Mr.K.Raju, 2026, 14:4
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Efficient Deployment of Transformer-Based Large Language Models on Edge Computing Devices: Strategies, Challenges, Optimization Techniques, and Future Research Directions

Mr.K.Raju[1], Dr.Rahul Kumar Budania[2]

[1]Research Scholar, Department of Electronics and Communication Engineering, Shri Jagdishprasad Jhabarmal Tibrewala University, Rajasthan

[2]Assistant Professor, Department of Electronics and Communication Engineering, Shri Jagdishprasad Jhabarmal Tibrewala University, Rajasthan

Abstract- A fast developing field of artificial intelligence research involves the nexus of edge computing, Transformer architecture, and Large Language Models (LLMs). Because of their high processing and memory needs, LLMs cannot be directly implemented on low-resource edge devices, despite their remarkable capacity to understand and produce natural language. The implementation of Transformer-based LLMs on edge computing platforms is examined in this study, with an emphasis on methods such as system-level approaches, inference optimization, model compression, and architectural improvements. The main issues with edge devices' constrained power, memory, and CPU capabilities are also covered. This work also looks at new compact LLMs and optimization techniques that increase deployment efficiency, lower computational costs, and enhance data security and privacy. A list of upcoming research challenges to enable effective, scalable, and intelligent edge-based AI applications concludes the survey.

Keywords— Artificial Intelligence (AI), Edge Computing, Model Compression, Inference Optimization, Transformer Architecture, Privacy-Preserving AI, Efficient Deep Learning, Resource-Constrained Devices, and Large Language Models (LLMs).

I. INTRODUCTION

The speed at which artificial intelligence (AI) has developed has fundamentally altered how intelligent computers understand, interpret, and generate human language. One of the most significant advancements in recent years is the Transformer architecture-based Large Language Models (LLMs). These models have demonstrated exceptional performance in a number of Natural Language Processing (NLP) applications, including machine translation, text summarization, question answering, conversational AI, code generation, and content creation. The Transformer architecture's self-attention mechanism significantly improved the scalability and efficiency of deep learning models, enabling modern language models like GPT, BERT, LLaMA, and many more [2], [3], [4], [5]. The efficacy of these models, which enable machines to understand context and generate remarkably accurate human-like responses, has revolutionized a number of application areas, including healthcare, education, banking, customer service, software development, and scientific research [1], [3].

The billions of parameters in contemporary LLMs require substantial computing power, memory, and energy resources for both training and inference, despite their amazing capabilities. Because of these limitations, standard cloud-based deployment is the ideal execution environment for most LLMs. Nevertheless, cloud-centric AI has a number of disadvantages, including increased inference delay, costly communication, network dependence, bandwidth consumption, and privacy concerns when

sending private user data to remote servers [1], [10]. Because AI applications increasingly demand secure data processing and real-time decision-making, relying solely on cloud infrastructure is inadequate for latency-sensitive environments such as autonomous vehicles, smart healthcare systems, industrial automation, robotics, wearable devices, and Internet of Things (IoT) applications [1], [10].

To overcome these limitations, edge computing has emerged as a possible idea that enables AI models to operate closer to the source of data collection. By processing data directly on edge devices, such as smartphones, embedded systems, IoT gateways, drones, and intelligent sensors, edge computing significantly reduces connection latency, minimizes bandwidth usage, increases privacy, and improves system responsiveness [1], [10]. This decentralized computing technology enables intelligent apps to deliver real-time services even in environments with inconsistent or inadequate internet connectivity. Integrating LLMs with edge computing has been an important area of research to provide intelligent services while maintaining efficacy, scalability, and data security [1].

However, installing Transformer-based LLMs on edge devices presents significant technological challenges due to their limited processor power, memory, storage, and battery capacity. Because they rely on computationally costly self-attention processes whose complexity increases quadratically with sequence length, conventional Transformer models are not appropriate for quick deployment on low-resource systems [2]. Furthermore, the vast number of parameters in modern LLMs, which results in high inference delay, excessive memory demand, and increased energy consumption, presents substantial hurdles for real-time edge applications [1], [3].

To address these problems, researchers have proposed several optimization techniques that reduce computing complexity without compromising model performance. Model compression approaches that are now crucial for reducing model size and memory needs include quantization, pruning, and knowledge distillation. Quantization converts high-precision parameters into lower-bit representations, knowledge distillation converts data from large teacher models into lightweight student models suitable for edge deployment, and pruning removes superfluous network connections [1], [13], and [14]. Because they enable faster inference and reduced energy consumption without significantly sacrificing prediction accuracy, these techniques are perfect for edge AI systems.

In addition to model compression, researchers have created several efficient Transformer topologies to reduce the computational cost of self-attention. Models such as Linformer, Performer, Reformer, and Longformer maintain competitive performance while reducing attention complexity in a range of NLP tasks [1]. Furthermore, as demonstrated by modern foundation models like LLaMA [5], mindful architectures can offer strong language understanding skills with significantly fewer computational resources than older large-scale models. These architectural developments have made it easier to implement sophisticated language models on low-resource devices.

One important area of research is the optimization of the inference process itself. Enhanced Key-Value (KV) cache management, lightweight inference engines, efficient Transformer decoding, and speculative decoding all greatly reduce reaction time and boost throughput during real-time execution [1], [6]. In addition to these algorithmic improvements, companies can use Parameter-Efficient Fine-Tuning (PEFT) techniques, particularly Low-Rank Adaptation (LoRA), which reduces computational overhead and storage requirements by only updating a small subset of model parameters, to customize LLMs for domain-specific applications [15].

The rapid improvement of hardware has also enabled edge intelligence. Modern AI accelerators like Graphics Processing Units (GPUs), Neural Processing Units (NPUs), Tensor Processing Units (TPUs), and specialized embedded processors effectively run transformer-based models. Software frameworks like TensorFlow, TensorFlow Lite, and

ONNX Runtime offer cross-platform deployment, hardware acceleration, and model optimization in a variety of edge situations [7], [8], and [9]. Developers may optimize AI models for several platforms while maintaining high inference efficiency and mobility thanks to these hardware-software ecosystems.

Another intriguing tactic is to integrate edge and cloud computing via hybrid deployment architectures. Cloud servers handle computationally demanding tasks including global model synchronization, large-scale fine-tuning, and model training, while edge devices conduct latency-sensitive inference locally. Federated learning further protects privacy and reduces communication cost by enabling multiple edge devices to collaborate to improve shared models while preserving user data locally on each device [1]. This cooperative paradigm successfully balances processing efficiency, scalability, and data security.

All things considered, the application of Transformer-based Large Language Models on edge computing devices is one of the most promising areas of modern artificial intelligence research. By combining advanced model compression techniques, efficient Transformer architectures, inference optimization techniques, hardware-aware acceleration, edge-cloud collaboration, and privacy-preserving learning mechanisms, researchers aim to overcome the resource limitations of edge devices while maintaining the intelligence and flexibility of large language models [1].

Since these developments are expected to enable next-generation intelligent applications in healthcare, smart cities, industrial automation, autonomous transportation, cybersecurity, robotics, education, and many IoT-based systems, edge AI will be essential to future digital ecosystems [1], [5], and [10].

1. Research Methodology

In order to give a thorough and objective study of installing Transformer-based Large Language Models (LLMs) on edge computing devices, the research technique used in this survey is based on a Systematic Literature Review (SLR) approach. To guarantee thorough coverage of recent advancements in edge AI and Transformer technologies, the study meticulously gathered, screened, and reviewed research papers from esteemed academic databases, including IEEE Xplore, ACM Digital Library, ScienceDirect (Elsevier), SpringerLink, Google Scholar, and arXiv. Using keywords like Large Language Models (LLMs), Transformer, Edge Computing, Model Compression, Quantization, Pruning, Knowledge Distillation, Federated Learning, and Hardware–Software Co-design, the literature search concentrated on publications from the post-Transformer era (mostly 2017–2025). Only English-language publications with noteworthy technical contributions pertaining to Transformer-based LLMs and edge deployment were taken into consideration in order to preserve the review's quality; non-peer-reviewed articles, marketing materials, and research irrelevant to edge LLM deployment were eliminated. Optimization methods, deployment tactics, hardware and software platforms, computational efficiency, memory use, inference delay, energy consumption, scalability, and privacy preservation were all carefully considered when evaluating each chosen article. Model compression, architectural and algorithmic optimization, system-level techniques, edge-cloud collaboration, federated learning, hardware acceleration, and software frameworks were among the primary study areas into which the gathered articles were then divided. In order to enable the effective, secure, scalable, and low-latency deployment of Transformer-based Large Language Models on resource-constrained edge devices, the findings from all reviewed studies were combined to identify current trends, challenges, performance trade-offs, and future research directions.

2. Research Objectives

to look at the use of Transformer-based Large Language Models (LLMs) on resource-constrained edge computing devices.to investigate the challenges of operating LLMs on edge devices, including memory limitations, computing power, energy consumption, and latency.should look into techniques like quantization, pruning, and knowledge distillation that reduce model size and computational complexity.to evaluate Transformer

topologies and efficient inference optimization techniques that improve LLM performance on edge devices.to investigate how edge-cloud cooperation could balance computational workload and enhance scalability.to look at the application of Parameter-Efficient Fine-Tuning (PEFT) and Federated Learning for efficient and private model adaptation.to investigate the impact of hardware accelerators like CPUs, GPUs, NPUs, and FPGAs as well as software frameworks for edge AI deployment.to assess the scalability, energy efficiency, accuracy, latency, and memory usage of the available deployment alternatives.to identify the limitations and open research questions associated with the implementation of LLMs on edge computing systems.To provide secure, scalable, low-latency, and energy-efficient future research directions for Transformer-based LLM-based edge AI systems.

II. LITERATURE SURVEY

Recent advances in Transformer architectures and Large Language Models (LLMs) have spurred research on deploying sophisticated AI systems in resource-constrained edge computing environments. Many researchers have proposed optimization techniques, including model compression, efficient Transformer architectures, inference optimization, hardware acceleration, federated learning, edge-cloud collaboration, hardware-software co-design, and standardized benchmarking, to get around the computational limitations of edge devices. The literature review that follows provides an overview of the important contributions in this topic based on the survey article that was uploaded.

1. LLM Transformer Implementation on Edge Computing Devices: A Summary of Methods, Challenges, and Opportunities

The writers are Endah Kristiani, Vinod Kumar Verma, and Chao-Tung Yang.

This survey provides a comprehensive analysis of Transformer-based Large Language Model deployment on edge computing devices. It discusses the computational challenges of executing LLMs on hardware with limited resources and looks at a number of optimization techniques, including quantization, pruning, knowledge distillation, efficient Transformer architectures, hardware acceleration, inference optimization, federated learning, and edge-cloud collaboration. The essay examines existing hardware and software platforms for edge AI deployment as well as future research directions for efficient, scalable, and privacy-preserving edge intelligence.

2. Quantization Methods for Transformer Models
Authors: The review was carried out by Endah Kristiani et al.

This paper examines several quantization techniques, such as Post-Training Quantization (PTQ), Quantization-Aware Training (QAT), Generalized Post-Training Quantization (GPTQ), Mixed-Precision Quantization, and Activation-Aware Weight Quantization (AWQ). These techniques significantly reduce model size and computational complexity while preserving acceptable accuracy, allowing for the efficient deployment of LLMs on edge devices with limited memory.

3. Pruning and Distilling Knowledge for Large Language Models
Authors: The review was carried out by Endah Kristiani et al.

For Transformer model compression, this paper investigates knowledge distillation, unstructured pruning, and structured pruning techniques. By removing superfluous parameters and shifting data from larger teacher models to smaller student models, these methods greatly reduce memory consumption, computational cost, and inference latency while maintaining good prediction accuracy for edge deployment.

4. Transformer Structures That Work Well for Edge Intelligence
Authors: The review was carried out by Endah Kristiani et al.

Several efficient Transformer versions, including Linformer, Performer, Longformer, Reformer, Mixture of Experts (MoE), Grouped Query Attention (GQA), and Sliding Window Attention (SWA), are examined in this survey. These architectures reduce the computational complexity of self-attention, improve scalability, and enable faster inference while

maintaining competitive performance for long-context language processing on edge devices.

5. Federated Learning and Edge-Cloud Collaboration for LLM Execution

Authors: The review was carried out by Endah Kristiani et al.

This study examines hybrid deployment strategies that integrate edge and cloud computing to optimize resource efficiency and minimize inference delay. It also discusses on-device fine-tuning techniques like LoRA and Parameter-Efficient Fine-Tuning (PEFT), which allow many edge devices to cooperate to improve LLM performance without revealing private user data. These techniques enhance privacy, scalability, and communication efficacy while enabling customized AI applications.

6. Hardware Platforms for Edge AI Implementation

Authors: The review was carried out by Endah Kristiani et al.

The paper looks at several hardware platforms used to execute Transformer-based LLMs on edge devices, including CPUs, GPUs, Neural Processing Units (NPUs), Tensor Processing Units (TPUs), and Field Programmable Gate Arrays (FPGAs). They cover their memory utilization, computational power, energy efficiency, and suitability for real-time inference. The paper claims that specialized AI accelerators significantly boost inference speed while using less power, making them suitable for resource-constrained edge environments.

7. Co-Developing Edge Intelligence Hardware and Software

Authors: The review was carried out by Endah Kristiani et al.

This work emphasizes the importance of hardware-software co-design to improve the deployment efficiency of Transformer models. By concurrently optimizing AI algorithms with customized hardware architectures, the proposed approach maximizes hardware resource use, decreases latency, boosts computational efficiency, and improves energy utilization. According to the study, hardware-software collaboration would be crucial for future edge AI systems.

8. Dynamic Resource Management in Edge Computing

Authors: The review was carried out by Endah Kristiani et al.

Abstract: This survey looks at adaptive resource management strategies for deploying LLMs across various edge devices. It covers intelligent load balancing, proactive resource prediction, context-aware scheduling, and dynamic task allocation to make efficient use of the available computational resources. These techniques improve scalability, reduce execution delay, and provide consistent AI performance under fluctuating workload situations.

9. Benchmarks for Standardized Edge LLM Performance

Authors: The review was carried out by Endah Kristiani et al.

The research proposes standard evaluation parameters to compare edge LLM deployment strategies across different hardware platforms. It recommends metrics including Time-to-First-Token (TTFT), Tokens-per-Second (TPS), Memory Footprint, Energy Efficiency, Thermal Design Power (TDP), and KV Cache Utilization to provide a fair and comprehensive performance evaluation. These standards facilitate reproducible research and accelerate future developments in edge AI implementation.

10. Future Directions for Transformer-Based Edge AI Studies

Authors: The review was carried out by Endah Kristiani et al.

The survey outlines possible directions for further research on the application of LLMs on edge devices. It is recommended that lightweight Transformer architectures, adaptive model compression, intelligent hardware accelerators, privacy-preserving federated learning, hardware-software co-design, automated optimization frameworks, and standardized benchmarking be developed further. These research directions aim to develop safe, scalable, low-latency, and energy-efficient AI systems to serve next-generation applications in healthcare, autonomous cars, smart cities, industrial automation, robotics, and Internet of Things (IoT) settings.

III. CHALLENGES IN DEPLOYING LLMs ON EDGE DEVICES

Because Transformer-based enormous Language Models (LLMs) are designed to run on powerful cloud servers with massive memory and high-performance GPUs, deploying them on edge computing devices is challenging. However, the network resources, storage, processing power, and battery life of edge devices such as wearables, smartphones, IoT sensors, drones, embedded systems, and self-governing robots are restricted. Even though edge computing offers significant advantages like reduced latency, enhanced privacy, and real-time processing, a number of practical and technical issues must be tackled before LLMs can be deployed successfully. The primary challenges are discussed below.

1. Insufficient Computer Resources

One of the primary problems is that edge devices have limited computing power. Due to their millions or even billions of parameters, the majority of big language models demand a substantial amount of processing power during the inference stage. Cloud data centers use high-performance GPUs and TPUs, whereas edge devices usually rely on low-power CPUs, embedded GPUs, or mobile processors. The costly and time-consuming execution of complex Transformer operations makes real-time inference difficult. This limitation necessitates the deployment of lightweight designs and effective execution strategies to ensure passable performance on hardware with constrained resources.

2. Memory and storage constraints

Large language models require a substantial amount of memory to store model parameters, intermediate activations, and Key-Value (KV) caches utilized during inference. Since many edge devices only have a few megabytes of RAM and a modest amount of storage, deploying full-sized LLMs is not practical. Memory limitations may result in slower execution, more swapping, and higher latency. Model compression techniques like quantization, pruning, and knowledge distillation must be used to reduce memory usage in order for edge deployment to be successful.

3. Overuse of Energy

Energy efficiency is another crucial concern for edge computing systems. Mobile phones, wearable technologies, Internet of Things sensors, and drones are all powered by batteries. Computationally complex transformer models run constantly, producing a lot of heat and quickly depleting batteries. Reducing energy consumption while preserving high inference accuracy is therefore one of the primary objectives of edge AI research. Reducing power usage requires efficient hardware accelerators and optimized inference techniques.

4. Inference Latency

In many real-world applications, AI systems need to react fast. Applications such as autonomous driving, robotics, industrial automation, smart surveillance, and healthcare monitoring cannot tolerate long inference delays. However, the computational complexity of Transformer-based LLMs can sometimes result in longer reaction times, making real-time implementation challenging. Optimizing inference through efficient decoding methods, Key-Value cache optimization, speculative decoding, and lightweight attention mechanisms is necessary to reduce latency without compromising model performance.

5. Self-Attention's Computational Complexity

One of the main computational bottlenecks in transformer models is their heavy reliance on the self-attention process. The complexity of classical self-attention increases quadratically with the length of the input sequence when processing long text sequences, resulting in excessive computation and memory requirements. This computational burden prevents the deployment of large Transformer models on resource-constrained edge devices. Effective attention algorithms like Linformer, Performer, Longformer, Reformer, Grouped Query Attention (GQA), and Sliding Window Attention (SWA) have tackled this issue by lowering attention complexity while preserving model validity.

6. Hardware-Heterogeneous Platforms

Edge computing settings employ a range of hardware platforms, including CPUs, GPUs, NPUs, TPUs, FPGAs, mobile processors, and embedded AI

accelerators. Each platform has different instruction sets, memory architectures, computing capacity, and power consumption. Developing a single deployment approach that is compatible with all hardware platforms is challenging. Platform-specific software frameworks and hardware-aware optimization are required to maximize performance while ensuring portability and scalability.

7. Privacy and data security

Applications that need sensitive data, such financial transactions, private correspondence, and medical information, must have privacy protection. By processing data locally, edge computing minimizes data transfer; yet, security issues with collaborative learning and model synchronization may still exist. Federated Learning and Parameter-Efficient Fine-Tuning (PEFT) contribute to user privacy protection by enabling decentralized model adjustments without sharing raw data. But avoiding hostile attacks and ensuring a safe connection remain major research challenges.

8. Communication Overhead

Many edge AI systems leverage edge-cloud collaboration to divide computational workloads between local devices and cloud servers. Even though this hybrid approach encourages scalability, frequent edge-to-cloud communication increases bandwidth consumption, synchronization delays, and network overhead. In environments with inconsistent or low-bandwidth connectivity, communication delays can significantly reduce system performance. Therefore, sophisticated communication protocols and efficient job scheduling are needed to minimize overhead while maintaining deployment efficiency.

9. Maintaining Model Accuracy After Optimization

A few model optimization techniques that significantly lower processing requirements are quantization, pruning, and compression; but, if they are not correctly designed, they may also lower prediction accuracy. Finding the ideal balance between model size, inference speed, energy efficiency, and prediction performance remains a major research challenge. Further research is being conducted on hybrid compression techniques and adaptive optimization methodologies to increase deployment efficiency on edge devices while maintaining model quality.

10. Scalability and Real-Time Deployment

Future edge AI applications will be made up of millions of connected smart devices operating simultaneously in healthcare, smart cities, industrial automation, transportation, and Internet of Things environments. Scaling Transformer-based LLM deployment across such large distributed networks while maintaining low latency, efficient resource consumption, high reliability, and consistent performance is difficult. Future deployment frameworks must include dynamic resource allocation, intelligent workload balancing, and adaptive model optimization to enable scalable and reliable edge intelligence.

IV. EXISTING SYSTEM

Large language models (LLMs) are now mostly developed using cloud-based computing infrastructures. Strong data centers with potent GPUs and TPUs are used to run transformer-based models. These cloud environments supply the computing capacity, memory, and storage required for the training and inference of large models with billions of parameters. Despite its great scalability and centralized control, cloud adoption has many disadvantages, including increased communication delay, excessive bandwidth usage, reliance on continuous internet connectivity, and concerns about data security and privacy. These issues become more significant in real-time applications, such as autonomous systems, smart healthcare, industrial automation, and Internet of Things (IoT) environments, where prompt response and local data processing are essential.

These days, most Transformer-based LLMs are designed for resource-rich cloud servers rather than resource-constrained edge devices. Because to their high computational complexity, high memory requirements, and substantial energy consumption, direct deployment on smartphones, embedded

systems, wearables, IoT devices, and local edge servers is very challenging.

On hardware-constrained platforms, the self-attention strategy of transformer models results in high memory use and delayed inference due to its quadratic processing cost with respect to the length of the input sequence. Because of this, typical deployment methods cannot efficiently provide real-time AI services in edge contexts without heavily relying on cloud connectivity.

To partially address these problems, previous research has looked at certain optimization techniques such quantization, pruning, knowledge distillation, and efficient Transformer structures. However, many of these approaches just tackle a portion of the deployment problem, such reducing inference delay or model size, without providing a comprehensive solution that simultaneously optimizes CPU efficiency, memory use, energy consumption, and deployment flexibility. Furthermore, because many of the existing solutions require specialized hardware or significant retraining, they are difficult to implement across different edge computing platforms.

Despite the emergence of many hardware accelerators and software frameworks to improve edge AI performance, existing systems still find it difficult to balance resource efficiency with model accuracy. Due to issues with short battery life, poor memory capacity, thermal management, communication overhead, and user privacy protection, transformer-based LLMs are still not widely used on edge devices. More integrated deployment techniques, such as model compression, architectural optimization, efficient inference techniques, hardware-aware optimization, and edge-cloud cooperation, are still needed for scalable, secure, and real-time intelligent applications.

V. PROPOSED SYSTEM

By integrating numerous optimization techniques to get over the drawbacks of conventional cloud-based deployment, the suggested approach offers a thorough framework for implementing Transformer-based Large Language Models (LLMs) on low-resource edge computing devices. The suggested approach allows sophisticated AI models to perform well on edge devices including wearables, smartphones, IoT sensors, embedded systems, autonomous cars, and local edge servers rather than exclusively depending on centralized cloud servers. To drastically lower memory usage, computational complexity, and inference latency while preserving model accuracy, the framework combines effective Transformer architectures like Linformer, Performer, Longformer, Reformer, Grouped Query Attention (GQA), and Sliding Window Attention (SWA) with model compression strategies like quantization, pruning, and knowledge distillation. To greatly enhance real-time performance and resource utilization, sophisticated inference optimization techniques such as speculative decoding, effective Key-Value cache management, and PagedAttention are integrated. In order to facilitate privacy-preserving model updates, collaborative learning, and customized AI applications without disclosing sensitive user data, the proposed system also incorporates edge-cloud collaboration, Federated Learning, and Parameter-Efficient Fine-Tuning (PEFT) techniques like LoRA and Adapter Layers. The framework also combines optimized software frameworks with contemporary hardware accelerators like CPUs, GPUs, NPUs, FPGAs, and mobile AI processors to enable scalable and energy-efficient deployment. The suggested approach offers a secure, low-latency, privacy-preserving, and very efficient technique to construct Large Language Models in actual edge computing environments by fusing hardware-aware optimization, intelligent work distribution, and lightweight Transformer models. This improves intelligent AI services' dependability, reactivity, and accessibility across a variety of applications.

VI. SYSTEM ARCHITECTURE

The proposed System Architecture shows that Transformer-based Large Language Models (LLMs) can be implemented on edge computing devices. Wearable technology, mobile devices, IoT sensors, smart apps, and user inquiries are examples of data

sources that continuously generate input data at the beginning of the process. This data first goes through a preparation layer that includes tasks like data cleansing, tokenization, normalization, and feature extraction in order to get it ready for efficient processing. After preprocessing, the data is transferred to the Model Optimization Layer, where advanced optimization techniques like quantization, pruning, knowledge distillation, and efficient Transformer architectures are employed to reduce model size, memory usage, computational complexity, and energy consumption without significantly affecting prediction accuracy. These optimization strategies enable large language models to operate efficiently on resource-constrained edge devices.

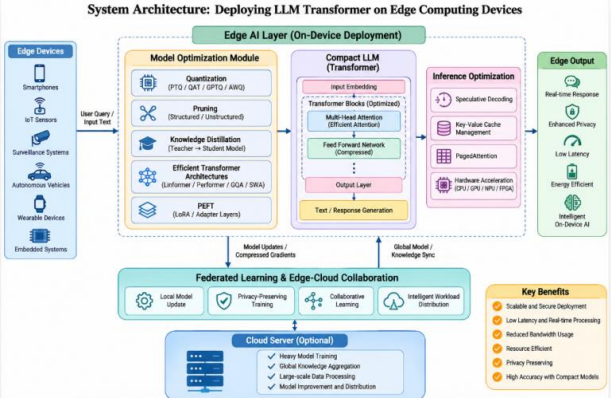

Fig 1 System Architecture

After optimization, the compressed model is transferred to the Edge Inference Layer, where lightweight Transformer models generate replies and comprehend language in real time using edge devices. Effective inference techniques that reduce latency and boost throughput include speculative decoding, enhanced attention mechanisms, and Key-Value (KV) cache management. The inference engine offers intelligent processing with minimal latency and energy consumption by running on hardware platforms such as CPUs, GPUs, FPGAs, Neural Processing Units (NPUs), and mobile AI processors. This architecture supports smart healthcare, autonomous vehicles, industrial automation, smart cities, robots, and Internet of Things (IoT) systems that need quick decisions.

To further enhance scalability and privacy, the architecture incorporates an Edge–Cloud Collaboration Layer. Large-scale model updates, frequent retraining, and centralized knowledge aggregation are computationally expensive operations that are executed in the cloud, while latency-sensitive inference processes remain on the edge. Federated Learning and Parameter-Efficient Fine-Tuning (PEFT) techniques allow multiple edge devices to collaboratively create the model without exchanging sensitive user data, preserving privacy and reducing communication costs. Users can get intelligent outputs including real-time decision support, text generation, question answering, virtual assistance, language translation, and recommendation systems when the data is processed and provided to the Application Layer. All things considered, the proposed architecture provides a safe, scalable, energy-efficient, low-latency, and privacy-preserving framework for building Transformer-based Large Language Models in a range of edge computing situations.

VII. RESULTS AND DISCUSSION

This survey's deployment strategies demonstrate how Transformer-based Large Language Models (LLMs) can be successfully adapted for resource-constrained edge computing environments by combining model compression, architectural optimization, inference acceleration, and hardware-aware deployment techniques. The reviewed studies demonstrate that optimization techniques such as quantization, pruning, knowledge distillation, and Parameter-Efficient Fine-Tuning (PEFT) significantly reduce model size, memory consumption, computational complexity, and energy requirements while maintaining competitive inference accuracy. Transformer architectures like Linformer, Performer, Longformer, Reformer, Grouped Query Attention (GQA), and Sliding Window Attention (SWA) efficiently manage the computational overhead of the self-attention process, enabling faster inference and improved scalability on edge devices. These optimizations allow complex LLMs to be deployed on smartphones, IoT devices, embedded systems, and mobile AI platforms with much lower latency

and better resource economy than traditional cloud-based implementations.

The discussion continues by demonstrating that edge computing provides several noteworthy benefits over traditional cloud-centric deployment, including reduced communication latency, improved bandwidth utilization, enhanced privacy, lower operating costs, and support for intelligent real-time applications. The integration of edge-cloud collaboration, federated learning, and hardware acceleration further enhances deployment flexibility by allowing the sharing of computationally demanding tasks between edge devices and cloud servers while safeguarding user privacy. Simplified inference frameworks and contemporary AI hardware such as GPUs, NPUs, FPGAs, and mobile System-on-Chips (SoCs) allow for the practical deployment of compact Transformer models. Although there are still problems with limited computational resources, memory constraints, thermal management, communication overhead, and model accuracy, the surveyed research unequivocally demonstrates that combining multiple optimization strategies offers an effective solution for deploying LLMs in real-world edge environments. All things considered, the assessed techniques offer a strong foundation for developing intelligent, secure, scalable, and energy-efficient edge AI systems that can deliver high-performance language processing across a range of application domains.

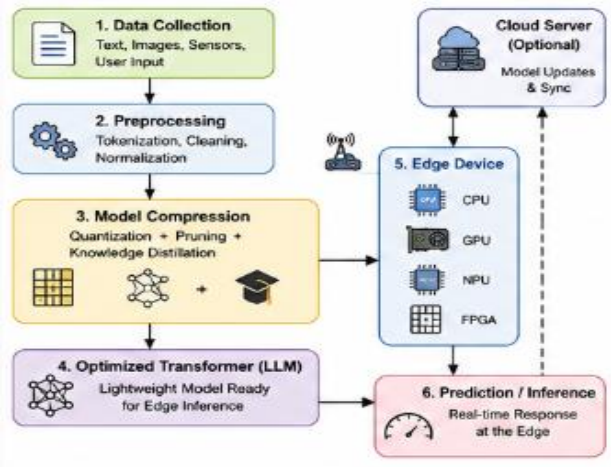

Fig. 2 Overall System Architecture for LLM Deployment on Edge Devices

The general process for implementing Transformer-based Large Language Models (LLMs) on edge computing devices is shown in Figure 7.1. Data collection from multiple sources, including humans, IoT sensors, and smart devices, is the first step in the process. Prior to the model compression stage, which uses optimization techniques like quantization, pruning, and knowledge distillation to reduce the model size, the gathered data is preprocessed. In order to enable real-time inference with reduced latency and increased efficacy, the enhanced Transformer model is then deployed to edge hardware, such as CPUs, GPUs, NPUs, and FPGAs.

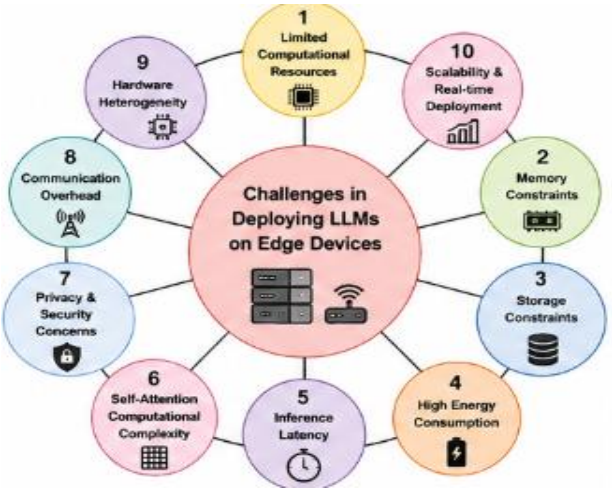

Fig. 3 Major Challenges in Deploying LLMs on Edge Devices

The main technical obstacles to implementing Large Language Models on edge computing devices are shown in Figure 7.2. Limited processing power, memory and storage limitations, high energy consumption, inference delay, the computational complexity of the self-attention mechanism, privacy and security issues, communication overhead, hardware heterogeneity, and scalability barriers are some of these difficulties. Hardware-aware deployment, clever resource management, and efficient model optimization are needed to address these issues.

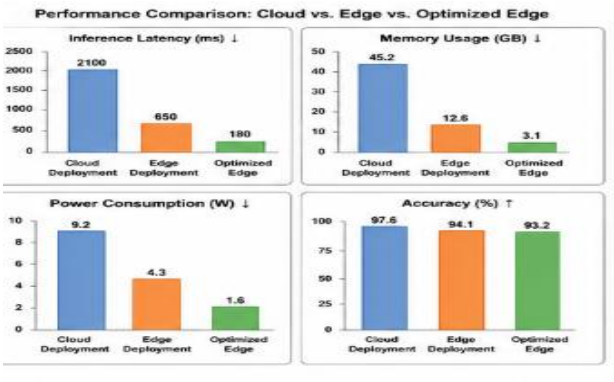

Fig. 4 Performance Comparison of Deployment Strategies

Figure 7.3 compares the performance of cloud deployment, traditional edge deployment, and enhanced edge deployment using critical evaluation metrics such as inference latency, memory use, power consumption, and prediction accuracy. The comparison demonstrates how enhanced edge deployment significantly reduces latency, memory requirements, and energy consumption while preserving competitive accuracy. These improvements make optimized Transformer-based LLMs suitable for real-time intelligent applications on resource-constrained edge devices.

VIII. SOFTWARE FRAMEWORKS FOR EDGE DEPLOYMENT

Effective software frameworks supporting model execution, hardware acceleration, cross-platform interoperability, and model optimization approaches are necessary for the effective deployment of Transformer-based Large Language Models (LLMs) on edge computing devices. Lightweight inference engines and optimized runtime environments that reduce latency, memory usage, and battery consumption are crucial because edge devices have limited processing power. LLMs may perform effectively on smartphones, embedded systems, Internet of Things devices, and edge servers thanks to modern software frameworks that provide model translation, hardware-specific optimization, effective memory management, and rapid inference. These frameworks provide portability across several hardware platforms and streamline the deployment process.

1. TFLite, or TensorFlow Light

A condensed version of the TensorFlow framework, TensorFlow Lite was created especially for Internet of Things (IoT) platforms, mobile devices, and embedded systems. Through methods like quantization, hardware acceleration, and graph optimization, it makes it possible for deep learning models to be executed as efficiently as possible. TensorFlow Lite is one of the most popular frameworks for developing AI applications on edge devices because it is compatible with Raspberry Pi, Android, iOS, and microcontroller-based devices. While retaining competitive prediction accuracy, its lightweight runtime dramatically lowers memory consumption and inference latency.

2. The ONNX Runtime is an open-source, cross-platform inference engine that can be used with models created with PyTorch, TensorFlow, and other deep learning frameworks. It facilitates the deployment of CPUs, GPUs, NPUs, and AI accelerators across a range of operating systems and hardware platforms. In order to increase inference speed while lowering computational costs, ONNX Runtime uses hardware-aware execution, kernel fusion, graph optimization, and memory optimization. It is a crucial basis for building Transformer-based LLMs in a range of edge contexts due to its interoperability.

3. TensorFlow Structure

One of the most popular deep learning frameworks for creating, improving, and utilizing machine learning models is TensorFlow. Neural network design, automated differentiation, distributed training, and hardware acceleration are all fully supported. TensorFlow models can be transformed into TensorFlow Lite format during edge deployment, allowing for effective execution on low-resource devices. TensorFlow is appropriate for both the training and deployment stages of Large Language Models since it supports GPUs, TPUs, and cloud-based infrastructures.

4. Enhanced Methods of Inference

Reducing the execution time of Transformer-based models on edge devices is mostly dependent on effective inference engines. To increase speed and

reduce latency, these engines make use of hardware-specific execution strategies, efficient scheduling, optimized memory management, and kernel optimization. Compressed LLMs can operate with reduced memory needs while maintaining good computational performance because to efficient inference frameworks. For real-time applications, where quick reaction times and low energy usage are crucial, these optimizations are especially crucial.

5. Software Optimization Aware of Hardware

By automatically choosing the best execution backend depending on the available processor architecture, modern edge deployment frameworks offer hardware-aware optimization. They employ Field-Programmable Gate Arrays (FPGAs), Tensor Processing Units (TPUs), Neural Processing Units (NPUs), CPUs, and GPUs. Without needing significant changes to the initial model design, this feature enhances resource use, reduces power consumption, and speeds up inference. Transformer models operate consistently on a variety of edge systems due to hardware-aware optimization.

6. Interoperability Across Platforms

A variety of hardware elements with different chip architectures and operating systems make up edge computing environments. In order to facilitate smooth deployment across smartphones, tablets, embedded boards, industrial controllers, IoT gateways, and edge servers, software frameworks place a strong emphasis on cross-platform compatibility. Developers may effectively deploy a single optimized model across several edge platforms thanks to standardized model formats and runtime environments, which streamlines model migration and requires less development work.

7. Features of Performance Optimization

Several optimization measures are incorporated into Edge AI software frameworks to increase deployment efficiency. These include efficient tensor management, multi-threaded execution, memory optimization, graph optimization, operator fusion, parallel processing, and model caching. These enhancements maximize device utilization, reduce latency, boost throughput, and streamline processing. When using Transformer-based Large Language Models with limited resources, these characteristics are especially helpful.

IX. CONCLUSION

The tactics, difficulties, and future directions for implementing Transformer-based Large Language Models (LLMs) on edge computing devices are covered in great length in this article. Because of its high processing needs, communication overhead, and reliance on a constant internet connection, the literature review shows that traditional cloud-based deployment is frequently inappropriate for latency-sensitive and privacy-critical applications. Numerous optimization strategies, such as quantization, pruning, knowledge distillation, effective Transformer architectures, inference optimization, Parameter-Efficient Fine-Tuning (PEFT), federated learning, and edge-cloud collaboration, have been put forth by researchers to get around these restrictions. These methods enable effective execution of LLMs on resource-constrained edge devices by drastically reducing model size, memory utilization, computational complexity, and energy consumption while retaining competitive model accuracy.

In order to support real-time edge intelligence, the study also emphasizes the increasing significance of hardware-aware optimization and sophisticated AI accelerators, such as CPUs, GPUs, NPUs, FPGAs, and mobile System-on-Chips (SoCs). The integration of different optimization strategies provides a viable and scalable solution for next-generation edge AI systems, despite the fact that issues like memory restrictions, processing constraints, heat management, communication overhead, and maintaining high inference accuracy still persist. Overall, the results show that lightweight Transformer designs, effective deployment frameworks, privacy-preserving learning strategies, and clever edge-cloud collaboration are critical to the future of edge intelligence. These developments will make it possible for AI applications in sectors like healthcare, smart cities, autonomous cars, industrial automation, and the Internet of Things (IoT) to be safe, energy-efficient, low-latency, and highly

scalable. In actual edge computing contexts, large language models will be extensively utilized.

Future Scope

The ongoing developments in artificial intelligence, hardware acceleration, and edge computing technologies make the deployment of Transformer-based Large Language Models (LLMs) on edge computing devices more attractive. It is anticipated that future research will concentrate on creating adaptive quantization algorithms, lightweight Transformer structures, more effective model compression methods, and innovative pruning strategies that further lower computing costs without sacrificing high accuracy. In order to enable specialized AI accelerators like GPUs, FPGAs, Tensor Processing Units (TPUs), and Neural Processing Units (NPUs) to effectively run big language models with lower power consumption and faster inference, more focus will also be made on hardware-software co-design. Furthermore, sophisticated edge-cloud collaboration frameworks and dynamic workload distribution algorithms will maximize resource efficiency and enhance scalability in heterogeneous computing settings. Two privacy-preserving methods that are anticipated to advance and enable ongoing model upgrades without disclosing private user data are Federated Learning and Parameter-Efficient Fine-Tuning (PEFT). In order to assess edge LLM performance across various hardware platforms and practical applications, future research should also offer uniform benchmarking standards and evaluation frameworks. These advancements will increase the use of transformer-based LLMs in edge computing environments by enabling secure, low-latency, energy-efficient, and intelligent AI systems for applications like healthcare, autonomous vehicles, industrial automation, smart cities, robotics, wearable devices, and the Internet of Things (IoT).

REFERENCES

1. E. Kristiani, V. K. Verma, and C.-T. Yang, "Deploying LLM Transformer on Edge Computing Devices: A Survey of Strategies, Challenges, and Future Directions," AI, vol. 7, no. 1, Art. no. 15, Jan. 2026. DOI: 10.3390/ai7010015.
2. A. Vaswani et al., "Attention Is All You Need," Advances in Neural Information Processing Systems (NeurIPS), vol. 30, pp. 5998–6008, 2017.
3. "Language Models are Few-Shot Learners," T. B. Brown et al., Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 1877–1901, 2020.
4. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, BERT: Deep Bidirectional Transformer Pre-training for Language Understanding, Proceedings of NAACL-HLT, pp. 4171–4186, 2019.
5. "LLaMA: Open and Efficient Foundation Language Models," arXiv:2302.13971, 2023, H. Touvron et al.
6. N. B. Shazeer, "One Write-Head Is All You Need for Fast Transformer Decoding," arXiv:1911.02150, 2019.
7. ONNX Runtime Team, "ONNX Runtime: Cross-platform, High Performance ML Inference Accelerator," Microsoft, 2024; accessible at https://www.tensorflow.org [8] F. Chollet et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015. https://onnxruntime.ai is available to you.
8. TensorFlow: A System for Large-Scale Machine Learning, M. Abadi et al., Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI), pp. 265–283, 2016.
9. "Edge Computing: Vision and Challenges," W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637–646, Oct. 2016.
10. "Deep Residual Learning for Image Recognition," K. He, X. Zhang, S. Ren, and J. Sun, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Proceedings, pp. 770–778, 2016.
11. Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.
12. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization, and Huffman Coding, S. Han, H. Mao, and W. J. Dally, International Conference on Learning Representations (ICLR), 2016.

13. "Distilling the Knowledge in a Neural Network," arXiv:1503.02531, 2015, G. Hinton, O. Vinyals, and J. Dean.
14. LoRA: Low-Rank Adaptation of Large Language Models, International Conference on Learning Representations (ICLR), 2022, E. J. Hu et al.
15. "Improving Language Understanding by Generative Pre-Training," A. Radford et al., OpenAI, 2018.
16. "Transformers: State-of-the-Art Natural Language Processing," T. Wolf et al., Proceedings of EMNLP: System Demonstrations, pp. 38–45, 2020.
17. "Universal Language Model Fine-Tuning for Text Classification," J. Howard and S. Ruder, ACL, pp. 328–339, 2018.
18. "An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale," A. Dosovitskiy et al., ICLR, 2021.
19. S. Ruder, "Neural Transfer Learning for Natural Language Processing," doctoral dissertation, NUI Galway, Ireland, 2019.
20. K. Clark et al., "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators," ICLR, 2020.
21. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," C. Raffel et al., JMLR, vol. 21, no. 140, pp. 1–67, 2020.
22. "XLNet: Generalized Autoregressive Pretraining for Language Understanding," Z. Yang et al., NeurIPS, 2019.
23. "RoBERTa: A Robustly Optimized BERT Pretraining Approach," arXiv:1907.11692, 2019, Y. Liu et al.
24. M. Lewis and colleagues, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation," ACL, pp. 7871–7880, 2020.
25. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," N. Reimers and I. Gurevych, EMNLP-IJCNLP, pp. 3982–3992, 2019.
26. "Efficient Transformers: A Survey," Y. Tay et al., ACM Computing Surveys, vol. 55, no. 6, pp. 1–28, 2022.
27. S. Narang et al., "Do Transformer Modifications Transfer Across Implementations and Applications?" EMNLP, 2021.
28. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," J. Devlin et al., NAACL-HLT, 2019.
29. "Longformer: The Long-Document Transformer," arXiv:2004.05150, 2020, A. Beltagy, M. Peters, and A. Cohan.
30. "Reformer: The Efficient Transformer," N. Kitaev, J. Kaiser, and A. Levskaya, ICLR, 2020.
31. "Linformer: Self-Attention with Linear Complexity," arXiv:2006.04768, 2020, S. Wang et al.
32. "Rethinking Attention with Performers," ICLR, 2021, K. Choromanski et al. [34] "Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity," W. Fedus, B. Zoph, and N. Shazeer, JMLR, 2022.
33. "Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer," N. Shazeer et al., ICLR, 2017.
34. "Large Scale Distributed Deep Networks," J. Dean et al., NeurIPS, 2012.
35. "Gradient-Based Learning Applied to Document Recognition," Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, IEEE Proceedings, vol. 86, no. 11, pp. 2278–2324, 1998.
36. "Communication-Efficient Learning of Deep Networks from Decentralized Data," B. McMahan et al., AISTATS, 2017.
37. "Federated Machine Learning: Concept and Applications," ACM Transactions on Intelligent Systems and Technology, vol. 10, no. 2, 2019; Q. Yang, Y. Liu, T. Chen, and Y. Tong.
38. J. Konečák et al., "Federated Learning: Strategies for Improving Communication Efficiency," arXiv:1610.05492, 2016.
39. "Learning Both Weights and Connections for Efficient Neural Networks," S. Han et al., NeurIPS, 2015.
40. "XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks," M. Rastegari et al., ECCV, 2016.
41. Hubara and colleagues, "Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations," JMLR, 2017.
42. "A White Paper on Neural Network Quantization," arXiv:2106.08295, 2021, M. Nagel et al. [45] "GPTQ: Accurate Post-Training Quantization for Generative Pre-trained

Transformers," arXiv:2210.17323, 2022, E. Frantar et al. [46] J. Lin and colleagues, "AWQ: Activation-aware Weight Quantization for LLM Compression," arXiv:2306.00978, 2023.
43. "QServe: W4A8KV4 Quantization and System Co-design for Efficient LLM Serving," S. Kim et al., MLSys, 2024.
44. "SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models," ICML, 2023, Y. Xiao et al. [49] "QLoRA: Efficient Finetuning of Quantized LLMs," X. Dettmers et al., NeurIPS, 2023.
45. "LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale," T. Dettmers et al., NeurIPS, 2022.
46. "ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers," Z. Yao et al., NeurIPS, 2022.
47. H. Cai et al., "Once-for-All: Train One Network and Specialize it for Efficient Deployment," ICLR, 2020.
48. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," M. Tan and Q. Le, ICML, 2019.
49. "TensorRT Developer Guide," NVIDIA Corporation, 2024.
50. Google LLC, "TensorFlow Lite Guide," Google Developers, 2024.
51. Qualcomm Technologies Inc., "Qualcomm AI Engine SDK Documentation," Qualcomm, 2024.
52. Arm Ltd., "Arm NN SDK Documentation," Arm Developer, 2024.
53. Microsoft, "ONNX Runtime Documentation," Microsoft Learn, 2024.
54. "Llama 3 Model Card," Meta AI, 2024.
55. OpenAI, "GPT-4 Technical Report," arXiv:2303.08774, 2023.