into PHM tasks or mapping these tasks to areas where LLMs excel. Furthermore, a systematic literature review on the methodologies and techniques for applying LLMs in PHM settings is lacking. Our review is therefore structured as follows. Section 2 examines architectural considerations and deployment strategies for LLMs in industrial environments, including acceleration techniques and various quantization methods that enable efficient operation on industrial hardware. Section 3 examines fine-tuning approaches, with a special focus on Parameter Efficient Fine-Tuning (PEFT) techniques such as adapter tuning, prompt tuning, prefix tuning, and low-rank adaptation (LORA), which enable domain-specific adaptation within industrial constraints. Section 4 explores multi-modal capabilities that extend LLMs beyond text to handle diverse industrial data types. Section 5 examines practical applications, including anomaly detection systems for equipment monitoring, fault diagnosis frameworks for root cause analysis, and specialized question-answering systems that improve worker productivity and decision making. Finally, we summarize current advances, identify persistent challenges, and outline promising future research directions for LLMs in PHM.

2 Architecture and deployment

A Large Language Model (LLM) typically refers to a language model with over ten billion parameters, trained on massive textual datasets. Expanding the model size, data volume, and computational complexity significantly enhances its modeling capabilities, as demonstrated by numerous studies [13, 14]. KM Scaling Law [15] and Chinchilla Scaling Law [16] provide a framework for predicting LLM performance during training. However, some abilities cannot be anticipated solely based on these scaling laws.

One of the most distinguishing features of LLMs is their emergent abilities [17], which do not appear in smaller models but emerge as the model size surpasses a certain threshold, leading to significant performance improvements.

In-Context Learning (ICL): First demonstrated by GPT-3 [13], ICL allows a language model to generate the expected output for a given task by processing a natural language instruction along with a few task demonstrations, without requiring additional training or gradient updates.

Instruction Following: Fine-tuning on datasets containing task descriptions enables LLMs to generalize better across unseen tasks. With instruction fine-tuning, an LLM can follow new instructions without needing explicit examples, improving adaptability and performance.

Step-by-Step Reasoning: Small language models often struggle with complex tasks requiring multiple reasoning steps, such as solving mathematical problems. However, Chain-of-Thought (CoT) prompting enhances LLM performance by guiding it to generate intermediate reasoning steps before arriving at the final answer. This structured reasoning process significantly improves problem-solving capabilities.

To better understand and use LLM, this section introduces the common structure of large models and common ways to reduce costs in industrial deployments. Table 1 organizes the architecture and deployment methods

2.1 Architecture

In 2017, Google introduced the Transformer [47], which quickly set a new benchmark for performance across various natural language processing (NLP) tasks. The Transformer architecture has since become the foundational framework for developing a wide range of LLMs, enabling them to scale to tens or even hundreds of billions of parameters.

Based on Transformer, OpenAI proposed Generative Pre-Training (GPT) [48] based on causal decoder that employs a unidirectional attention mask to ensure that each token can only attend to itself and preceding tokens. This design enables autoregressive generation, where input and output tokens follow the same processing pipeline within the decoder. A prime example of this architecture is GPT-3 [13], which demonstrated not only the effectiveness of causal decoders but also the remarkable ICL capabilities

Table 1 Summary of architectures and deployment methods

Architecture	Mamba	[18, 19]
	Retnet	[20]
	Mixture of Experts (MoE)	Switch Transformer [21]
		Mixtral [22]
Accelerate	Sparsity	StreamingLLM [23]
		LM-Infinite [24]
		H2O [25]
		LESS [26]
	Allocation	PagedAttention [27]
	Sharing	MQA [28]
		GQA [29]
		YOCO [30]
		CLA [31]
	FlashAttention	[32–34]
Quantization	Quantization-aware training (QAT)	LLM-QAT [35]
		PEQA [36]
		QLoRA [37]
	Post-training quantization (PTQ)	GPTQ [38]
		LLM.int8() [39]
		SmoothQuant [40]
		SpQR [41]
		AWQ [42]
		ZeroQuant [43–46]

of LLMs. Today, the causal decoder remains the dominant architecture for many state-of-the-art LLMs, including GPT-4, LLaMA 3, and DeepSeek V3, among others. In addition to causal decoders, GLM [49] proposes prefix decoder that allows bidirectional attention over the prefix portion of a sequence while maintaining unidirectional attention for the generated portion. This hybrid approach enables the model to encode the prefix sequence bidirectionally while autoregressively predicting output tokens, with shared parameters for both encoding and decoding.

To achieve efficient training and inference for large-scale models, several solutions have emerged. One approach is to directly modify the model architecture. For instance, architectures like Mamba, based on state space models (SSMs) [18, 19], and RetNet [20] offer structural innovations to improve efficiency. Another widely adopted strategy is the Mixture of Experts (MoE), where only a subset of the neural networks weights is activated during training and inference, reducing computational overhead. Examples include Switch Transformer [21], Mixtral [22], and many state-of-the-art models. Research has also demonstrated that increasing the number of experts or the total parameter size can lead to notable performance improvements while maintaining efficiency [50].

2.2 Acceleration

Beyond architectural modifications, inference speed can be significantly improved by optimizing GPU memory access strategies. When computing attention, the key-value (KV) pairs for all tokens—except the last one—have already been processed in previous iterations. To leverage this, we can pre-cache these computed KV values for the next step, reducing redundant computations. However, as sequence length increases, the KV cache consumes a substantial amount of GPU memory, making efficient memory management a critical challenge. Given the limited GPU memory available, KV caching plays a crucial role in extending the models context window while minimizing GPU memory consumption. Consequently, KV cache optimization becomes essential in practical applications. Broadly, KV cache optimization can be categorized into three approaches: sparsity, allocation, and sharing.

Sparsity: Sparsity-based methods leverage the observation that only a small subset of tokens receives significant attention during computation. StreamingLLM [23] and LM-Infinite [24] found that initial tokens in a sequence tend to receive disproportionate attention. To exploit this, StreamingLLM retains only the KV pairs of the most recent sliding window, while LM-Infinite truncates KV sequences beyond the pre-training length. Another approach, Heavy Hitter Oracle (H2O) [25], dynamically balances between retaining recent tokens and preserving important historical tokens using a greedy algorithm. A more advanced method, LESS (Low-rank Embedding Sidekick with Sparse policy) [26], improves on sparsity-based KV caching by learning residuals between the original attention output and the sparsified attention output. This allows the model to recover omitted regions in the attention graph, ensuring that critical information is not lost despite aggressive KV compression.

Allocation: Inspired by paging mechanisms in operating systems, PagedAttention [27] optimizes KV caching by organizing KV pairs into fixed-size blocks. Each block contains the keys and values for a specific number of tokens. When computing attention, PagedAttention treats these blocks like virtual memory pages, mapping contiguous logical blocks to non-contiguous physical blocks via a block table. This on-demand allocation of KV blocks allows for more flexible memory management, significantly improving inference efficiency.

Sharing: KV cache sharing is an optimization technique in multi-head attention where different attention heads share a common set of KV pairs, significantly reducing GPU memory consumption. This approach can be categorized into intra-layer sharing and inter-layer sharing. Intra-layer sharing primarily includes Multi-Query Attention (MQA) [28] and Group-Query Attention (GQA) [29], where MQA maintains a single KV cache for all attention heads, while GQA groups attention heads, with each group sharing a dedicated KV cache. Inter-layer sharing methods such as YOCO [30] and Cross-Layer Attention (CLA) [31] further optimize KV cache usage. YOCO divides the decoder into a self-decoder, responsible for generating the KV cache, and a cross-decoder, which implements cross-attention without generating additional KV cache, thereby reducing redundancy. CLA, unlike YOCO, does not limit

KV cache generation to the top L/2 layers but instead alternates KV cache generation across different layers, ensuring efficient reuse of KV pairs between similar layers and optimizing both memory usage and computational efficiency.

FlashAttention: In addition to KV cache optimization, another important speedup direction is based on GPU memory architecture optimization. FlashAttention [32–34] optimizes computation and reduces GPU memory consumption by restructuring the attention computation process. By reordering the attention computation and leveraging tiling and recomputation techniques, it effectively minimizes data movement to and from High Bandwidth Memory (HBM). Unlike traditional methods that store large intermediate attention matrices in memory, FlashAttention avoids these costly writes, leading to substantial efficiency gains. This strategic reduction in memory reads and writes not only lowers the computational overhead but also achieves an impressive 2-4x speedup, transforming the scalability of attention-based models, particularly in large-scale applications.

Techniques like KV caching and FlashAttention complement each other to enhance the efficiency of large language models (LLMs). KV caching reduces redundant computations by storing key and value pairs from previous tokens, while FlashAttention optimizes memory usage and speeds up attention computation. Together, these methods enable LLMs to scale effectively while maintaining feasibility for deployment in complex industrial applications. By improving both computational speed and memory efficiency, they help ensure that large-scale models can operate smoothly in real-world scenarios without excessive resource demands.

2.3 Quantization

Scaling laws [15, 16] indicate that increasing the number of parameters significantly enhances the performance of large language models (LLMs). However, in practical applications, fully leveraging this scaling potential is constrained by factors such as hardware limitations, memory bandwidth, and computational costs. Deploying LLMs efficiently requires optimization techniques to mitigate these challenges, with quantization being one of the most widely adopted approaches. Quantization reduces the storage and computational complexity of models by compressing their parameters, making deployment on resource-constrained hardware more feasible. The most common quantization technique involves converting floating-point parameters into lower-precision floating-point or integer representations. This transformation not only reduces the GPU memory requirement of the model but also accelerates model loading times, making inference more efficient. Moreover, since integer arithmetic is significantly faster than floating-point operations on modern GPUs and specialized AI accelerators, quantization can substantially improve inference speed while maintaining comparable model performance. For large-scale models, 8-bit quantization (FP8/INT8) and 4-bit quantization (FP4/NF4/INT4) are among the most commonly used precision levels. Research suggests that larger models exhibit greater resilience to precision reduction, with 4-bit quantization preserving performance when the model size exceeds 70 billion parameters. Consequently, for these massive models, 4-bit quantization offers an optimal trade-off between efficiency and performance. Conversely, for smaller models, 8-bit quantization often strikes a better balance, as they tend to suffer more from aggressive precision reductions.

Quantization techniques can be broadly classified into two categories: quantization-aware training (QAT) and post-training quantization (PTQ), based on when the quantization process is applied. QAT integrates quantization into the training process, allowing the model to adapt to lower precision during learning, thereby minimizing accuracy degradation. As LLM deployment continues to expand across various industries, quantization remains a crucial tool for balancing computational efficiency and model accuracy.

2.3.1 QAT

Quantization-aware training (QAT) [51] incorporates a weight transformation process directly into the training phase. While this approach typically preserves model performance more effectively than post-training quantization, it comes at the cost of significantly higher computational demands. QAT introduces quantization error during fine-tuning by first quantizing and then dequantizing the weights and activations. This allows the model to adapt to the effects of quantization during training, thereby reducing quantization-induced errors.

Despite its advantages in maintaining accuracy, QAT imposes substantial computational overhead, particularly for ultra-large-scale LLMs. To address this challenge, LLM-QAT [35] extends the quantization process beyond weights and activations to include KV caches. By applying quantization to both weights and KV caches, this method enables distillation of the LLM into a highly efficient 4-bit model, demonstrating the feasibility of low-bit quantization for large-scale language models.

Another approach, PEQA (Progressive Extreme Quantization and Adaptation) [36], divides the quantization and training process into two distinct phases. In the first phase, it quantizes the fully connected layer weights, significantly reducing model size. In the second phase, it fine-tunes the model for specific downstream tasks, ensuring adaptation to various applications. This two-stage strategy achieves both rapid fine-tuning and flexible task switching.

To further optimize quantization efficiency, QLoRA (Quantized LoRA) [37] introduces several key innovations, including the NF4 (Normalized Float 4) data type,

double quantization, and a paging optimizer. These enhancements help mitigate performance loss while significantly reducing GPU Memory usage. NF4, in particular, enables better representation of weight distributions, while double quantization compresses the quantization metadata itself, further reducing the GPU memory usage.

2.3.2 PTQ

Unlike QAT, post-training quantization (PTQ) does not require additional training. Instead, it directly converts the weights of a pre-trained model to a lower precision, making it a simpler and more practical approach for deploying LLMs efficiently. However, PTQ may introduce slight performance degradation due to the loss of precision in weight values.

In PTQ, weight quantization is relatively straightforward since quantization coefficients can be precomputed before inference. Activation quantization, however, is more complex because activation values depend on the specific input during inference. Based on when activations are quantized, PTQ can be categorized into dynamic and static quantization. Dynamic quantization calculates the quantization coefficients for activations in real time during inference, allowing for adaptive precision adjustments. In contrast, static quantization precomputes the activation quantization coefficients before inference, applying them directly during model execution.

Since weights are easier to quantize than activations, GPTQ [38] adopts a weight-only quantization W4A16 based on OBQ [52]. In this approach, weights are quantized to int4, while activations remain in float16. During inference, the model dynamically dequantizes the weights back to float16, ensuring that actual computations maintain numerical stability and precision. LLM.int8() [39] employs mixed-precision decomposition, where the majority of weights and activations are quantized to 8-bit precision, while a small subset of outlier features is retained at 16-bit precision. This selective higher-precision retention helps mitigate accuracy loss by preserving critical outlier information. SmoothQuant [40] introduces an innovative approach to activation quantization by incorporating a smoothing factor s. This factor helps to smooth activation outliers, effectively shifting the quantization from activations to weights through a mathematically equivalent transformation.

These quantization techniques aim to identify optimal quantized parameters that minimize the discrepancy between the model's outputs before and after quantization. However, a key challenge in quantizing LLMs is the presence of significant outlier features in both input activations and output representations. SpQR (Sparse Quantization with Reconstruction) [41], employs hybrid sparse quantization, which selectively identifies and isolates outlier weights, storing them at higher precision. Meanwhile, the remaining weights are aggressively compressed to 3-4 bits, enabling efficient quantization with less than 1% precision loss. This method strikes a balance between compression and model fidelity. AWQ (Activation-Weighted Quantization) [42] enhances quantization performance by prioritizing weights based on activation magnitudes. By retaining only 0.1% to 1% of the weights corresponding to larger activations, AWQ significantly improves quantized model accuracy, achieving performance levels comparable to reconstruction-based GPTQ. This selective preservation of critical weights mitigates the impact of quantization on model performance.

The aforementioned quantization methods mainly focus on weight quantization, while activation quantization remains a more challenging problem. Activation outliers can be up to 100 times larger than most activation values, and directly quantizing them can lead to significant information loss. Therefore, ZeroQuant [43–46] explores various approaches to achieve effective activation quantization, improving the practicality and efficiency of W8A8 quantization.

3 Fine-tuning

Despite the impressive performance of LLMs and their success in NLP tasks, several challenges persist in real-world applications, particularly in adapting these models to downstream tasks. A growing body of research suggests that while LLMs exhibit strong general capabilities, their effectiveness in specific applications often requires further adaptation to align with particular objectives. One widely adopted approach to address this challenge is instruction tuning [53], a fine-tuning technique that enhances the capabilities of pre-trained LLMs by training them on a curated set of task-specific examples formatted in natural language. Instruction tuning has been shown to significantly improve LLM's ability to generalize to previously unseen tasks, even in multilingual contexts [54]. Many state-of-the-art LLMs, such as InstructGPT [12] and GPT-4, leverage instruction tuning extensively to better align their outputs with user requirements. Achieving effective instruction-based fine-tuning typically involves transitioning the model's outputs from a general-purpose source domain to a more specialized target domain. This can be accomplished through full fine-tuning, where all model parameters are updated. However, this approach is computationally expensive and memory-intensive due to the vast number of parameters in LLMs. Additionally, many downstream tasks have limited datasets, making it difficult to fully optimize all parameters without overfitting or requiring prohibitively large computational resources. To address these issues, the industry has increasingly adopted Parameter Efficient Fine-Tuning (PEFT), which enable LLM adaptation with reduced computational overhead. Beyond fine-tuning challenges, foundational LLMs are primarily trained on large-scale textual

data from the internet. Extending their capabilities to multi-modal tasks—such as processing and generating images, audio, and video—necessitates a much higher computational cost compared to text-based models. Consequently, researchers are striving to develop multi-modal LLMs that integrate diverse data modalities, thereby expanding the applicability of LLMs beyond text-based tasks and unlocking new possibilities across various industrial domains.

3.1 PEFT

Parameter Efficient Fine-Tuning (PEFT) [55] is a technique designed to reduce the number of trainable parameters by fine-tuning only a subset of a model's parameters, thereby achieving performance comparable to full fine-tuning while significantly lowering computational costs. This approach is particularly beneficial for large-scale models, such as large language models (LLMs), where full fine-tuning would be expensive due to the sheer number of parameters. Rather than simply freezing certain parameters, this section explore various PEFT methods using the Transformer architecture as an example. including adapter tuning, prefix tuning, prompt tuning, and low-rank adaptation (LoRA) [56].

3.1.1 Adapter tuning

Adapter tuning [57] introduces lightweight neural network modules, known as adapters, into the transformer architecture. These modules are integrated into each transformer block, typically inserted serially after the attention and feedforward layers. Alternatively, parallel adapters can be employed [58], where two adapter modules are placed alongside the attention and feedforward layers rather than in sequence. The flexibility of adapter tuning allows different adapter architectures to be tailored to various downstream tasks.

The adapter module operates by first compressing the original feature vector into a lower-dimensional representation, followed by a non-linear transformation, and then expanding it back to the original dimension. During fine-tuning, only the parameters within the adapter module are updated, while the core parameters of the pre-trained LLM remain frozen. This selective tuning approach significantly reduces computational demands while allowing the model to adapt effectively to specific tasks. By isolating task-specific modifications within the adapter modules, adapter tuning not only improves efficiency but also enhances the modularity and reusability of fine-tuned models across different domains.

3.1.2 Prompt tuning

The discovery of in-context learning (ICL) has sparked significant interest in prompt-based techniques, leading to the development of prompt tuning [59, 60]—a method that enhances model adaptability by incorporating trainable prompt vectors. Unlike traditional fine-tuning, which modifies a large number of model parameters, prompt tuning focuses on optimizing only a small set of task-specific prompt embeddings, making it a highly efficient approach for adapting large language models (LLMs) to new tasks.

In discrete prompt tuning [61, 62], manually designed textual prompts are appended to the input text, guiding the model toward the desired task behavior. In contrast, soft prompt tuning introduces trainable prompt vectors into the input layer, allowing the model to learn optimal prompt representations dynamically. These trainable prompt embeddings are combined with input text embeddings and fed into the language model, enabling it to effectively solve specific downstream tasks. P-tuning [59] proposes a free-form combination of context, prompts, and target tokens, making it well-suited for both natural language understanding and generation tasks. Prompt tuning can be implemented in various ways, such as learning soft prompt token representations via bi-directional LSTM (Bi-LSTM) or prepending trainable prefix prompts directly to the input sequence [60]. Crucially, during training, only the prompt embeddings are optimized, while the model's core parameters remain unchanged. This makes prompt tuning an efficient and scalable technique for adapting LLMs to diverse applications with minimal computational overhead.

3.1.3 Prefix tuning

Unlike prompt tuning, which modifies only the input embeddings, prefix tuning [63] introduces a set of trainable prefix vectors at the input of each transformer block. These prefix vectors act as virtual token embeddings that are task-specific and help steer the model toward the desired task behavior without modifying its core parameters. By injecting these learnable prefix embeddings at multiple layers of the model, prefix tuning allows for deeper interaction with the model's internal representations, making it a more expressive fine-tuning approach compared to standard prompt tuning.

To optimize the prefix vectors efficiently, reparameterization techniques can be applied. For instance, Prefix-tuning [63] proposes learning a multi-layer perception (MLP) function that maps smaller matrices onto the prefix parameter matrices, rather than directly optimizing the prefixes. This reparameterization strategy has been shown to enhance training stability. Once optimization is complete, the mapping function is discarded, and only the final prefix vectors are retained, ensuring task-specific performance improvements without additional computational overhead during inference. P-Tuning v2 [64] removes the heavily parameterized encoder used in earlier versions and instead adopts variable-length prompts for different tasks. Additionally, it incorporates multi-task learning to

jointly optimize shared prompts, thereby improving efficiency and enhancing model performance across different parameter scales in natural language understanding tasks.

3.1.4 LoRA

Low-Rank Adaptation (LoRA) [65] leverages the low intrinsic rank of large models to efficiently fine-tune them for downstream tasks. Instead of updating all model parameters, LoRA approximates the parameter update matrices at each layer using low-rank constrained bypass matrices, significantly reducing the number of trainable parameters while maintaining model performance. One of the key advantages of LoRA is its ability to drastically reduce GPU memory consumption, making fine-tuning large language models (LLMs) more computationally efficient. Additionally, LoRA enables parameter sharing across multiple tasks—a single large pre-trained model can be maintained, while multiple task-specific low-rank decomposition matrices can be stored separately. This allows for rapid adaptation to different downstream tasks without requiring multiple copies of the full model, thereby improving both efficiency and flexibility. Building upon LoRA, researchers have explored rank-setting strategies to further optimize its effectiveness. AdaLoRA [66] automatically assigns a budget of fine-tunable parameters to each parameter based on its importance. DyLoRA [67] can dynamically adjust the rank during inference without retraining. LoRA+ [68] sets different learning rates for the fitness matrix and chooses a fixed rate. QLoRA [37] combines LoRA with quantization to further reduce GPU memory requirements.

3.2 Multimodal

A common approach to multimodal modeling is encoding each modality separately and optimizing the encoder using a dedicated objective function [69–71]. Building on this foundation, the encoded representations of multiple modalities can be fused and subsequently decoded by an additional fusion decoder to accomplish the target task [72–75]. With the rapid advancement of textual modality performance, an alternative strategy has emerged—directly feeding both the encoded outputs of other modalities and raw textual inputs into a language model [76–78]. This approach enables the multi-modal extension of textual models, leveraging the strengths of natural language processing to integrate diverse data sources seamlessly. However, training multi-modal models from scratch presents a significant computational challenge, as the scale of both models and datasets continues to expand. To mitigate these computational costs, LLMs have been leveraged to enhance the efficiency of multimodal pretraining, giving rise to a new research direction: Multimodal Large Language Models (MM-LLMs). These models incorporate encoders that translate different modalities into representations compatible with LLMs, allowing the powerful reasoning and generation capabilities of LLMs to be extended across multiple modalities. The key challenge in developing MM-LLMs lies in effectively bridging the gap between LLMs and modality-specific models to enable seamless collaborative reasoning.

Recent breakthroughs, such as GPT-4o and Gemini, have demonstrated impressive multi-modal understanding and generation capabilities, sparking a surge of research interest in MM-LLMs. Initial investigations primarily focused on multi-modal content understanding and text generation, leading to the development of models such as BLIP-2 [79], LLaVA [80], and MiniGPT-4 [81] for image-text understanding; LLaMA-VID [82] for video-text comprehension; and QwenAudio [83] for audio-text interactions. As the field evolved, MM-LLMs were extended to support modality-specific outputs, broadening their application scope. Notable advancements include GILL [84], Emu [85], and MiniGPT-5 [86] for image-text generation, as well as SpeechGPT [87] for speech-to-text and audio-text transformations.

The general architecture of MM-LLMs is composed of three key components: modal encoding and fusion, the LLM backbone, and decoding generation. During training, the parameters of both the multi-modal encoders and the LLM itself are typically frozen, with optimization primarily focused on refining modal fusion and generation techniques. Since the number of trainable parameters in the modal fusion stage is relatively small, the proportion of parameters requiring optimization in an MM-LLM is significantly lower than the total model size—typically around 2%. The overall number of parameters in an MM-LLM is largely determined by the size of the core LLM used. This design enables efficient training for various multi-modal tasks, allowing MM-LLMs to adapt flexibly to different applications without excessive computational costs.

A wide range of modality-specific encoders exists to process different types of input data. For image encoding, commonly used architectures include MAE [88], ViT [89], CLIP [69], and Swin. In the case of video processing, a standard approach involves uniformly sampling a small number of frames (e.g., five frames) and applying the same preprocessing techniques as images. Audio and time-series data are typically encoded using models such as HuBERT [90], BEATs [91], Whisper [92], and CLAP [93]. Given the challenge of managing multiple heterogeneous encoders, some MM-LLMs adopt ImageBind [94], a unified encoder capable of handling six diverse modalities: image/video, text, audio, heatmap, acceleration, and depth.

Modal fusion plays a crucial role in integrating information across different modalities. A simple approach involves using a feed-forward network or an MLP to merge multi-modal representations. However, more advanced fusion techniques have been developed to enhance integration, such as cross-attention [95], where trainable query

vectors compress the feature sequence into a fixed-length representation. This compressed multi-modal representation can either be directly fed into the LLM or further refined through additional cross-attention fusion layers, enabling more effective cross-modal reasoning and seamless multi-modal interactions.

Recent research advancements have increasingly focused on achieving flexible transformations between arbitrary modalities, and [96] gives a more detailed summary. HuggingGPT [97] and AudioGPT [98] integrate LLMs with external tools to facilitate near-arbitrary modality understanding and generation, enabling the dynamic processing of diverse multi-modal inputs. However, to minimize propagation errors inherent in cascaded systems, researchers have developed end-to-end MM-LLMs capable of directly handling arbitrary modalities, including NExT-GPT [99], CoDi-2 [100], ModaVerse [101], and GPT-4o, which push toward generalized multi-modal intelligence.

4 Application

The emergence of Industry 4.0 has significantly accelerated the evolution of Prognostics and Health Management (PHM). At the forefront of this revolution is LLMs, a cutting-edge technology that seamlessly integrates with and enhances the capabilities of PHM. As LLMs continue to evolve, they foster mutual development with various industrial systems, enabling new breakthroughs. This chapter delves into the diverse applications of LLMs in the PHM, providing an in-depth analysis of their current impact and future prospects. Specifically, it examines how LLMs are being applied in three key areas: anomaly detection, fault analysis, and question answering. These domains reflect the pivotal directions of image processing, time-series data analysis, and human-computer interaction in PHM, offering a comprehensive view of the vast potential of LLMs in shaping the future of PHM.

4.1 Anomaly detection

Anomaly detection refers to the identification of data points that significantly deviate from the norm, signaling potential irregularities or critical insights within a dataset. This field integrates a diverse array of research methodologies, including data mining, machine learning, computer vision, and statistical learning, making it a cornerstone of numerous industries such as finance, healthcare, cybersecurity, and manufacturing. In recent years, deep learning has excelled in learning expressive representations from complex, multimodal data. The ability of deep neural networks to automatically extract meaningful features and assign anomaly scores has led to significant improvements over traditional detection methods, offering enhanced accuracy, adaptability, and robustness across various real-world applications.

Deep anomaly detection is generally divided into two key stages: feature extraction and anomaly classification. The feature extraction phase leverages deep learning networks to transform the original data space into latent representation, capturing richer semantic information and complex non-linear relationships between features. In the domain of image anomaly detection, features of normal samples can be directly extracted using general pre-trained deep learning models such as VGG [102] and ResNet [103]. Alternatively, a more tailored approach involves mapping pre-trained feature spaces to domain-specific representations, optimizing the extraction process for specific anomaly detection tasks [104].

Once the data features have been successfully extracted, anomaly classification can be performed using unsupervised machine learning techniques such as clustering, which can identify deviations in the feature space. Alternatively, a deep learning model can be trained to directly learn feature representations tailored to anomaly detection. For time-series anomaly detection, which is commonly applied to vibration signals, video streams, and sensor data, anomalies often manifest as deviations in sequential patterns. In such cases, recursive time-point prediction can be employed, where a model predicts the next value in the sequence based on historical data. Anomalies are then identified by calculating the error between the predicted and actual values.

Since anomalous data is typically sparse and difficult to annotate, anomaly detection often relies on semi-supervised learning approaches, particularly reconstruction-based methods. These methods involve training a model on a large of normal data, learning its underlying distribution, and then measuring reconstruction errors at inference time. If a given instance deviates significantly from its reconstructed counterpart, it is flagged as an anomaly. Common deep learning architectures for reconstruction-based anomaly detection include autoencoder (AE), generative adversarial network (GAN) [105], diffusion model [106] and their variants [107]. The choice of reconstruction method depends on the specific task and the nature of the dataset. A detailed discussion of these techniques, along with representative research findings, can be found in [108].

As LLMs continue to advance in performance and capability, researchers have begun integrating them with anomaly detection to enhance accuracy and efficiency. In the image domain, AnomalyGPT [109] leverages LLMs by generating synthetic training data, simulating anomalous images, and producing corresponding textual descriptions. To further refine the model's understanding, an image decoder is employed to provide fine-grained semantic information, while a prompt learner fine-tunes the large vision-language model (LVLM) using prompt embeddings. [110] introduces an innovative approach where a vision-language model (VLM) generates textual descriptions for each frame of a test video. These textual scene descriptions are then utilized to design a cueing mechanism

that activates the LLM's anomaly detection capabilities, effectively transforming it into a powerful video anomaly detector.

Similarly, in log based anomaly detection, LLMs demonstrate remarkable proficiency in understanding and interpreting the contextual nuances of log entries. This enables a more detailed and efficient anomaly detection process, significantly reducing the time and resources traditionally required for manual log reviews [111]. Furthermore, in IoT anomaly detection, [112] highlights that the command information in attack traffic is human-readable. By treating IoT-generated data as a "special language", the study utilizes LLM embeddings to extract traffic load characteristics, facilitating more effective and scalable anomaly detection in IoT environments.

4.2 Fault diagnosis

A critical step following anomaly detection is fault diagnosis, which involves identifying the root causes of anomalies based on observed abnormal phenomena. In PHM, fault analysis is typically conducted by examining process data recorded from production systems or equipment. As a result, time-series data analysis plays a crucial role in fault diagnosis. Similar to anomaly detection, fault diagnosis methods begin by extracting fault-related features from data and subsequently classifying these features to determine the underlying issue. Commonly used feature extraction networks include AE [113], CNN [114], RNN [115] and their variants. One-dimensional CNNs are particularly effective for analyzing time-series data, while two-dimensional CNNs are well-suited for handling multi-dimensional data with spatial and temporal correlations. RNNs, on the other hand, are designed to process sequential inputs using feedback loops, enabling them to retain past states—an essential capability for analyzing the temporal characteristics of mechanical signals such as vibration and temperature variations. Unlike anomaly detection, which often relies on unsupervised learning, fault diagnosis can be approached using both supervised techniques, such as transfer learning, and unsupervised methods, such as reinforcement learning [116].

Beyond time-series analysis, Artificial Intelligence for IT Operations (AIOps) is gaining significant traction in industrial applications. AI-driven techniques are increasingly used to automate various stages of the event lifecycle, including event prioritization, identifying recurring issues, and accelerating resolution times. Similar to its role in anomaly detection, Large Language Models (LLMs) excel in natural language processing, making them valuable for predicting and analyzing fault categories based on log data. [117] has shown that general-purpose LLMs, such as BERT [118], struggle with processing log data effectively. This is primarily due to differences in vocabulary, structure, and semantic patterns between log data and natural language. To address this, the authors pre-trained BERTOps using both public and proprietary log datasets, developing representations specifically optimized for log analysis tasks. The model was evaluated across three key downstream tasks: log format detection, golden signal classification, and fault category prediction. Results demonstrated that domain-specific LLMs, fine-tuned on log data, exhibited a deeper understanding of specialized terminology and log structures, ultimately leading to more accurate and reliable fault diagnosis. [119] highlights the potential of GPT-3.5 in cloud computing incident management, demonstrating its superiority over traditional methods in root cause analysis and mitigation recommendations. Since machine-detected incidents often follow recognizable patterns, LLMs are particularly effective in identifying their root causes and suggesting appropriate mitigation strategies. In some cases, LLMs can even outperform human analysts in fault diagnosing due to their ability to recognize recurring patterns at scale. Similarly, [120] emphasizes the challenges of accurately pinpointing the root cause of incidents using a single data source. The study suggests that integrating multiple diagnostic data sources such as logs, metrics, and trace data significantly improves the accuracy and quality of root cause analysis. Additionally, past incidents with similar causes tend to recur within short time frames, meaning that leveraging historical data can accelerate the troubleshooting process. While human operators may struggle with analyzing previously unseen issues, LLM combined with CoT can enhance understanding and facilitate more effective problem resolution.

4.3 Question answering (QA)

The need for fast and accurate access to information is the main reason for the rise of research into question and answer systems (QA) as the first step in the application of LLM in various sectors, particularly industry. The primary driver behind its rise is the growing demand for fast and accurate information retrieval. Before the emergence of LLMs, research efforts primarily focused on improving the interaction layer, leading to notable advancements. Among these, BERT demonstrated the potential of the Transformer architecture, significantly enhancing QA performance. However, with the advent and outstanding capabilities of LLMs, QA as a downstream task has gained even greater attention.

Despite their impressive performance, LLMs still struggle with answering questions correctly due to several limitations. The reasons for these inaccuracies can be categorized into five key factors [121]:

Domain knowledge deficit: LLMs may lack deep expertise in specialized fields that were not well represented in their training data.

Outdated Information: The knowledge of an LLM is restricted to the information available in its pre-training

dataset, making it unable to answer questions about events or developments that occurred afterward.

Immemorization: Some knowledge, particularly low-frequency information, may not be fully retained by the model, leading to gaps in its responses [122].

Forgetting: Fine-tuning can lead to catastrophic forgetting, where previously learned knowledge is lost or overwritten [123].

Reasoning Failure: Even when an LLM has learned relevant knowledge, it may struggle to apply it correctly in complex reasoning tasks, particularly in multi-step reasoning processes [124].

To address these challenges, ICL enables a shift in question answering (QA) systems from traditional approaches to prompt engineering. [125] compares the effectiveness of fine-tuning and retrieval-augmented generation (RAG) [126] for injecting new knowledge into language models. The results indicate that RAG significantly outperforms fine-tuning in adapting to new information, making it a more effective approach for knowledge enhancement.

Retrieval-Augmented Generation (RAG) is a technique in which a language model dynamically retrieves external information during the generation process to enhance its natural language generation. Instead of relying solely on pre-trained knowledge, RAG actively retrieves relevant information and integrates it into the model's prompts, allowing it to generate more accurate and contextually appropriate responses. The core components of RAG are retrieval and generation: Retrieval leverages the efficient storage and search capabilities of vector databases to locate relevant knowledge. Generation utilizes the LLM and prompt engineering to effectively incorporate the retrieved information, ensuring a more precise and informed response. The detailed technical roadmap of RAG and recent advancements in the field are explored [127, 128], providing insights into the latest research on improving retrieval efficiency and integrating external knowledge sources into LLMs.

5 Conclusion and future

This review has examined the integration of Large Language Models (LLMs) in PHM, highlighting their transformative potential across various domains of image processing, time-series data analysis, and human-computer interaction. Throughout our analysis, we have demonstrated how architectural considerations, optimization techniques, and fine-tuning approaches collectively enable the practical deployment of these powerful models in resource-constrained industrial environments.

The evolution of LLM architecture and deployment strategies (Sect. 2) has been critical in overcoming computational barriers. Acceleration methods and quantization techniques—particularly FlashAttention (Sect. 2.2) and PTQ (Sect. 2.3.2)—have made it possible to run sophisticated language models on edge devices and industrial controllers without sacrificing essential performance characteristics. Parameter-efficient fine-tuning (Sect. 3.1), especially LORA (Sect. 3.1.4), has democratized the adaptation of foundation models to specialized industrial tasks while minimizing computational overhead.

The expansion into multi-modal capabilities (Sect. 3.2) represents a particularly significant advancement for PHM, as it enables LLMs to process and reason across diverse data types including time-series data, images, and textual documentation. This cross-modal integration allows for more robust anomaly detection (Sect. 4.1), comprehensive fault diagnosis (Sect. 4.2), and contextually aware question-answering systems (Sect. 4.3) that can interpret multiple input formats simultaneously.

However, several critical challenges and research gaps remain:

- **Reliability and Safety:** The deployment of LLMs in mission-critical settings demands rigorous validation to ensure deterministic and safe behavior, especially when model outputs influence physical processes.
- **Interpretability:** Many LLMs function as black boxes, hindering trust and adoption in industrial settings. Research into more transparent and explainable model behavior is urgently needed.
- **Data Privacy and Security:** With access to sensitive operational data, LLMs must be designed with robust privacy-preserving mechanisms, including federated learning and differential privacy.
- **Continual and Lifelong Learning:** Industrial environments are dynamic. LLMs must be capable of continuous learning without catastrophic forgetting, which remains an open challenge.
- **Energy Efficiency and Sustainability:** Despite progress through quantization, the high energy demands of LLMs still pose challenges for sustainable deployment, particularly on edge devices.

To accelerate the safe and effective integration of LLMs into PHM, the following research avenues are proposed:

- **Development of Lightweight, Interpretable Models:** Focus on hybrid architectures that combine symbolic reasoning with LLMs to improve transparency and reduce computational load.
- **Federated and Edge Learning Frameworks:** Advance privacy-preserving strategies that enable model training and inference directly at the edge, reducing data transfer and latency.
- **Task-Specific Multi-modal Pretraining:** Create domain-adapted multi-modal foundation models tailored for PHM tasks, incorporating structured sensor data, images, and technical documents.
- **Robust Benchmarking and Simulation Environments:** Establish standardized benchmarks

and high-fidelity simulators for evaluating LLM performance in industrial PHM scenarios.
- **Energy-Aware Model Optimization:** Explore adaptive inference strategies and hardware-software co-design to minimize energy consumption without compromising accuracy.

In conclusion, LLMs have emerged as powerful tools for PHM, capable of transforming how knowledge is accessed, processes are monitored, and decisions are made. While technical and practical challenges persist, the rapid pace of innovation in this field suggests that language models will become increasingly integral components of industrial systems, contributing to enhanced efficiency, reduced downtime, and improved decision-making across manufacturing sectors. As research continues to address current limitations, LLMs stand poised to become fundamental building blocks of the intelligent industrial systems of tomorrow.

Acknowledgements
The authors would like to express their gratitude to all the teachers and fellow lab students who provided assistance.

Author contributions
Tingyi Yu: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization. Junya Tang: Investigation, Resources, Writing -Review & Editing, Supervision. Qingyun Yu: Writing - Review & Editing, Supervision, Project Administration. Li Li: Conceptualization, Writing - Review & Editing, Supervision, Project Administration, Funding Acquisition. Liu Ying: Conceptualization, Writing - Review & Editing, Supervision. Raul Poler: Conceptualization, Writing - Review & Editing, Supervision. All authors read and approved the final manuscript.

Funding information
This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 72171172 and 92367101; the Aeronautical Science Foundation of China under Grant No. 2023Z066038001; the National Natural Science Foundation of China Basic Science Research Center Program under Grant No. 62088101; Shanghai Municipal Science and Technology Major Project under Grant No. 2021SHZDZX0100; Chinese Academy of Engineering, Strategic Research and Consulting Program, under Grant No. 2023-XZ-65.

Data availability
No data was used for the research described in the article.

Declarations

Competing interests
The authors declare no conflicts of interest.

Author details
[1]College of Electronic and Information Engineering, Tongji University, Shanghai, 201804, China. [2]School of Engineering, Cardiff University, Cardiff, United Kingdom. [3]Escuela Politécnica Superior de Alcoy, Research Centre on Production Management and Engineering, Universitat Politècnica de València, Calle Alarcón, Alcoy (Alicante), 03801, Spain.

Received: 25 March 2025 Revised: 10 May 2025 Accepted: 15 May 2025
Published online: 19 August 2025

References

1. R. Yan, R.X. Gao, X. Chen, Wavelets for fault diagnosis of rotary machines: a review with applications. Signal Process. **96**, 1–15 (2014). https://doi.org/10.1016/j.sigpro.2013.04.015
2. Y. Lei, J. Lin, Z. He, M.J. Zuo, A review on empirical mode decomposition in fault diagnosis of rotating machinery. Mech. Syst. Signal Process. **35**(1), 108–126 (2013). https://doi.org/10.1016/j.ymssp.2012.09.015
3. A. Widodo, B.-S. Yang, Support vector machine in machine condition monitoring and fault diagnosis. Mech. Syst. Signal Process. **21**(6), 2560–2574 (2007). https://doi.org/10.1016/j.ymssp.2006.12.007
4. L.M.R. Baccarini, V.V. Rocha e Silva, B.R. de Menezes, W.M. Caminhas, SVM practical industrial application for mechanical faults diagnostic. Expert Syst. Appl. **38**(6), 6980–6984 (2011). https://doi.org/10.1016/j.eswa.2010.12.017
5. Q.P. He, J. Wang, Fault detection using the k-nearest neighbor rule for semiconductor manufacturing processes. IEEE Trans. Semicond. Manuf. **20**(4), 345–354 (2007). https://doi.org/10.1109/TSM.2007.907607
6. Z. Zhou, C. Wen, C. Yang, Fault isolation based on k-nearest neighbor rule for industrial processes. IEEE Trans. Ind. Electron. **63**(4), 2578–2586 (2016). https://doi.org/10.1109/TIE.2016.2520898
7. Y. LeCun, Y. Bengio, G. Hinton, Deep learning. Nature **521**(7553), 436–444 (2015). https://doi.org/10.1038/nature14539
8. K. Berahmand, F. Daneshfar, E.S. Salehi, Y. Li, Y. Xu, Autoencoders and their applications in machine learning: a survey. Artif. Intell. Rev. **57**(2), 28 (2024). https://doi.org/10.1007/s10462-023-10662-6
9. Z. Li, F. Liu, W. Yang, S. Peng, J. Zhou, A survey of convolutional neural networks: analysis, applications, and prospects. IEEE Trans. Neural Netw. Learn. Syst. **33**(12), 6999–7019 (2022). https://doi.org/10.1109/TNNLS.2021.3084827
10. M. Kaur, A. Mohta, A review of deep learning with recurrent neural network, in *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)* (2019), pp. 460–465. https://doi.org/10.1109/ICSSIT46314.2019.8987837
11. R.K. Inapakurthi, S.S. Miriyala, K. Mitra, Recurrent neural networks based modelling of industrial grinding operation. Chem. Eng. Sci. **219**, 115585 (2020). https://doi.org/10.1016/j.ces.2020.115585
12. L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P.F. Christiano, J. Leike, R. Lowe, Training language models to follow instructions with human feedback. Adv. Neural Inf. Process. Syst. **35**, 27730–27744 (2022)
13. T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal, et al., Language models are few-shot learners, in *Advances in Neural Information Processing Systems*, vol. 33 (Curran Associates, Red Hook, 2020), pp. 1877–1901
14. A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, et al., PaLM: scaling language modeling with pathways. J. Mach. Learn. Res. **24**(240), 1–113 (2023)
15. J. Kaplan, S. McCandlish, T. Henighan, T.B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, D. Amodei, Scaling Laws for Neural Language Models (2020)
16. J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D.d.L. Casas, L.A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J.W. Rae, O. Vinyals, L. Sifre, *Training Compute-Optimal Large Language Models* (2022). https://doi.org/10.48550/arXiv.2203.15556
17. J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E.H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, W. Fedus, Emergent Abilities of Large Language Models (2022). https://doi.org/10.48550/arXiv.2206.07682
18. A. Gu, T. Dao, Mamba: Linear-Time Sequence Modeling with Selective State Spaces (2023). https://doi.org/10.48550/arXiv.2312.00752
19. T. Dao, A. Gu, Transformers Are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality (2024)
20. Y. Sun, L. Dong, S. Huang, S. Ma, Y. Xia, J. Xue, J. Wang, F. Wei, Retentive Network: a Successor to Transformer for Large Language Models (2023). https://doi.org/10.48550/arXiv.2307.08621
21. W. Fedus, B. Zoph, N. Shazeer, Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. J. Mach. Learn. Res. **23**(120), 1–39 (2022)
22. A.Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D.S. Chaplot, D. Casas, E.B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L.R. Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T.L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, W.E. Sayed, Mixtral of Experts (2024). https://doi.org/10.48550/arXiv.2401.04088

23. G. Xiao, Y. Tian, B. Chen, S. Han, M. Lewis, Efficient Streaming Language Models with Attention Sinks (2024)
24. C. Han, Q. Wang, H. Peng, W. Xiong, Y. Chen, H. Ji, S. Wang, LM-Infinite: Zero-Shot Extreme Length Generalization for Large Language Models (2024). https://doi.org/10.48550/arXiv.2308.16137
25. Z. Zhang, Y. Sheng, T. Zhou, T. Chen, L. Zheng, R. Cai, Z. Song, Y. Tian, C. Ré, C. Barrett, Z. Wang, B. Chen, H2O: heavy-hitter oracle for efficient generative inference of large language models, in *Proceedings of the 37th International Conference on Neural Information Processing Systems. NIPS'23* (Curran Associates, Red Hook, 2023), pp. 34661–34710
26. H. Dong, X. Yang, Z. Zhang, Z. Wang, Y. Chi, B. Chen, Get more with LESS: synthesizing recurrence with KV cache compression for efficient LLM inference, in *Forty-First International Conference on Machine Learning* (2024)
27. W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C.H. Yu, J. Gonzalez, H. Zhang, I. Stoica, Efficient memory management for large language model serving with PagedAttention, in *Proceedings of the 29th Symposium on Operating Systems Principles. SOSP'23* (Association for Computing Machinery, New York, 2023), pp. 611–626. https://doi.org/10.1145/3600006.3613165
28. N. Shazeer, Fast Transformer Decoding: One Write-Head Is All You Need (2019). https://doi.org/10.48550/arXiv.1911.02150
29. J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebron, S. Sanghai, GQA: training generalized multi-query transformer models from multi-head checkpoints, in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, ed. by H. Bouamor, J. Pino, K. Bali (Association for Computational Linguistics, Singapore, 2023), pp. 4895–4901. https://doi.org/10.18653/v1/2023.emnlp-main.298
30. Y. Sun, L. Dong, Y. Zhu, S. Huang, W. Wang, S. Ma, Q. Zhang, J. Wang, F. Wei, You only cache once: decoder-decoder architectures for language models. Adv. Neural Inf. Process. Syst. **37**, 7339–7361 (2024)
31. W. Brandon, M. Mishra, A. Nrusimha, R. Panda, J. Ragan-Kelley, Reducing transformer key-value cache size with cross-layer attention, in *The Thirty-Eighth Annual Conference on Neural Information Processing Systems* (2024)
32. T. Dao, D. Fu, S. Ermon, A. Rudra, C. Ré, FlashAttention: fast and memory-efficient exact attention with IO-awareness. Adv. Neural Inf. Process. Syst. **35**, 16344–16359 (2022)
33. T. Dao, FlashAttention-2: faster attention with better parallelism and work partitioning, in *The Twelfth International Conference on Learning Representations* (2023)
34. J. Shah, G. Bikshandi, Y. Zhang, V. Thakkar, P. Ramani, T. Dao, FlashAttention-3: fast and accurate attention with asynchrony and low-precision. Adv. Neural Inf. Process. Syst. **37**, 68658–68685 (2024)
35. Z. Liu, B. Oguz, C. Zhao, E. Chang, P. Stock, Y. Mehdad, Y. Shi, R. Krishnamoorthi, V. Chandra, LLM-QAT: data-free quantization aware training for large language models, in *Findings of the Association for Computational Linguistics: ACL 2024*, ed. by L.-W. Ku, A. Martins, V. Srikumar (Association for Computational Linguistics, Bangkok, 2024), pp. 467–484. https://doi.org/10.18653/v1/2024.findings-acl.26
36. J. Kim, J.H. Lee, S. Kim, J. Park, K.M. Yoo, S.J. Kwon, D. Lee, Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization. Adv. Neural Inf. Process. Syst. **36**, 36187–36207 (2023)
37. T. Dettmers, A. Pagnoni, A. Holtzman, L. Zettlemoyer, QLoRA: efficient finetuning of quantized LLMs. Adv. Neural Inf. Process. Syst. **36**, 10088–10115 (2023)
38. E. Frantar, S. Ashkboos, T. Hoefler, D. Alistarh, GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers (2023). https://doi.org/10.48550/arXiv.2210.17323
39. T. Dettmers, M. Lewis, Y. Belkada, L. Zettlemoyer, GPT3.int8(): 8-bit matrix multiplication for transformers at scale. Adv. Neural Inf. Process. Syst. **35**, 30318–30332 (2022)
40. G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, S. Han, SmoothQuant: accurate and efficient post-training quantization for large language models, in *Proceedings of the 40th International Conference on Machine Learning* (PMLR, 2023), pp. 38087–38099
41. T. Dettmers, R.A. Svirschevski, V. Egiazarian, D. Kuznedelev, E. Frantar, S. Ashkboos, A. Borzunov, T. Hoefler, D. Alistarh, SpQR: a sparse-quantized representation for near-lossless LLM weight compression, in *The Twelfth International Conference on Learning Representations* (2023)
42. J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan, S. Han, AWQ: activation-aware weight quantization for on-device LLM compression and acceleration. Proc. Mach. Learn. Syst. **6**, 87–100 (2024)
43. Z. Yao, R. Yazdani Aminabadi, M. Zhang, X. Wu, C. Li, Y. He, ZeroQuant: efficient and affordable post-training quantization for large-scale transformers. Adv. Neural Inf. Process. Syst. **35**, 27168–27183 (2022)
44. Z. Yao, X. Wu, C. Li, S. Youn, Y. He, Exploring post-training quantization in LLMs from comprehensive study to low rank compensation, in *Proceedings of the AAAI Conference on Artificial Intelligence 38(17)* (2024), pp. 19377–19385. https://doi.org/10.1609/aaai.v38i17.29908
45. X. Wu, Z. Yao, Y. He, ZeroQuant-FP: a Leap Forward in LLMs Post-Training W4A8 Quantization Using Floating-Point Formats (2023). https://doi.org/10.48550/arXiv.2307.09782
46. Z. Yao, R.Y. Aminabadi, S. Youn, X. Wu, E. Zheng, Y. He, ZeroQuant-HERO: Hardware-Enhanced Robust Optimized Post-Training Quantization Framework for W8A8 Transformers (2023). https://doi.org/10.48550/arXiv.2310.17723
47. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, *Attention Is All You Need*. Advances in Neural Information Processing Systems, vol. 30 (Curran Associates, Red Hook, 2017)
48. A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, Improving Language Understanding by Generative Pre-Training (2018)
49. Team GLM, A. Zeng, B. Xu, B. Wang, C. Zhang, D. Yin, D. Zhang, D. Rojas, G. Feng, H. Zhao, H. Lai, H. Yu, H. Wang, J. Sun, J. Zhang, J. Cheng, J. Gui, J. Tang, J. Zhang, J. Sun, J. Li, L. Zhao, L. Wu, L. Zhong, M. Liu, M. Huang, P. Zhang, Q. Zheng, R. Lu, S. Duan, S. Zhang, S. Cao, S. Yang, W.L. Tam, W. Zhao, X. Liu, X. Xia, X. Zhang, X. Gu, X. Lv, X. Liu, X. Liu, X. Yang, X. Song, X. Zhang, Y. An, Y. Xu, Y. Niu, Y. Yang, Y. Li, Y. Bai, Y. Dong, Z. Qi, Z. Wang, Z. Yang, Z. Du, Z. Hou, Z. Wang, ChatGLM: a Family of Large Language Models from GLM-130B to GLM-4 All Tools (2024). https://doi.org/10.48550/arXiv.2406.12793
50. A. Clark, D.D.L. Casas, A. Guy, A. Mensch, M. Paganini, J. Hoffmann, B. Damoc, B. Hechtman, T. Cai, S. Borgeaud, G.B.V.D. Driessche, E. Rutherford, T. Hennigan, M.J. Johnson, A. Cassirer, C. Jones, E. Buchatskaya, D. Budden, L. Sifre, S. Osindero, O. Vinyals, M. Ranzato, J. Rae, E. Elsen, K. Kavukcuoglu, K. Simonyan, Unified scaling laws for routed language models, in *Proceedings of the 39th International Conference on Machine Learning* (PMLR, Baltimore, 2022), pp. 4057–4086
51. B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, D. Kalenichenko, Quantization and training of neural networks for efficient integer-arithmetic-only inference, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 2704–2713
52. E. Frantar, D. Alistarh, Optimal brain compression: a framework for accurate post-training quantization and pruning. Adv. Neural Inf. Process. Syst. **35**, 4475–4488 (2022)
53. J. Wei, M. Bosma, V. Zhao, K. Guu, A.W. Yu, B. Lester, N. Du, A.M. Dai, Q.V. Le, Finetuned language models are zero-shot learners, in *International Conference on Learning Representations* (2021)
54. N. Muennighoff, T. Wang, L. Sutawika, A. Roberts, S. Biderman, T. Le Scao, M.S. Bari, S. Shen, Z.X. Yong, H. Schoelkopf, X. Tang, D. Radev, A.F. Aji, K. Almubarak, S. Albanie, Z. Alyafeai, A. Webson, E. Raff, C. Raffel, Crosslingual generalization through multitask finetuning, in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ed. by A. Rogers, J. Boyd-Graber, N. Okazaki (Association for Computational Linguistics, Toronto, 2023), pp. 15991–16111. https://doi.org/10.18653/v1/2023.acl-long.891
55. N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q.D. Laroussilhe, A. Gesmundo, M. Attariyan, S. Gelly, Parameter-efficient transfer learning for NLP, in *Proceedings of the 36th International Conference on Machine Learning* (PMLR, 2019), pp. 2790–2799
56. N. Ding, Y. Qin, G. Yang, F. Wei, Z. Yang, Y. Su, S. Hu, Y. Chen, C.-M. Chan, W. Chen, J. Yi, W. Zhao, X. Wang, Z. Liu, H.-T. Zheng, J. Chen, Y. Liu, J. Tang, J. Li, M. Sun, Parameter-efficient fine-tuning of large-scale pre-trained language models. Nat. Mach. Intell. **5**(3), 220–235 (2023). https://doi.org/10.1038/s42256-023-00626-4
57. S.-A. Rebuffi, H. Bilen, A. Vedaldi, Learning Multiple Visual Domains with Residual Adapters (2017). https://doi.org/10.48550/arXiv.1705.08045
58. J. He, C. Zhou, X. Ma, T. Berg-Kirkpatrick, G. Neubig, Towards a unified view of parameter-efficient transfer learning, in *International Conference on Learning Representations* (2021)
59. X. Liu, Y. Zheng, Z. Du, M. Ding, Y. Qian, Z. Yang, J. Tang, GPT understands, too. AI Open **5**, 208–215 (2024). https://doi.org/10.1016/j.aiopen.2023.08.012

60. B. Lester, R. Al-Rfou, N. Constant, The power of scale for parameter-efficient prompt tuning, in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, ed. by M.-F. Moens, X. Huang, L. Specia, S.W.-T. Yih (Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021), pp. 3045–3059. https://doi.org/10.18653/v1/2021.emnlp-main.243
61. Z. Jiang, F.F. Xu, J. Araki, G. Neubig, How can we know what language models know? Trans. Assoc. Comput. Linguist. **8**, 423–438 (2020). https://doi.org/10.1162/tacl_a_00324
62. T. Shin, Y. Razeghi, Y. Logan IV, E. Wallace, S. Singh, AutoPrompt: eliciting knowledge from language models with automatically generated prompts, in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, ed. by B. Webber, T. Cohn, Y. He, Y. Liu (Association for Computational Linguistics, Online, 2020), pp. 4222–4235. https://doi.org/10.18653/v1/2020.emnlp-main.346
63. X.L. Li, P. Liang, Prefix-tuning: optimizing continuous prompts for generation, in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, ed. by C. Zong, F. Xia, W. Li, R. Navigli (Association for Computational Linguistics, Online, 2021), pp. 4582–4597. https://doi.org/10.18653/v1/2021.acl-long.353
64. X. Liu, K. Ji, Y. Fu, W. Tam, Z. Du, Z. Yang, J. Tang, P-tuning: prompt tuning can be comparable to fine-tuning across scales and tasks, in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, ed. by S. Muresan, P. Nakov, A. Villavicencio (Association for Computational Linguistics, Dublin, 2022), pp. 61–68. https://doi.org/10.18653/v1/2022.acl-short.8
65. E.J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, LoRA: low-rank adaptation of large language models, in *International Conference on Learning Representations* (2021)
66. Q. Zhang, M. Chen, A. Bukharin, P. He, Y. Cheng, W. Chen, T. Zhao, Adaptive budget allocation for parameter-efficient fine-tuning, in *The Eleventh International Conference on Learning Representations* (2022)
67. M. Valipour, M. Rezagholizadeh, I. Kobyzev, A. Ghodsi, DyLoRA: parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation, in *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, ed. by A. Vlachos, I. Augenstein (Association for Computational Linguistics, Dubrovnik, 2023), pp. 3274–3287. https://doi.org/10.18653/v1/2023.eacl-main.239
68. S. Hayou, N. Ghosh, B. Yu, LoRA+: Efficient Low Rank Adaptation of Large Models (2024)
69. A. Radford, J.W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever, Learning transferable visual models from natural language supervision, in *Proceedings of the 38th International Conference on Machine Learning* (PMLR, 2021), pp. 8748–8763
70. C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, T. Duerig, Scaling up visual and vision-language representation learning with noisy text supervision, in *Proceedings of the 38th International Conference on Machine Learning* (PMLR, 2021), pp. 4904–4916
71. L. Yuan, D. Chen, Y.-L. Chen, N. Codella, X. Dai, J. Gao, H. Hu, X. Huang, B. Li, C. Li, C. Liu, M. Liu, Z. Liu, Y. Lu, Y. Shi, L. Wang, J. Wang, B. Xiao, Z. Xiao, J. Yang, M. Zeng, L. Zhou, P. Zhang, Florence: a New Foundation Model for Computer Vision (2021). https://doi.org/10.48550/arXiv.2111.11432
72. L.H. Li, P. Zhang, H. Zhang, J. Yang, C. Li, Y. Zhong, L. Wang, L. Yuan, L. Zhang, J.-N. Hwang, K.-W. Chang, J. Gao, Grounded language-image pre-training, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 10965–10975
73. J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, Y. Wu, CoCa: Contrastive Captioners are Image-Text Foundation Models. Trans. Mach. Learn. Res. (2022)
74. A. Singh, R. Hu, V. Goswami, G. Couairon, W. Galuba, M. Rohrbach, D. Kiela, FLAVA: a foundational language and vision alignment model, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 15638–15650
75. A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A.C. Berg, W.-Y. Lo, P. Dollar, R. Girshick, Segment anything, in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2023), pp. 4015–4026
76. M. Tsimpoukelli, J.L. Menick, S. Cabi, S.M.A. Eslami, O. Vinyals, F. Hill, Multimodal few-shot learning with frozen language models, in *Advances in Neural Information Processing Systems*, vol. 34 (Curran Associates, Red Hook, 2021), pp. 200–212
77. Z. Wang, J. Yu, A.W. Yu, Z. Dai, Y. Tsvetkov, Y. Cao, SimVLM: simple visual language model pretraining with weak supervision, in *International Conference on Learning Representations* (2021)
78. X. Chen, X. Wang, S. Changpinyo, A.J. Piergiovanni, P. Padlewski, D. Salz, S. Goodman, A. Grycner, B. Mustafa, L. Beyer, A. Kolesnikov, J. Puigcerver, N. Ding, K. Rong, H. Akbari, G. Mishra, L. Xue, A.V. Thapliyal, J. Bradbury, W. Kuo, M. Seyedhosseini, C. Jia, B.K. Ayan, C.R. Ruiz, A.P. Steiner, A. Angelova, X. Zhai, N. Houlsby, R. Soricut, PaLI: a jointly-scaled multilingual language-image model, in *The Eleventh International Conference on Learning Representations* (2022)
79. J. Li, D. Li, S. Savarese, S. Hoi, BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models, in *Proceedings of the 40th International Conference on Machine Learning* (PMLR, 2023), pp. 19730–19742
80. H. Liu, C. Li, Q. Wu, Y.J. Lee, Visual instruction tuning. Adv. Neural Inf. Process. Syst. **36**, 34892–34916 (2023)
81. D. Zhu, J. Chen, X. Shen, X. Li, M. Elhoseiny, MiniGPT-4: enhancing vision-language understanding with advanced large language models, in *The Twelfth International Conference on Learning Representations* (2023)
82. Y. Li, C. Wang, J. Jia, LLaMA-VID: an image is worth 2 tokens in large language models, in *Computer Vision – ECCV 2024*, ed. by A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, G. Varol (Springer, Cham, 2024), pp. 323–340. https://doi.org/10.1007/978-3-031-72952-2_19
83. Y. Chu, J. Xu, X. Zhou, Q. Yang, S. Zhang, Z. Yan, C. Zhou, J. Zhou, Qwen-Audio: Advancing Universal Audio Understanding via Unified Large-Scale Audio-Language Models (2023). https://doi.org/10.48550/arXiv.2311.07919
84. J.Y. Koh, D. Fried, R.R. Salakhutdinov, Generating images with multimodal language models. Adv. Neural Inf. Process. Syst. **36**, 21487–21506 (2023)
85. Q. Sun, Q. Yu, Y. Cui, F. Zhang, X. Zhang, Y. Wang, H. Gao, J. Liu, T. Huang, X. Wang, Emu: generative pretraining in multimodality, in *The Twelfth International Conference on Learning Representations* (2023)
86. K. Zheng, X. He, X.E. Wang, MiniGPT-5: interleaved vision-and-language generation via generative vokens, in *ICLR* (2024)
87. D. Zhang, S. Li, X. Zhang, J. Zhan, P. Wang, Y. Zhou, X. Qiu, SpeechGPT: empowering large language models with intrinsic cross-modal conversational abilities, in *Findings of the Association for Computational Linguistics: EMNLP 2023*, ed. by H. Bouamor, J. Pino, K. Bali (Association for Computational Linguistics, Singapore, 2023), pp. 15757–15773. https://doi.org/10.18653/v1/2023.findings-emnlp.1055
88. K. He, X. Chen, S. Xie, Y. Li, P. Dollár, R. Girshick, Masked autoencoders are scalable vision learners, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 16000–16009
89. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: transformers for image recognition at scale, in *International Conference on Learning Representations* (2020)
90. W.-N. Hsu, B. Bolte, Y.-H.H. Tsai, K. Lakhotia, R. Salakhutdinov, A. Mohamed, HuBERT: self-supervised speech representation learning by masked prediction of hidden units. IEEE/ACM Trans. Audio Speech Lang. Process. **29**, 3451–3460 (2021). https://doi.org/10.1109/TASLP.2021.3122291
91. S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen, W. Che, X. Yu, F. Wei, BEATs: audio pre-training with acoustic tokenizers, in *Proceedings of the 40th International Conference on Machine Learning* (PMLR, 2023), pp. 5178–5193
92. A. Radford, J.W. Kim, T. Xu, G. Brockman, C. Mcleavey, I. Sutskever, Robust speech recognition via large-scale weak supervision, in *Proceedings of the 40th International Conference on Machine Learning* (PMLR, 2023), pp. 28492–28518
93. Y. Wu, K. Chen, T. Zhang, Y. Hui, T. Berg-Kirkpatrick, S. Dubnov, Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation, in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2023), pp. 1–5. https://doi.org/10.1109/ICASSP49357.2023.10095969
94. R. Girdhar, A. El-Nouby, Z. Liu, M. Singh, K.V. Alwala, A. Joulin, I. Misra, ImageBind: one embedding space to bind them all, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 15180–15190

95. J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J.L. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Bińkowski, R. Barreira, O. Vinyals, A. Zisserman, K. Simonyan, Flamingo: a visual language model for few-shot learning. Adv. Neural Inf. Process. Syst. **35**, 23716–23736 (2022)
96. D. Zhang, Y. Yu, J. Dong, C. Li, D. Su, C. Chu, D. Yu, MM-LLMs: recent advances in MultiModal large language models, in *Findings of the Association for Computational Linguistics: ACL 2024*, ed. by L.-W. Ku, A. Martins, V. Srikumar (Association for Computational Linguistics, Bangkok, 2024), pp. 12401–12430. https://doi.org/10.18653/v1/2024.findings-acl.738
97. Y. Shen, K. Song, X. Tan, D. Li, W. Lu, Y. Zhuang, HuggingGPT: solving AI tasks with ChatGPT and its friends in hugging face. Adv. Neural Inf. Process. Syst. **36**, 38154–38180 (2023)
98. R. Huang, M. Li, D. Yang, J. Shi, X. Chang, Z. Ye, Y. Wu, Z. Hong, J. Huang, J. Liu, Y. Ren, Y. Zou, Z. Zhao, S. Watanabe, AudioGPT: understanding and generating speech, music, sound, and talking head, in *Proceedings of the AAAI Conference on Artificial Intelligence 38(21)* (2024), pp. 23802–23804. https://doi.org/10.1609/aaai.v38i21.30570
99. S. Wu, H. Fei, L. Qu, W. Ji, T.-S.Chua, NExT-GPT: any-to-any multimodal LLM, in *Forty-First International Conference on Machine Learning* (2024)
100. Z. Tang, Z. Yang, C. Zhu, M. Zeng, M. Bansal, Any-to-any generation via composable diffusion. Adv. Neural Inf. Process. Syst. **36**, 16083–16099 (2023)
101. X. Wang, B. Zhuang, Q. Wu, ModaVerse: efficiently transforming modalities with LLMs, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 26606–26616
102. K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition (2015). https://doi.org/10.48550/arXiv.1409.1556
103. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778
104. Z. Liu, Y. Zhou, Y. Xu, Z. Wang, SimpleNet: a simple network for image anomaly detection and localization, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 20402–20411
105. I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2. NIPS'14*, vol. 2 (MIT Press, Cambridge, 2014), pp. 2672–2680
106. J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, in *Advances in Neural Information Processing Systems*, vol. 33 (Curran Associates, Red Hook, 2020), pp. 6840–6851
107. D.P. Kingma, M. Welling, Auto-Encoding Variational Bayes (2022). https://doi.org/10.48550/arXiv.1312.6114
108. G. Pang, C. Shen, L. Cao, A.V.D. Hengel, Deep learning for anomaly detection: a review. ACM Comput. Surv. **54**(2), 38 (2021). https://doi.org/10.1145/3439950
109. Z. Gu, B. Zhu, G. Zhu, Y. Chen, M. Tang, J. Wang, AnomalyGPT: detecting industrial anomalies using large vision-language models, in *Proceedings of the AAAI Conference on Artificial Intelligence 38(3)* (2024), pp. 1932–1940. https://doi.org/10.1609/aaai.v38i3.27963
110. L. Zanella, W. Menapace, M. Mancini, Y. Wang, E. Ricci, Harnessing large language models for training-free video anomaly detection, in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Seattle, 2024), pp. 18527–18536. https://doi.org/10.1109/CVPR52733.2024.01753
111. J. Su, C. Jiang, X. Jin, Y. Qiao, T. Xiao, H. Ma, R. Wei, Z. Jing, J. Xu, J. Lin, Large Language Models for Forecasting and Anomaly Detection: a Systematic Literature Review (2024). https://doi.org/10.48550/arXiv.2402.10350
112. T. Wang, Z. Zhao, K. Wu, Exploiting LLM embeddings for content-based IoT anomaly detection, in *2024 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)* (2024), pp. 1–6. https://doi.org/10.1109/PACRIM61180.2024.10690230
113. Z. Yang, B. Xu, W. Luo, F. Chen, Autoencoder-based representation learning and its application in intelligent fault diagnosis: a review. Measurement **189**, 110460 (2022). https://doi.org/10.1016/j.measurement.2021.110460
114. J. Jiao, M. Zhao, J. Lin, K. Liang, A comprehensive review on convolutional neural network in machine fault diagnosis. Neurocomputing **417**, 36–63 (2020). https://doi.org/10.1016/j.neucom.2020.07.088
115. S. Wang, J. Chen, H. Wang, D. Zhang, Degradation evaluation of slewing bearing using HMM and improved GRU. Measurement **146**, 385–395 (2019). https://doi.org/10.1016/j.measurement.2019.06.038
116. D. Neupane, M.R. Bouadjenek, R. Dazeley, S. Aryal, Data-driven machinery fault diagnosis: a comprehensive review. Neurocomputing **627**, 129588 (2025). https://doi.org/10.1016/j.neucom.2025.129588
117. P. Gupta, H. Kumar, D. Kar, K. Bhukar, P. Aggarwal, P. Mohapatra, Learning representations on logs for AIOps, in *2023 IEEE 16th International Conference on Cloud Computing (CLOUD)* (2023), pp. 155–166. https://doi.org/10.1109/CLOUD60044.2023.00026
118. J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (2019). https://doi.org/10.48550/arXiv.1810.04805
119. T. Ahmed, S. Ghosh, C. Bansal, T. Zimmermann, X. Zhang, S. Rajmohan, Recommending root-cause and mitigation steps for cloud incidents using large language models, in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)* (2023), pp. 1737–1749. https://doi.org/10.1109/ICSE48619.2023.00149
120. Y. Chen, H. Xie, M. Ma, Y. Kang, X. Gao, L. Shi, Y. Cao, X. Gao, H. Fan, M. Wen, J. Zeng, S. Ghosh, X. Zhang, C. Zhang, Q. Lin, S. Rajmohan, D. Zhang, T. Xu, Automatic root cause analysis via large language models for cloud incidents, in *Proceedings of the Nineteenth European Conference on Computer Systems. EuroSys'24* (Association for Computing Machinery, New York, 2024), pp. 674–688. https://doi.org/10.1145/3627703.3629553
121. C. Wang, X. Liu, Y. Yue, X. Tang, T. Zhang, C. Jiayang, Y. Yao, W. Gao, X. Hu, Z. Qi, Y. Wang, L. Yang, J. Wang, X. Xie, Z. Zhang, Y. Zhang, Survey on Factuality in Large Language Models: Knowledge, Retrieval and Domain-Specificity (2023). https://doi.org/10.48550/arXiv.2310.07521
122. N. Kandpal, H. Deng, A. Roberts, E. Wallace, C. Raffel, Large language models struggle to learn long-tail knowledge, in *Proceedings of the 40th International Conference on Machine Learning* (PMLR, 2023), pp. 15696–15707
123. Y. Luo, Z. Yang, F. Meng, Y. Li, J. Zhou, Y. Zhang, An Empirical Study of Catastrophic Forgetting in Large Language Models During Continual Fine-tuning (2025). https://doi.org/10.48550/arXiv.2308.08747
124. Y. Tan, D. Min, Y. Li, W. Li, N. Hu, Y. Chen, G. Qi, Can ChatGPT replace traditional KBQA models? An in-depth analysis of the question answering performance of the GPT LLM family, in *The Semantic Web – ISWC 2023*, ed. by T.R. Payne, V. Presutti, G. Qi, M. Poveda-Villalón, G. Stoilos, L. Hollink, Z. Kaoudi, G. Cheng, J. Li (Springer, Cham, 2023), pp. 348–367. https://doi.org/10.1007/978-3-031-47240-4_19
125. O. Ovadia, M. Brief, M. Mishaeli, O. Elisha, Fine-tuning or retrieval? Comparing knowledge injection in LLMs, in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, ed. by Y. Al-Onaizan, M. Bansal, Y.-N. Chen (Association for Computational Linguistics, Miami, 2024), pp. 237–250. https://doi.org/10.18653/v1/2024.emnlp-main.15
126. P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-T. Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive NLP tasks, in *Advances in Neural Information Processing Systems*, vol. 33 (Curran Associates, Red Hook, 2020), pp. 9459–9474
127. Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, Retrieval-Augmented Generation for Large Language Models: a Survey (2023). https://doi.org/10.48550/arXiv.2312.10997
128. S. Zhao, Y. Yang, Z. Wang, Z. He, L.K. Qiu, L. Qiu, Retrieval Augmented Generation (RAG) and Beyond: a Comprehensive Survey on How to Make Your LLMs Use External Data More Wisely (2024). https://doi.org/10.48550/arXiv.2409.14924
129. Z. Ye, L. Chen, R. Lai, W. Lin, Y. Zhang, S. Wang, T. Chen, B. Kasikci, V. Grover, A. Krishnamurthy, L. Ceze, FlashInfer: Efficient and Customizable Attention Engine for LLM Inference Serving (2025). https://doi.org/10.48550/arXiv.2501.01005
130. J. Juravsky, B. Brown, R.S. Ehrlich, D.Y. Fu, C. Re, A. Mirhoseini, Hydragen: high-throughput LLM inference with shared prefixes, in *Workshop on Efficient Systems for Foundation Models II @ ICML2024* (2024)

Publisher's note