IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2

Edge AI Optimization: Techniques for Shrinking Large Language Models (LLMs) to Run Efficiently on Mobile Hardware

Vaishnavi Siddhtekakar and Prof. Sandeep Vishwakarma
Student of MCA
Head, Department of Information Technology
Chandrabhan Sharma College of Arts Commerce and Science, Powai, Mumbai, India
siddhatekakarvaishnavi27@gmail.com and sandeepvcbs@gmail.com

Abstract: *Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language understanding, generation, and reasoning. However, their deployment on mobile and edge devices remains challenging due to constraints in memory, computational power, energy consumption, and storage. Edge AI optimization focuses on reducing model size and computational requirements while maintaining acceptable performance. This paper explores key techniques used to shrink LLMs for efficient execution on mobile hardware, including model pruning, quantization, knowledge distillation, parameter-efficient fine-tuning, low-rank factorization, and hardware-aware optimization. The study also discusses challenges, applications, and future research directions in edge AI deployment.*

Keywords: Edge AI, Large Language Models, Model Compression, Quantization, Knowledge Distillation, Mobile AI, Tiny ML

I. INTRODUCTION

Large Language Models such as GPT, LLMA, Gemma, and Mistral have revolutionized artificial intelligence. Despite their impressive performance, these models often contain billions of parameters, requiring significant computational resources and memory.

Need for Edge AI Optimization

Traditional AI models are often large and require powerful cloud servers. Running these models on edge devices presents challenges due to limited hardware resources. Optimization techniques help reduce model size and computational requirements while maintaining acceptable accuracy.

Challenges in Edge AI Optimization

Challenge	Description
Limited Memory	Edge devices have restricted RAM and storage
Limited Power	Battery-powered devices require energy-efficient models
Computational Constraints	Lower processing capability compared to cloud servers
Accuracy Trade-off	Optimization may reduce model accuracy
Security Risks	Edge devices may be vulnerable to attacks

DOI: 10.48175/IJARSCT-37136

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2

What is Edge Computing?

Edge Computing is a distributed computing paradigm in which data processing occurs near the data source rather than in centralized cloud data centers.

Traditional Cloud-Based AI

- Device collects data.
- Data is sent to cloud servers.
- AI model processes the data.
- Results are returned to the device.

Edge AI Approach

- Device collects data.
- AI model runs directly on the device.
- Immediate results are generated locally.

limited hardware capabilities is Running LLMs directly on these devices can lead to:

- High memory consumption
- Increased battery drain
- Thermal issues

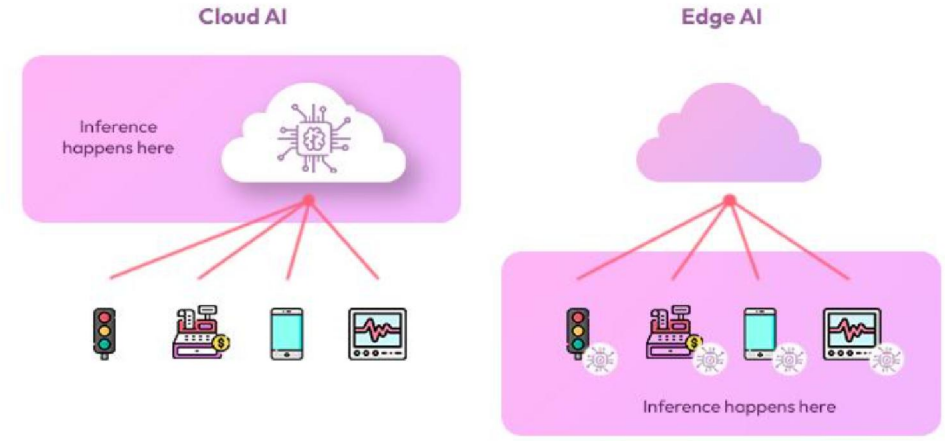
II. NEED FOR EDGE AI OPTIMIZATION

Benefits of running LLMs on mobile hardware include:

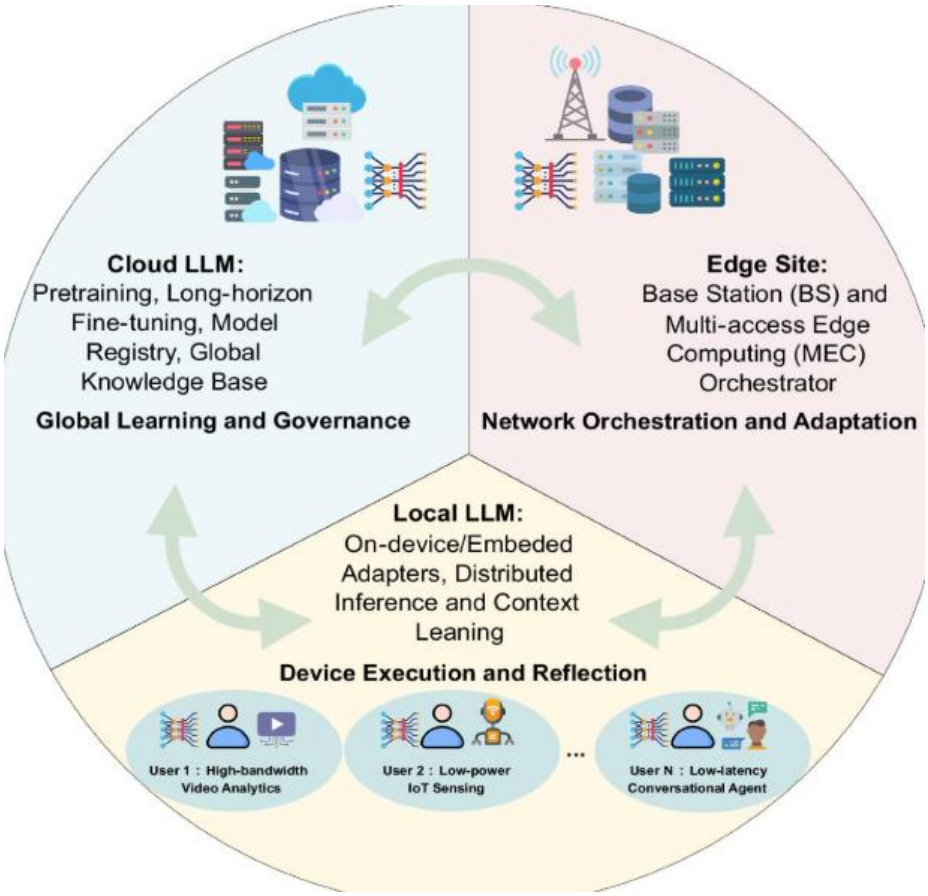
2.1 Reduced Latency

Processing occurs locally without sending data to cloud servers

Edge AI Architecture Diagram

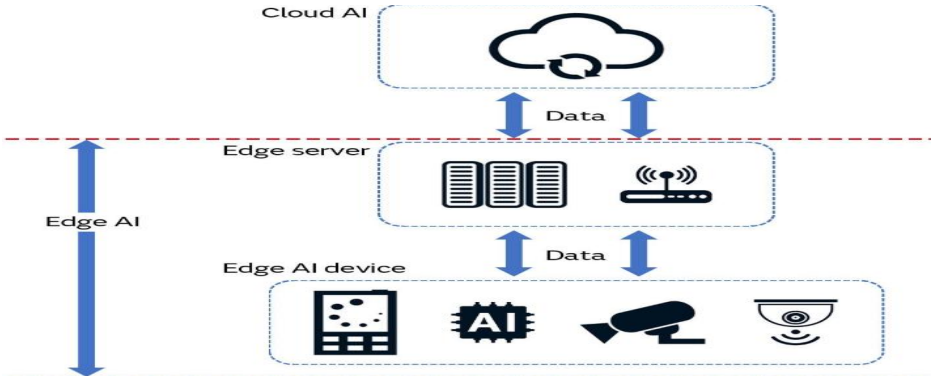


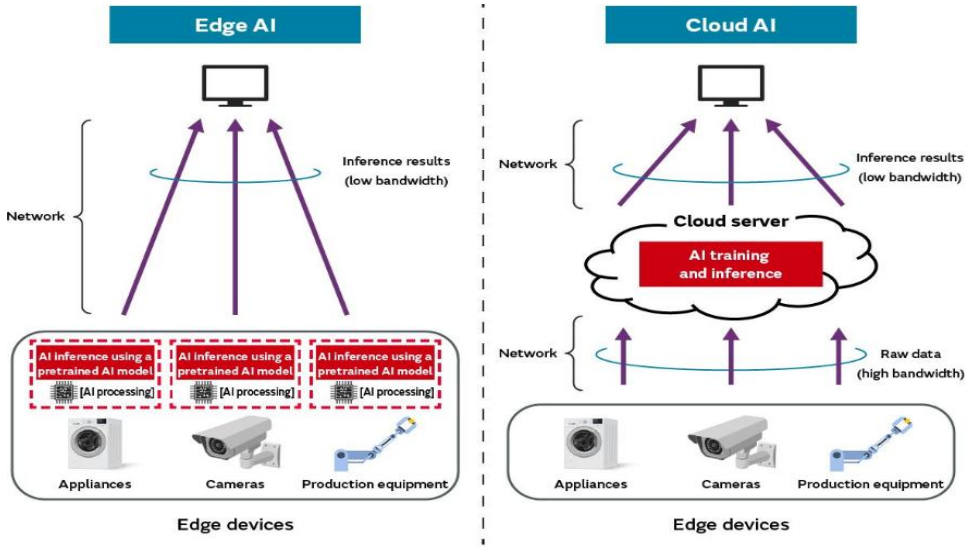
IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2



DOI: 10.48175/IJARSCT-37136

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2





2.2 Enhanced Privacy
Sensitive user data remains on the device.

2.3 Offline Functionality
Models continue working without internet connectivity.

2.4 Lower Cloud Costs
Organizations reduce server and bandwidth expenses.

2.5 Improved User Experience
Faster response times improve application performance.

III. TECHNIQUES FOR SHRINKING LARGE LANGUAGE MODELS

3.1 Model Pruning
Model pruning removes redundant or less important parameters from neural networks.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2

Types of Pruning

Unstructured Pruning

- Removes individual weights.
- Achieves high compression ratios.
- Difficult to accelerate on hardware.

Structured Pruning

- Removes entire neurons, channels, or layers.
- Better hardware compatibility.
- Faster execution on mobile processors.

Advantages

- Smaller model size
- Reduced memory requirements
- Faster inference

Challenges

- Potential accuracy degradation
- Requires retraining

3.2 Quantization

Quantization reduces numerical precision of model parameters.

Common Formats

Format	Bits
FP32	32
FP16	16
INT8	8
INT4	4

Example:

A 7-billion parameter model:

FP32 ≈ 28 GB

INT8 ≈ 7 GB

INT4 ≈ 3.5 GB

Benefits:

- Significant memory reduction
- Popular Quantization Methods
- Post-Training Quantization (PTQ)
- Quantization-Aware Training (QAT)

3.3 Knowledge Distillation

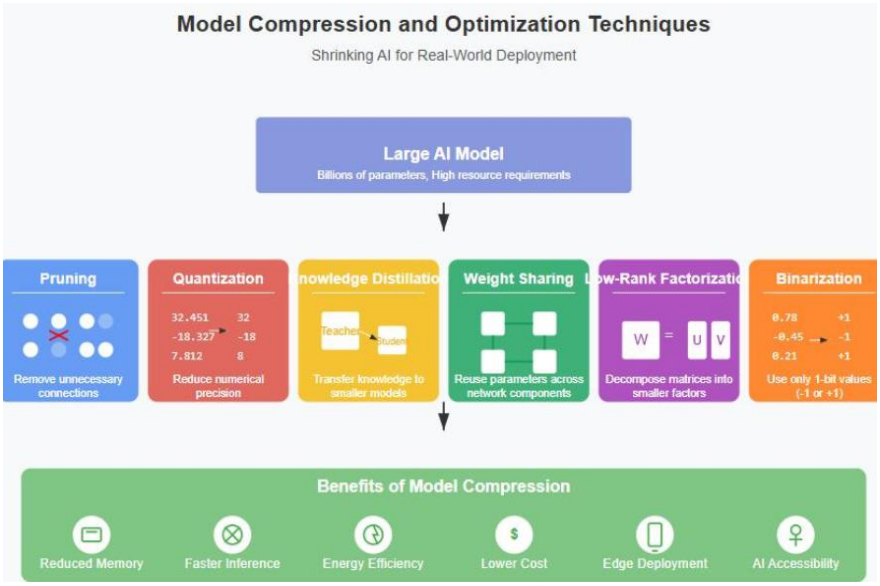
Knowledge Distillation transfers knowledge from a large "teacher" model to a smaller "student" model.

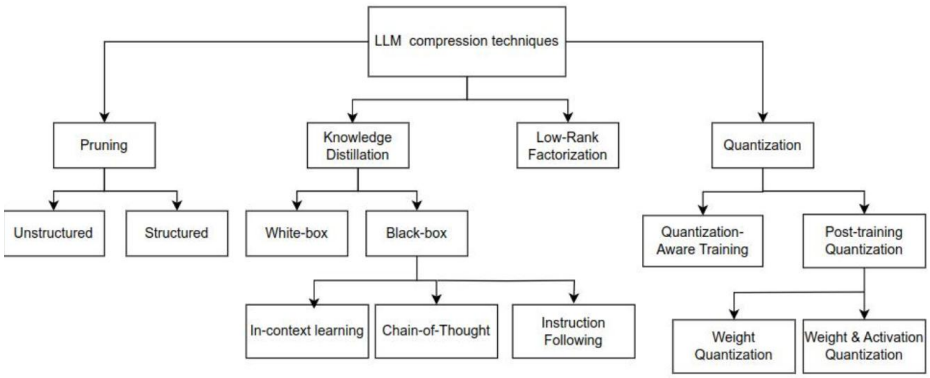
IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2

Process:
Train a large teacher model.
Generate soft labels.
Example:
Distil BERT achieves approximately 97% of BERT's performance while being much smaller and faster.

3.4 Low-Rank Factorization

Weight matrices in transformers often contain redundancy.
Low-rank decomposition splits large matrices into smaller matrices.
W ≈ A × B
Where:
A and B have fewer parameters
Storage requirements decrease





3.5 Parameter-Efficient Fine-Tuning (PEFT)

Instead of updating all model parameters, PEFT updates only a small subset.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2

LoRA (Low-Rank Adaptation)
Adds trainable low-rank matrices while keeping original weights frozen.
QLoRA
Combines:
Quantization
LoRA fine-tuning

3.6 Weight Sharing
Multiple network connections share the same weight values.

Advantages
- Reduced memory footprint
- Better compression rates

Applications
- Transformer compression
- Mobile AI deployment

3.7 Neural Architecture Search (NAS)
NAS automatically discovers efficient model architectures.

Objectives
- Minimize latency
- Reduce energy consumption
- Maintain accuracy

Mobile-Oriented Architectures
- MobileBERT
- TinyBERT
- MobileViT

IV. HARDWARE-AWARE OPTIMIZATION

4.1 Mobile GPUs
Examples:
- Adreno GPUs
- Mali GPUs
- Apple GPU

Optimizations:
- Parallel processing
- Quantized operations

4.2 Neural Processing Units (NPUs)
Modern smartphones include dedicated AI accelerators.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2

Examples:

- Apple Neural Engine
- Qualcomm Hexagon
- Google Tensor TPU

Benefits:

- Faster inference
- Lower power consumption

4.3 Compiler Optimizations

Frameworks include:

- TensorFlow Lite
- ONNX Runtime Mobile
- Core ML
- AI Engine

V. REAL-WORLD MOBILE LLM EXAMPLES

Google Gemma

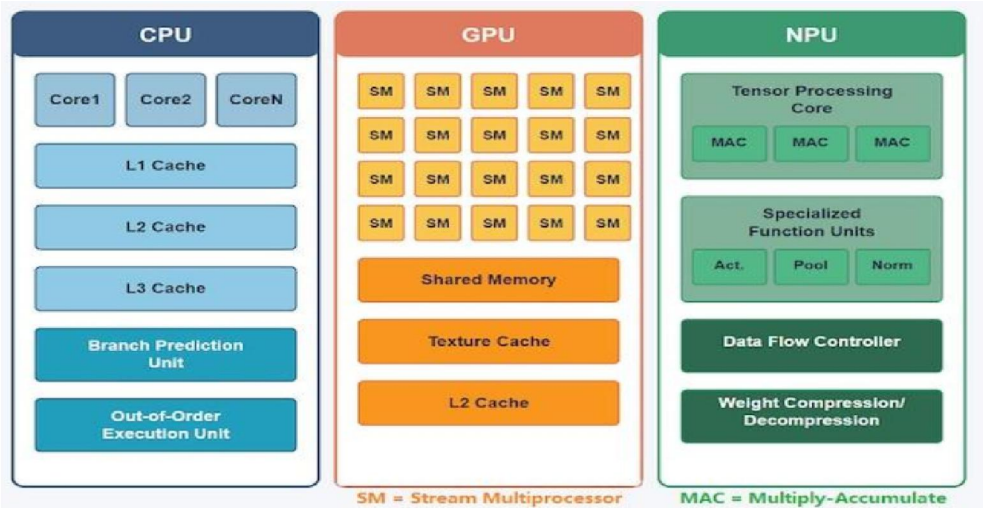
Designed to be lightweight and efficient for local deployment.

Microsoft Phi-3

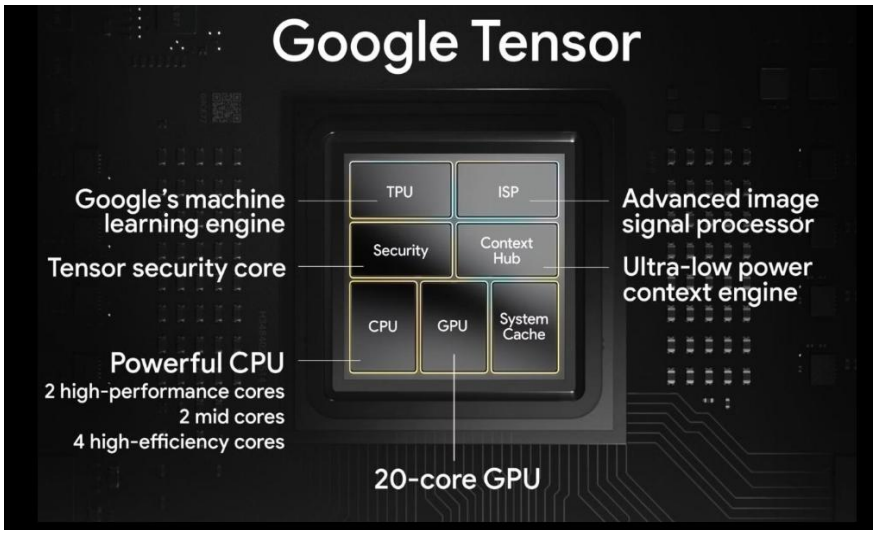
Small language models optimized for resource-constrained devices.

TinyLlama

Open-source compact LLM suitable for edge applications.



IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2



VI. CHALLENGES

Memory Constraints
Mobile devices typically have limited RAM.
Battery Consumption
Continuous inference impacts battery life.
Thermal Management
Heavy workloads generate heat.
Accuracy Trade-Offs
Compression may reduce model performance.
Security Risks
On-device models may be vulnerable to extraction attacks.

VII. FUTURE RESEARCH DIRECTIONS

7.1 Hybrid Edge-Cloud Systems
Combining local inference with cloud processing.

7.2 Dynamic Model Scaling
Adjusting model size based on available resources.

7.3 Sparse Transformer Architectures
Reducing computation through sparse attention mechanisms.

7.4 AI-Specific Mobile Hardware
Future NPUs designed specifically for LLM execution.

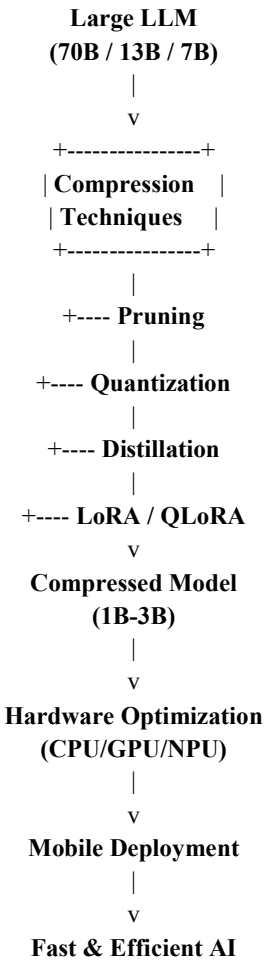
7.5 Federated Learning
Training models across devices while preserving privacy.

DOI: 10.48175/IJARSCT-37136

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2

7.6 Hardware Optimization

Fewer parameters and Less memory.



VIII. CONCLUSION

Edge AI optimization is essential for deploying Large Language Models on mobile and resource-constrained devices. Techniques such as pruning, quantization, knowledge distillation, low-rank factorization, and hardware-aware optimization significantly reduce model size and computational requirements. As mobile hardware continues to evolve, efficient edge deployment of LLMs will become increasingly practical, enabling privacy-preserving, low-latency, and offline AI applications. Future research will focus on balancing model performance with resource efficiency, making advanced AI accessible across a wide range of edge devices.

REFERENCES

[1]. G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” arXiv preprint arXiv:1503.02531, 2015.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, June 2026
Impact Factor: 8.2

[2]. S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," International Conference on Learning Representations (ICLR), 2016.
[3]. T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs," NeurIPS, 2023.
[4]. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv:1704.04861, 2017.
[5]. V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT: A Distilled Version of BERT," NeurIPS Workshop, 2019.
[6]. J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," NAACL, 2019.
[7]. A Vaswani et al., "Attention Is All You Need," NeurIPS, 2017.
[8]. T. Brown et al., "Language Models are Few-Shot Learners," NeurIPS, 2020.
[9]. H. Touvron et al., "LLaMA: Open and Efficient Foundation Language Models," Meta AI, 2023.
[10]. H. Touvron et al., "Llama 2: Open Foundation and Fine-Tuned Chat Models," Meta AI Research, 2023.
[11]. Google DeepMind, "Gemma: Open Models Based on Gemini Research and Technology," Technical Report, 2024.
[12]. TensorFlow Lite Documentation, TensorFlow, Google.
[13]. Core ML Documentation, Apple Developer.
[14]. Qualcomm AI Engine Documentation, Qualcomm Technologies.