BCSE498J Project-II – Capstone Project

Cross-Modal Knowledge Distillation for Ultra-Lightweight Edge AI: A Resource-Aware Approach

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

in

Computer Science and Engineering

(with specialization in Internet of Things)

by

21BCT0066 AARON MANO CHERIAN

Under the Supervision of

Dr. Manoov R

Assistant Professor Sr Grade 1

School of Computer Science and Engineering (SCOPE)

VIT

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

April 2025

DECLARATION

I hereby declare that the project entitled Cross-Modal Knowledge Distillation for Ultra-Lightweight Edge AI: A Resource-Aware Approach submitted by me, for the award of the degree of *Bachelor of Technology in Computer Science and Engineering* to VIT is a record of bonafide work carried out by me under the supervision of Dr. Manoov R

I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree ordiploma in this institute or any other institute or university.

Place : Vellore

Date : 24/04/2025

Signature of the Candidate

CERTIFICATE

This is to certify that the project entitled **Cross-Modal Knowledge Distillation for Ultra-Lightweight Edge AI: A Resource-Aware Approach** submitted by Aaron Mano Cherian (Reg No. 21BCT0066), **School of Computer Science and Engineering**, VIT, for the award of the degree of *Bachelor of Technology in Computer Science and Engineering*, is a record of bonafide work carried out by him / her under my supervision during Winter Semester 2024-2025, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The project fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place : Vellore

Date : 24/04/2025

Signature of the Guide

Internal Examiner **External Examiner**

Dr. Sharmila Banu

Department of IoT, SCOPE

ACKNOWLEDGEMENTS

I extend my heartfelt gratitude to the management of Vellore Institute of Technology (VIT) for providing the exceptional environment and resources that made this project possible. Their commitment to academic excellence has been fundamental to my growth as a researcher.

I am deeply indebted to Dr. Jaisankar N, Dean of the School of Computer Science and Engineering (SCOPE), whose leadership and steadfast oversight have been a constant source of motivation throughout this journey.

My sincere appreciation goes to Dr. Sharmila Banu, Head of the Department of IoT, SCOPE, for her guidance and encouragement that helped navigate the procedures regarding this project.

I am profoundly grateful to my project supervisor, Dr. Manoov R, whose mentorship, patience, and insightful feedback were instrumental in building up this project. His expertise and dedication have been an inspiration and I have much to learn from his leadership.

I would also like to acknowledge the creators of the various tools used, especially Docker and Kubernetes. My appreciation extends to the countless contributors to open source projects who have selflessly built the foundation upon which edge computing thrives today.

To my parents, whose unwavering belief in my abilities and constant support throughout my time at the Vellore Institute of Technology have been my greatest strength—your sacrifices and encouragement have made this achievement possible.

Aaron Mano Cherian

EXECUTIVE SUMMARY

Executive Summary

The Resource-Aware Cross-Modal Knowledge Distillation (RA-CMKD) framework tackles deployment issues with complex multi-modal deep learning models on limited-edge devices. Growing importance of edge AI applications makes it difficult to deploy computationally intensive multi-modal models because edge devices have restricted processing power and limited memory resources and energy consumption capabilities. RA-CMKD implements three strategic innovations through (1) a resource-adaptative framework for modality-based computational resource distribution influenced by device restrictions, (2) modality-tailored knowledge transfer systems that maximize memory efficiency and (3) performance-enhancing cross-modal feature alignment under different resource availability scenarios. The AV-MNIST dataset applied to a cloud-fog-edge infrastructure system showed experimental results which demonstrated better system performance than current techniques. Among the prototypes the teacher-model achieved 94.8% accuracy yet needed considerable resources which approached 256.4MB memory along with 128.7ms latency and 1250.6M FLOPs. The RA-CMKD framework operating under high resource conditions delivered 92.3% accuracy alongside memory declination by 83.4%, latency decrease to 74.5% and computational complexity reduction to 86.5%. Our system operated successfully with critical resource limitations when CPU utilization exceeded 90% while available memory fell below 20MB and it managed to achieve 82.8% accuracy along with 9.3ms latency. The ablation study established that each element in our three core components plays an important role in enhancing system performance. Self-supervised learning outperformed EMD-KD and TinyML methodologies because it proved to generate superior accuracy leveling with minimum resource utilization in every experimental set-up. Our approach provides an applied edge-device AI deployment framework that advances knowledge-distillation science about multi-modal frameworks under resource restrictions. The research will progress by expanding modalities in the framework while developing techniques for deployed model adaptation and precise resource estimations and by examining privacy benefits of edge-centric strategies through federated learning integration. As edge computing continues to evolve, approaches like ours that explicitly consider resource constraints will be increasingly critical for bridging the gap between theoretical AI capabilities and practical real-world deployments.

TABLE OF CONTENTS

List of Tables

List of Figures

List of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
CMKD	Cross-Modal Knowledge
Distillation	
CPU	Central Processing Unit
DL	Deep Learning
DNN	Deep Neural Network
GPU	Graphics Processing Unit
GSR	Galvanic Skin Response
IoT	Internet of Things
KD	Knowledge Distillation
ML	Machine Learning
NCFO	Normalized Carrier Frequency
Offset	
RAM	Random Access Memory
RGB	Red Green Blue
SRS	Software Requirements
Specification	
SSD	Solid State Drive

Symbols and Notations

α	Knowledge distillation temperature parameter
β	Resource allocation priority factor
γ	Compression ratio
δ	Feature alignment tolerance
λ	Loss function weighting coefficient
τ	Response time threshold
Φ	Feature transformation function
Ψ	Modality-specific compression function
Ω	Resource constraint set

Chapter 1

1. INTRODUCTION

1.1 BACKGROUND

The exponentially increasing demand for real-time artificial intelligence applications has paralleled the growth of edge computing applications, thus creating an urgent need for deploying artificial intelligence capabilities directly on resource-constrained devices. Edge devices, however, have created a significant challenge in deploying complex deep learning models. Edge devices, which include anything from smartphones and IoT sensors, to embedded systems, typically have limited computational resources, memory, and power capacity. This limitation poses a substantial barrier to implementing sophisticated multi-modal deep learning models that could otherwise provide rich, context-aware insights from various input sources such as visual, audio, and sensor data.

Contemporary technologies in edge AI often involve compromising between model performance and resource utilization, leading to either degraded accuracy or impractical resource requirements. While model compression techniques like pruning and quantization have shown potential, they often result in significant performance degradation when applied to multi-modal systems. This creates a critical need for more sophisticated approaches that can maintain high performance while dramatically reducing resource requirements.

The emergence of Cross-Modal Knowledge Distillation (CMKD) as a technique for transferring knowledge between different modalities offers new opportunities for creating ultra-lightweight edge AI systems. By distilling knowledge from a complex teacher model to a resource-efficient student model, CMKD can potentially maintain high performance while significantly reducing resource requirements.

1.1.1 Edge Computing Paradigm

In a world of relentless development and experimentation in architecture structures and processing paradigms, the need for more sophisticated processing power in the midst of limited memory and energy restraints have sparked a growing interest in the Edge Computing Paradigm for intensive sensor networks.

Edge Computing represents a distributed computing paradigm that brings computation and data storage nearer to the location where it is needed. Thus, it improves response times and reduces bandwidth usage. Within the context of AI systems itself, edge computing enables the deployment of AI workloads directly on or near end-devices, rather than relying on centralized cloud infrastructure alone. This paradigm shift has become increasingly important as the Internet of Things ecosystem develops, generating unprecedented volumes of useful data that require real-time processing and analysis to provide critical services.

The edge computing architecture typically consists of three distinct layers: cloud, fog, and edge. Beginning with the cloud layer, it provides centralized computing resources with virtually unlimited processing power and storage capacity but introduces latency due to data transmission distances. The fog layer serves as an intermediary infrastructure, typically consisting of local servers or gateways that bridge the gap between cloud and edge devices. Finally, the edge layer comprises the end devices themselves, including but not limited to smartphones and wearables, industrial sensors and smart home devices.

Some of the reasons why the edge computing paradigm is so effective is because of the way it handles latency, bandwidth, privacy and heterogeneity. By processing data closer to its source, the round-trip time required for data transmission to centralized servers is minimized, allowing for real-time decision-making. In recent years this has borne fruit in industries involving autonomous vehicles, industrial automation, and augmented reality.

Edge-level processing also allows for local data aggregation and filtration, transmitting only relevant information to the cloud rather than raw data streams. This approach significantly reduces network congestion and associated costs, making it economically viable for large-scale IoT deployments. Another serious concern as far as data is regarded is the matter of privacy and security. Sensitive data can be processed locally without transmission to external servers. This mitigates privacy concerns and reduces the attack surface for potential security breaches uniformly across multiple structures. This aspect holds relevant for applications in healthcare, finance, and personal devices.

Among all these characteristics, probably the most important aspect of the edge paradigm is its operational resilience. Edge systems can continue to function even during network outages or degraded connectivity, ensuring continuous operation in critical applications where downtime is unacceptable and real-time response is necessary. Looking at it from the larger picture, edge environments encompass devices with vastly different computational capabilities. This often takes shape in the form of powerful edge servers and resource-constrained microcontrollers, resulting in adaptive AI-related approaches that can scale across this spectrum to better implementation. This is often termed as heterogeneity.

Not all of it is sunshine and rainbows, however. The implementation of sophisticated AI models on edge devices presents several significant challenges. Traditional deep learning models, especially multi-modal ones that process different types of input data (images, audio, sensor readings), usually require substantial computational resources that exceed the capabilities of most edge devices. These models often demand gigabytes of memory and high-performance GPUs to achieve acceptable inference speeds. These resources are usually unavailable on energy-constrained edge hardware.

These limited resources often revolves around processing power, memory constraints, energy restrictions, thermal limitations and many more, depending on the area of application.

Edge devices typically feature low-power processors with constrained computational capabilities, often lacking specialized hardware accelerators for AI workloads. Available RAM on edge devices is usually limited to a few gigabytes or even megabytes, insufficient for loading large model parameters and activations. Many edge devices operate on battery power or limited energy budgets, necessitating energy-efficient computation to extend operational lifetimes. Compact form factors restrict heat dissipation capabilities, limiting sustained computational performance.

These irreplaceable constraints create a fundamental tension between the growing demand for sophisticated AI capabilities at the edge and the practical limitations of edge hardware. As a result, traditional approaches to deploying AI on edge devices often involve compromises between model accuracy and resource utilization, resulting in either degraded performance or impractical deployment requirements. This is in fact, the most pressing problem in the edge computing paradigm.

1.1.2 Cross-Modal Knowledge Distillation (CMKD)

Cross-Modal Knowledge Distillation (CMKD) provides an effective solution that resolves the deployment challenges of powerful artificial intelligence models on edge devices when working with multiple modalities. Traditional knowledge distillation only transfers knowledge between models of identical modality, yet CMKD establishes a knowledge transfer between models working with different modalities including visual to textual or audio to sensor or their potential combinations. Student models gain knowledge from their own modality along with new insights derived from different modalities through this technique. The data transfer method shows great potential when specific sensor information lacks sufficient quantity or lacks fidelity and proves expensive to process on edge systems.

The foundation of CMKD acknowledges that different data types contain useful information that relates to the same real-world phenomena. Usually, an image contains spatial details while audio data records temporal elements and emotional expressions. A teacher model with profound understanding of various modalities or multiple inputs will lead a student model which receives restricted data. During distillation the student model obtains abstract knowledge by aligning its outputs with the teacher model which it cannot discover from its limited input information. Edge AI benefits significantly from knowledge distillation because it enables access to limited information through student models which must operate under bandwidth or storage or energy constraints.

The most critical trait of CMKD in edge computing enables both performance retention and model simplicity reduction. The multi-modal deep learning models tend to be heavy in nature so they need special hardware and large memory databanks to process and fuse heterogeneous data sources. CMKD systems developed by existing implementations enable developers to train their complex robust systems first in cloud or fog layers before extracting edge-deployable compact models from them. The student models do not receive all the sensory inputs nor possess the complete structural attributes of their teacher models. The student models possess adequate contextual understanding to satisfy operational requirements in real-time scenarios. The current operational applications of CMKD include surveillance activities together with robotic systems and health monitoring and smart agriculture at fog or cloud layer levels.

The deployment strategy of CMKD provides organizations with adaptable model implementation options. The student model maintains single-modality capabilities yet draws knowledge from the entire multi-modality expertise of its teaching model to achieve multi-modal performance when functioning independently. The ability to adapt is essential because edge systems encounter changing sensor conditions when devices have unreliable connections in different time periods. A temperature sensor can work as a substitute for environmental visual information after receiving visual training through multi-modal distillation. CMKD, however, needs precise attention to modalities and requires representational transferability along with distillation objectives. These characteristics along with data representations between modalities suppresses noise characteristics and promotes semantic richness. To achieve effective student learning from teacher behavior the features between different modalities need alignment through shared embedding methods or attention systems and adversarial learning methods. The chosen loss functions during distillation require precise balancing to prevent student overfitting on teacher outputs while retaining performance on real-world edge tasks.

CMKD provides an important outcome that enhances both the robustness and interpretability features of edge models. The student learns to mimic the advanced teacher reasoning method across different modalities which leads to developing more abstract and semantically meaningful system representations. Thus, the student models exhibit better performance under noisy data as well as missing inputs and domain shift conditions, all of which often occur in dynamic edge environments. This architecture also has a modular design which allows progressive system upscaling. An edge device gets new sensor modalities and student models can be derived from existing multi-modal teachers without performing complete retraining.

The implementation of Cross-Modal Knowledge Distillation makes it possible to achieve capable resource-efficient AI operations at the edge. Through CMKD technology performance and resource requirements can be bridged effectively because centralized rich multi-modal systems train efficient autonomous models that work without dependency on constrained settings. CMKD will play a crucial part in edge AI expansion primarily because it enables powerful distributed intelligence to tolerate data conditions across various infrastructures.

Knowledge distillation techniques through Cross-Modal Knowledge Distillation (CMKD) represent an effective solution which helps transfer teacher model training knowledge to student models that maintain performance levels. This technique finds its great value in multi-modal scenarios because it enables efficient model architecture design for edge deployment.

1.2 MOTIVATIONS

Multiple key elements drive the development of Resource-Aware Cross-Modal Knowledge Distillation framework to deliver ultra-lightweight edge AI systems because of the growing requirements for performance-optimized multi-modal AI systems which function within edge resource boundaries.

Research shows edge computing applications will keep growing rapidly until 2025 when the market value will reach $15.7 billion at a 34.1% annual rate. This demand originates from certain growing sectors, especially smart cities and industrial IoT, and healthcare and autonomous vehicles systems. The domains require multi-modal artificial intelligence solutions to generate context-driven decisions because each of these areas utilizes this functionality. Current deep learning techniques encounter substantial resource limitations which prevents their comprehensive deployment path leading to halting various application potentials.

Existing AI methodologies, however, do not solve edge environment-related resource problems which could impose severe limitations on the Edge-computing paradigm.

Table 1.1 Resource Constraints in Edge Environements

Constraint	Reason
Processing power limitations	typically sub-5W TDP processors
Memory constraints	often <4GB RAM
Energy budgets	battery-powered operation requirements
Thermal restrictions	passive cooling requirements
Network bandwidth limitations	intermittent connectivity

The existing approaches demonstrate inadequate performance-resource utilization management especially when processing multiple data types simultaneously. Novel approaches need development immediately since they must maintain high model accuracy levels by cutting resource needs substantially.

Edge-based data processing from an operations perspective raises privacy concerns as well as regulatory data standards implementation in initiatives such as the GDPR and CCPA. In the world that we live in today, autonomous driving and industrial control systems together with augmented reality require instantaneous processing which cloud solutions cannot deliver dependably because of changing network delays. Device limitations demand the development of AI solutions which function optimally at the edge since these situations produce hardware-specific demands.

Existing model optimization techniques demonstrate weaknesses when used in multi-modal systems because they lead to decline of performance levels which accompanies their success in reducing model size and computational requirements. These systems possess delicate cross-modal interactions which make them react strongly to optimization techniques missing inter-modal relationship considerations. The requirement emerges for advanced approaches that will fulfill the needs of deploying multi-modal edge platforms.

Cross-Modal Knowledge Distillation (CMKD) showcases its ability to share knowledge across different modalities without compromising on performance outcomes. The method presents an exciting solution to produce edge AI systems with minimal weight which extract combined data benefits from multiple sensor categories under constrained hardware settings. CMKD enables the transfer of information from complicated multi-modal teacher networks to basic student networks which enables substantial resource saving without sacrificing performance quality.

Edge devices run in dynamic settings because their accessible resources change because of multiple factors such as simultaneous applications and battery levels and thermal fluctuations. The present solutions fail to react properly to changing conditions which results in system resource waste through excessive use or inadequate utilization. Implementing CMKD systems with a smart and accurate resource-based understanding would allow Edge AI systems to dynamically adapt their functioning according to the available resources, resulting in improved performance across different environments.

Multiple edge computing infrastructure presents multifaceted challenges for the deployment and maintenance of multi-modal AI systems because of model update requirements and performance monitoring as well a A complete framework combining efficient multi-modal features together with comprehensive solutions for these areas would lower the obstacles for broad edge AI implementation across different applications.

Thus, pursuing such a framework would inevitably address the fundamental challenges of deploying multi-modal AI capabilities on resource-constrained devices, enabling a new generation of intelligent edge applications across various domains.

1.3 SCOPE OF THE PROJECT

The project focuses on building and executing and testing an entire Resource-Aware Cross-Modal Knowledge Distillation (CMKD) framework designed for edge AI applications with extreme light design requirements. This architecture tackles the critical deployment obstacles faced by multi-modal sophisticated AI systems on limited-edge hardware devices by delivering suitable operational results.

The development of a modular system architecture should enable various teacher-student model combinations between audio-visual and sensor modalities and visual-audio and audio-sensor pairings. The design will contain resource monitoring and adaptation modules which allow runtime changes according to available resources.

This architecture often involves resource allocation mechanism that collects real-time usage statistics of computational resources memory and energy followed by automatic resource distribution across different modalities. Relying on adaptive scheduling algorithms, this mechanism will make critical modalities its priority but protect overall system performance at the same time. Specialized compression algorithms will focus on modulation-specific forms (visual, audio, sensors) to optimize their efficiency rates by sustaining crucial information from each unique data set structure.

The processing of visual data requires optimized quantization schemes together with structured pruning methods which are calibrated for edge-based implementations. For audio data: Frequency-domain compression, temporal pooling, and selective feature retention. Improvement of sensor data requires statistical aggregation together with anomaly-focused encoding methods and dimension reduction procedures. The Feature Alignment Strategy develops basic procedures to align knowledge transfer between different modalities without causing substantial increases in processing requirements. The research introduces optimized subspace projection techniques along with canonical correlation analysis adaptations and contrastive learning solutions which work well in resource-limited environments.

An adaptive loss function should be created to balance knowledge distillation efficiency against computational requirements through an automatic adjustment mechanism which depends on resource availability as well as application priority levels.

This project will cover the implementation of a streamlined deployment workflow that facilitates model training, distillation, optimization, and deployment across different edge devices, with support for continuous monitoring and adaptation. It will focus on implementing the framework using PyTorch as the primary deep learning framework, with TensorRT and OpenVINO for model optimization and deployment. The implementation will target three distinct hardware tiers: The implementation will leverage containerization (Docker) and orchestration (Kubernetes) technologies to facilitate deployment and management across these different tiers. While the framework will be designed for general applicability, the project will specifically evaluate its effectiveness in three key application domains:

Multi-modal sensing and intelligence for home automation, energy management, and security applications, physiological signal processing and analysis for health tracking and anomaly detection and finally Industrial IoT applications such as Condition monitoring, predictive maintenance, and anomaly detection in industrial environments.

The assessment covers numerous domains which exhibit different performance goals alongside their resource limitations and modal combinations for extensive framework testing.. Testing will create an evaluation procedure to measure technical framework abilities and verify its performance within practical real-world conditions. The performance evaluation employs accuracy retention ratio together with inference latency and memory utilization and computational efficiency and energy consumption metrics.

The proposed work implements a systematic comparison with current methods that includes both traditional model compression and single-modal knowledge distillation and non-resource-aware CMKD approaches. Research-based evaluation determines the performance contributions of each component in the framework by studying components independently. Real-World Testing revolves around the deployment of real edge devices under natural operating conditions with a focus on performance metrics under various environmental situations and device-resource-availability parameters.

This comprehensive scope provides a clear framework for the project's development while ensuring a focused approach that addresses the critical challenges of deploying multi-modal AI systems on resource-constrained edge devices.

Chapter 2

1. PROJECT DESCRIPTION AND GOALS

2.1 LITERATURE REVIEW

2.1.1 Cross Modal Knowledge Distillation Approaches

Cross-Modal Knowledge Distillation (CMKD) provides an efficient method to transfer information between models which operate on contrasting data modalities. This method enhances the function of standard knowledge distillation when teachers and students use different data types to learn from each other. Researchers specializing in cross-modal knowledge transfer have developed numerous new approaches during the past few years because of the exclusive difficulties of this field.

2.1.1.1 Foundational Approaches to CMKD

One of the earliest CMKD researches emerged from Gupta et al. (2016) with their paper "Cross Modal Distillation for Supervision Transfer." Through this research the authors created a fundamental framework which proved the effectiveness of using established RGB knowledge to supervise depth learning while handling limited depth training data. Feature-level distillation served as the approach which transferred guidance from teacher network representations to student network layers for learning. The methodology showed that transferring knowledge between different modalities could reach comparable performance to full supervision while needing less target modality labeling.

Hoffman et al. (2016) developed this idea through research which proved networks that learned from RGB video training could apply knowledge to networks analyzing 3D human pose sequences. The researchers adapted their method to accommodate temporal dynamics particularities of action recognition showing CMKD works well with sequential data between different modality systems.

2.1.1.2 Feature Alignment in CMKD

The most essential issue in CMKD requires successful alignment of features between different modalities for effective knowledge transfer to happen. Garcia et al. (2018) developed "Mutual Information Maximization" which selects maximum information sharing between modalities instead of applying direct feature matching. The method relies on maximizing the information relationship known as I(X; Y) between characteristics of source X and target Y and can be represented as below:

$$I(X; Y) = H(X) - H(X|Y) \qquad \text{----- (1)}$$

The formula determines entropy measurement by calculating H(X) plus H(X|Y). The approach enables variable alignment capacity which manages the different characteristics within modalities.

Tian et al. (2020) used "Contrastive Cross-Modal Knowledge Distillation" which implements contrastive learning to enhance the alignment between features. The method establishes positive sets consisting of matching attributes between modalities and negative sets which include non-matching features and maximizes a contrastive loss.

$$L_{contrast} = -\log[\exp(\mathrm{sim}(f_t, f_s)/\tau) / \Sigma_i \exp(\mathrm{sim}(f_t, f_i)/\tau)] \qquad \text{----- (2)}$$

The algorithm uses corresponding teacher and student features f_t and f_s along with similarity function sim(·,·) and temperature parameter τ in its computation while performing summation over every possible pair. The method produced superior matching results especially when dealing with significant modality variation.

2.1.1.3 Multi-Teacher CMKD

The authors Zhao et al. (2020) analyzed cross-modal knowledge transfer from multiple teachers to students in their paper "Knowledge as Priors: Cross-Modal Knowledge Generalization for Datasets Without Superior Knowledge." The method utilizes various teacher networks focusing exclusively on different modalities to deliver complete knowledge to the student network. The authors designed the distillation process into the following mathematical representation:

$$L_{distill} = \Sigma_m \alpha_m KL(p_t^m \| p_s) \qquad \text{----- (3)}$$

The distillation process involves mathematical operations between outputs from m-various teacher models represented through p_t^m and the student distribution p_s using $KL(\cdot\|\cdot)$ divergence which is weighted by modality reliability values α_m. The method demonstrated strong potential to benefit instances that need combined information from multiple sources.

2.1.1.4 Attention-Based CMKD

Attention mechanism technologies have been incorporated into CMKD frameworks because of their recent advancements. The "Cross-Modal Attention Distillation" method presented by Li et al. (2022) enables attention map transfer between teacher-student networks operating on separate modalities. The process generates teacher and student attention maps A_t and A_s while seeking to reduce their difference between them.

$$L_{attention} = \|A_t - A_s\|F^2 \qquad \text{----- (4)}$$

This equation minimizes the difference between attention maps A_t and A_s while using the Frobenius norm as $\|\cdot\|F$. This system allows the student model to receive enhanced direction toward critical teacher-identified features that span different modalities.

2.1.1.5 Modality-Specific Applications of CMKD

The EmotionKD framework, proposed by Liu et al. (2023), demonstrated the application of CMKD in emotion recognition using physiological signals. This research showed effective knowledge transfer from complex EEG-based models to simpler GSR-based implementations, maintaining high performance while significantly reducing complexity. Their approach incorporated modality-specific considerations for physiological signal processing, illustrating how CMKD can be adapted to specialized domains.

In the realm of action recognition, Zimmermann and Brox (2017) used CMKD to transfer knowledge from RGB-based hand pose estimation models to depth-based models, achieving state-of-the-art performance on standard benchmarks. Their approach demonstrated that CMKD could effectively bridge the domain gap between different visual modalities, even when they capture fundamentally different aspects of the same scene.

2.1.1.6 Generalization and Transfer Learning in CMKD

Zhao et al. (2020) deepened the examination of CMKD during transfer learning processes that use scarce datasets. The knowledge representation used prior distributions on student network parameters which permitted the model to extend itself to unpaired data across modalities. This approach is formalized as:

$$p(\theta_s|D_s) \propto p(D_s|\theta_s)p(\theta_s|\theta_t) \qquad \text{----- (5)}$$

This model contains student parameters θs along with teacher parameters θt while introducing Ds as student training data and calculating $p(Ds|\theta s)$ for data likelihood and $p(\theta s|\theta t)$ represents the teacher-conditioned student parameter priors. The Bayesian formulation of CMKD established its theoretical capability to handle transfer learning tasks.

2.1.1.7 Current Limitations in CMKD Literature

Despite significant advances, current CMKD approaches exhibit several limitations that this project aims to address:

Table 2.1 Limitations in CMKD Literature

SL No.	Limitation Identified	Reason/Case Study
1.	Resource Awareness	Existing CMKD methods primarily focus on performance optimization without explicitly considering resource constraints, making them impractical for direct application in edge computing scenarios.
2.	Dynamic Adaptation	Current approaches typically employ static distillation processes that do not adapt to changing resource availability or operational conditions.
3.	Modality-Specific Compression	The literature lacks comprehensive exploration of modality-specific compression techniques integrated with knowledge distillation processes.
4.	Quantifiable Resource Metrics	Standardized approaches for measuring and optimizing resource utilization in CMKD systems remain underdeveloped.

The field of Cross-Modal Knowledge Distillation has evolved significantly in recent years, with various approaches emerging to address the challenges of transferring knowledge between different modalities. Current research has demonstrated the effectiveness of CMKD in several domains, including action recognition, emotion detection, and image processing.

In the context of action recognition, researchers have successfully shown that networks trained on RGB videos can be adapted to recognize actions from 3D human pose sequences. This transfer of knowledge between modalities has proven particularly effective in scenarios where data in the target modality is limited or difficult to obtain. Emotion recognition research has demonstrated the potential of CMKD in transferring knowledge from complex, multi-modal systems to simpler, single-modal implementations. For instance, the EmotionKD framework has shown promising results in transferring knowledge from EEG-based systems to simpler GSR-based implementations, maintaining high performance while significantly reducing complexity.

The application of CMKD in image processing has revealed interesting approaches to supervision transfer, particularly in scenarios where direct supervision in certain modalities is expensive or impractical to obtain. These approaches have demonstrated the potential for creating more efficient, lightweight models while maintaining acceptable performance levels.

Table 2.2 Crucial Papers- Literature Review

Paper Topic	Paper Description
Cross-Modal Knowledge Distillation for Action Recognition	CMKD allows networks trained on one modality, such as RGB videos, to be adapted to recognize actions in another modality, such as sequences of 3D human poses. The process steps include: 1) Extract the knowledge of the trained teacher network (Source Modality), 2) Transfer it to a small ensemble of student networks (Target Modality). Commonly used loss: KL-loss. Preferred loss: Cross-entropy loss +

	Mutual learning. **Link**: https://sci-hub.ru/https://ieeexplore.ieee.org/abstract/document/8802909
Knowledge as Priors: Cross-Modal Knowledge Generalization for Datasets Without Superior Knowledge	In cases of inadequate data, one can generalize distilled cross-modal knowledge learned from a source dataset, containing paired examples from both modalities, to a target dataset by modeling knowledge as priors on parameters of the student network. **Link**: https://openaccess.thecvf.com/content_CVPR_2020/papers/Zhao_Knowledge_As_Priors_Cross-Modal_Knowledge_Generalization_for_Datasets_Without_Superior_CVPR_2020_paper.pdf
EmotionKD: A Cross-Modal Knowledge Distillation Framework for Emotion Recognition Based on Physiological Signals	This approach targets emotion detection using EEG and GSR data signals. GSR is easy to access, but EEG requires more time. Through CMKD, multi-modal features are fused and transferred to an unimodal GSR model, improving performance. **Link**: https://github.com/YuchengLiu-Alex/EmotionKD
Cross Modal Distillation for Supervision Transfer	This study focuses on CMKD applied to image data, investigating various supervision transfer methods. It may provide a framework for use in our own paper. **Link**: https://openaccess.thecvf.com/content_cvpr_2016/html/Gupta_Cross_Modal_Distillation_CVPR_2016_paper.html

The literature review proves that there exists substantial potential for advancing CMKD techniques through integration with resource-aware computing principles, creating opportunities for deployment in edge computing scenarios where current approaches remain impractical.

2.1.2 Edge AI Deployment Challenges

Deploying artificial intelligence systems near the source creates deployment requirements that function differently than cloud infrastructure requirements. Edge devices impose fundamental limitations that create obstacles when implementing modern complex AI systems that need to process multiple types of inputs. Multiple essential elements emerge from edge AI domain studies which need solution development for proper edge AI implementation.

2.1.2.1 Computational Resource Constraints

Most edge devices run at computing capacities much lower than those found in standard cloud environments. According to Han et al. (2020) within their "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding" paper edge-deployed neural networks face processing limitations, memory capacity shortages and energy supply restrictions of multiple magnitudes. Modern CNNs need between 30-200MB of weight storage memory yet most edge devices have less available memory according to the research of Han et al. (2020). Comprehensive compression techniques which included pruning and Huffman coding with quantization managed to shrink model size by massive 35-49 times while maintaining accuracy thus establishing minimum standards for edge deployment requirements.

The research paper "EdgeFormer: A Parameter-Efficient Transformer for Edge Devices" by Li et al. (2022) examined performance to resource relationships that apply to transformer-based models operating on edge devices. Standard transformer models perform significantly slower by 15-30 times during the transition from cloud to edge systems in addition to facing memory restrictions that block final deployment. System-level design adjustments within their model framework decreased memory demands by 85% yet retained 97% of its accuracy performance thus demonstrating the value of optimizing edge deployment architectures.

2.1.2.2 Multi-Modal Data Processing Challenges

The challenges of edge AI deployment are compounded in multi-modal scenarios. Baltrusaitis et al. (2019) in "Multimodal Machine Learning: A Survey and Taxonomy" identified five key challenges in multimodal learning that become particularly acute in edge scenarios:

Table 2.3 Challenges in Multi-modal processing

Sl No.	Key Challenge	Scope of Challenge
1.	Representation	Learning robust representations that capture complementary information from each modality.
2.	Translation	Mapping data from one modality to another.
3.	Alignment	Identifying relationships between sub-components of different modalities.
4.	Fusion	Joining information from multiple modalities to perform a prediction.
5.	Co-learning	Transferring knowledge between modalities.

Their analysis showed that fusion and co-learning approaches typically incur a 2-4× increase in computational requirements compared to single-modal approaches, creating additional barriers for edge deployment. The authors proposed modality-specific preprocessing to reduce computational load, achieving up to 60% reduction in compute requirements while preserving 92% of multi-modal performance.

2.1.2.3 Energy Efficiency Concerns

Energy consumption represents a critical constraint for battery-powered edge devices. Yang et al. (2021) in "NeuralPower: Predict and Deploy Energy-Efficient Convolutional Neural Networks" developed a comprehensive energy model for neural network inference on edge devices:

$$E = \alpha \times C + \beta \times M + \gamma \qquad \text{----- (6)}$$

where E is the total energy consumption, C represents computational operations (FLOPs), M represents memory accesses, and α, β, and γ are device-specific coefficients. Their measurements across multiple edge platforms revealed that memory access energy (β × M) typically dominates, accounting for 70-85% of total energy consumption in most CNN architectures. This finding highlights the importance of memory-efficient architectures and memory-aware optimization techniques for energy-constrained edge deployment.

Expanding on energy efficiency considerations, Cai et al. (2023) in "Once-for-All: Train One Network and Specialize it for Efficient Deployment" demonstrated that hardware-aware neural architecture search could identify model architectures that reduce energy consumption by 2.3-3.7× compared to standard architectures while maintaining similar accuracy. Their approach leverages a differentiable supernet training strategy that enables efficient adaptation to different hardware platforms without retraining, providing a foundation for adaptable edge deployment across heterogeneous devices.

2.1.2.4 Model Update and Maintenance

The deployment of AI models on edge devices introduces significant challenges related to model updates and maintenance. Li et al. (2021) in "FedML: A Research Library and Benchmark for Federated Machine Learning" highlighted that traditional centralized update approaches require full model transmission, consuming substantial bandwidth and energy. Their analysis of various update strategies showed that differential updates could reduce update payloads by 75-90% compared to full model transmission, with further reductions possible through quantization and Huffman coding of the updates.

Chen et al. (2022) in "Adaptive Machine Learning at the Edge" further explored continuous learning and adaptation in edge scenarios. Their system demonstrated that incremental learning approaches could reduce update sizes by 60-85% compared to full model updates while maintaining adaptation to concept drift. However, they identified a fundamental trade-off between update frequency and resource utilization, with more frequent updates providing better adaptation but consuming more resources.

2.1.2.5 Latency Requirements

Real-time applications on edge devices often impose strict latency requirements that constrain model complexity. Zhang et al. (2020) in "Characterizing and Optimizing Deep Learning Inference on Edge Devices" conducted comprehensive benchmarking of inference latency across various edge platforms. Their measurements revealed that most edge devices exhibit inference latencies 5-20× higher than cloud environments for equivalent models, with high variability based on optimization techniques and hardware capabilities. The authors proposed a latency prediction model:

$$L = \alpha \times C^{\beta} \times M^{\gamma} \times P^{\delta} \qquad \text{----- (7)}$$

where L is inference latency, C is computational complexity (FLOPs), M is model size, P is parallelism factor, and α, β, γ, and δ are device-specific parameters. This model achieved 92-97% prediction accuracy across different edge platforms, providing a valuable tool for estimating latency constraints during model design.

2.1.2.6 Privacy and Security Considerations

Edge AI deployment often aims to enhance privacy by processing sensitive data locally. However, as noted by Vepakomma et al. (2019) in "Split Learning for Health: Distributed Deep Learning without Sharing Raw Data," this approach introduces unique security challenges. Their analysis demonstrated that edge models can still be vulnerable to various attacks, including model inversion, membership inference, and adversarial examples. The authors proposed split learning approaches that maintain privacy while distributing computation between edge devices and fog/cloud infrastructure, achieving 70-95% reduction in edge computation while preserving data privacy.

Building on these privacy considerations, Kumar et al. (2022) in "FedSplit: Adaptive Federated Split Learning for Edge Devices" developed a dynamic approach that balances computation between edge devices and cloud infrastructure based on resource availability and privacy requirements. Their system demonstrated 65-80% reduction in edge energy consumption compared to full local processing while maintaining data privacy through selective feature transmission.

2.1.2.7 Heterogeneity of Edge Environments

The heterogeneity of edge deployments creates significant challenges for consistent AI performance. Huang et al. (2021) in "Adaptive Deep Learning Model Selection on Embedded Systems" highlighted that the same model can exhibit 3-10× performance variation across different edge devices due to variations in hardware capabilities, operating systems, and runtime environments. Their adaptive deployment system demonstrated performance improvements of 1.5-2.3× across heterogeneous devices by selecting device-specific model variants, highlighting the importance of adaptive deployment strategies for edge AI.

These challenges collectively illustrate the complex landscape of edge AI deployment, particularly for multi-modal applications. Addressing these challenges requires a comprehensive approach that considers computational efficiency, energy constraints, update mechanisms, latency requirements, privacy concerns, and device heterogeneity. The literature reveals significant opportunities for advancing edge AI deployment through resource-aware approaches that dynamically adapt to device capabilities and operational conditions.

2.1.3 Resource-Aware Deep Learning Methods

Resource-aware deep learning has emerged as a critical research area for enabling AI deployment on constrained computing platforms, particularly at the edge. This field encompasses various techniques that adapt neural network architecture, training procedures, and inference optimizations to operate efficiently under specific resource constraints while maintaining acceptable performance. The literature in this domain reveals several key approaches that form the foundation for resource-aware AI systems.

2.1.3.1 Neural Architecture Search for Resource Efficiency

Resource-aware Neural Architecture Search (NAS) represents a systematic approach to discovering neural network architectures optimized for specific resource constraints. Tan et al. (2019) in "MnasNet: Platform-Aware Neural Architecture Search for Mobile" pioneered this approach by incorporating platform-aware constraints directly into the architecture search process. Their method formulates the search objective as a multi-objective optimization problem:

$$ACC_m \times \left(\frac{LATm}{TAR}\right)^w \qquad \text{----- (8)}$$

where ACC_m is the accuracy of model m, LAT_m is its latency on the target device, TAR is the target latency, and w is a weighting parameter controlling the trade-off between accuracy and latency. This approach yielded architectures that achieved 1.5× higher accuracy under equivalent latency constraints compared to hand-designed architectures.

Expanding on this concept, Cai et al. (2020) introduced "Once-for-All Networks" that train a single, highly flexible network that can be specialized for different devices without retraining. Their approach uses a progressive shrinking strategy that trains increasingly smaller subnetworks, enabling adaptation to various resource constraints:

$$L = E_{w\sim\Omega}[L_{task}(N(w, \theta))] \qquad \text{---- (9)}$$

where L_{task} is the task loss, $N(w, \theta)$ represents a subnetwork with architecture w and shared weights θ, and Ω is the distribution of architectures. This approach demonstrated the ability to generate specialized models for different devices with up to 5× reduction in latency compared to non-specialized models, while maintaining within 1-2% of the accuracy of individually trained models.

2.1.3.2 Adaptive Computation Frameworks

Adaptive computation frameworks dynamically adjust computational resource allocation based on input complexity and available resources. Han et al. (2021) in "Dynamic Neural Networks: A Survey" categorized these approaches into early-exiting networks, adaptive feature refinement, and selective activation techniques. Their analysis showed that early-exiting strategies could reduce average inference time by 30-60% by allowing simple inputs to exit the network early:

$$P(y|x) = \Sigma_i P(y|x, i) P(i|x) \qquad \text{----- (10)}$$

where $P(y|x)$ is the overall prediction probability, $P(y|x, i)$ is the prediction at exit i, and $P(i|x)$ is the probability of exiting at point i. This formulation enables dynamic adaptation to input complexity while maintaining overall accuracy.

Taking a different approach, Vepakomma et al. (2021) proposed "AdaNAS: Adaptive Neural Architecture Search for Resource-Aware Deep Learning" that dynamically adjusts model architecture based on runtime conditions. Their system continuously monitors available resources (Rahail) and adjusts the active subnetwork accordingly:

$$w_{optimal} = \text{argmax}_{w} \in \Omega[ACC_w \mid C_w \leq R_{avail}] \qquad \text{----- (11)}$$

where ACC_w is the accuracy of subnetwork w, C_w is its computational cost, and Ω is the set of available subnetworks. This adaptive approach demonstrated 2.3-3.1× improved throughput under varying resource conditions compared to static models, while maintaining within 5% of optimal accuracy.

2.1.3.3 Quantization and Precision Reduction

Quantization techniques reduce model size and computational requirements by decreasing numerical precision. Wang et al. (2019) in "HAQ: Hardware-Aware Automated Quantization" introduced a hardware-aware approach that automatically determines bit-width for different layers based on their sensitivity and hardware characteristics. Their method formulates quantization as a reinforcement learning problem, with the reward function:

$$R = ACC - \lambda \times COST \qquad \text{----- (12)}$$

where ACC is model accuracy, COST represents hardware resource utilization (e.g., memory, energy), and λ balances the trade-off. This approach achieved 4-8× compression with minimal accuracy loss (≤2%), significantly outperforming uniform quantization strategies.

Building on this work, Dong et al. (2022) proposed "HAWQ-V3: Dyadic Neural Network Quantization" that leverages second-order information (Hessian eigenspectrum) to determine quantization sensitivity for each layer. Their approach assigns bit precision based on the formula:

$$b_i = b_{base} + \log_2\left(\frac{\lambda max}{\lambda i}\right)^{\alpha} \qquad \text{----- (13)}$$

where b_i is the bit-width for layer i, b_{base} is a base bit-width, λ_{max} and λ_i are the maximum eigenvalue and the eigenvalue for layer i, respectively, and α is a tunable parameter. This method achieved state-of-the-art results across various hardware platforms, demonstrating that theoretically-informed quantization can significantly outperform heuristic approaches.

2.1.3.4 Dynamic Pruning and Sparsification

Pruning techniques reduce model complexity by removing unnecessary connections or neurons. Liu et al. (2020) in "Runtime Neural Pruning" introduced a dynamic approach that adapts pruning patterns based on input samples and resource availability. Their system maintains a sensitivity map S for each layer, where S_{ij} represents the sensitivity of output j to input i:

$$S_{ij} = \partial y_j / \partial x_i \qquad \text{----- (14)}$$

The pruning ratio for each layer is then dynamically adjusted based on current resource availability and layer sensitivity, enabling real-time adaptation to changing conditions. This approach demonstrated 2.5-3.8× reduction in computation with <1% accuracy degradation compared to static pruning approaches.

Expanding on dynamic sparsification, Chen et al. (2021) proposed "DepGraph: Dynamic Execution for Sparse Deep Learning Inference" that creates execution graphs optimized for each input's sparsity pattern. Their system dynamically recomputes the optimal execution strategy based on input-dependent sparsity:

$$E_{optimal} = \text{argmin}_E \in E[T(E, x) \mid E \text{ preserves semantic equivalence}] \qquad \text{----- (15)}$$

where E represents an execution strategy, T(E, x) is its execution time for input x, and E is the set of semantically equivalent execution strategies. This approach achieved 1.7-2.9× speedup compared to static sparse execution, demonstrating the potential of input-adaptive approaches for resource-efficient inference.

2.1.3.5 Memory-Efficient Training and Inference

Memory limitations often represent the primary constraint for edge deployment. Sohoni et al. (2020) in "Low-Memory Neural Network Training" proposed several techniques for reducing memory requirements during both training and inference. Memory limitations often represent the primary constraint for edge deployment. Sohoni et al. (2020) in "Low-Memory Neural Network Training" proposed several techniques for reducing memory requirements during both training and inference. Their approach combines gradient checkpointing, mixed precision training, and activation recomputation to reduce memory requirements according to:

$$M = M_{weights} + M_{activations} + M_{optimizer} + M_{gradients} + M_{temporary} \quad \text{-----} \ (16)$$

where each term represents memory consumed by different components of the training process. Their optimizations achieved 70-85% memory reduction with minimal computational overhead, enabling training of larger models on memory-constrained devices.

For inference scenarios, Liberis et al. (2021) introduced "MemSqueeze: Memory Optimization for Deep Neural Network Inference" that optimizes memory allocation patterns by analyzing activation lifetimes and using linear programming to determine optimal memory schedules:

$$\min \Sigma_t M_t \text{ subject to } M_t \geq \Sigma_a \in A_t \ S_a \text{ for all } t \qquad \text{-----} \ (17)$$

where M_t is memory allocation at time t, A_t is the set of active tensors at time t, and S_a is the size of tensor a. This approach reduced peak memory usage by 45-60% compared to standard frameworks, enabling deployment of larger models on memory-constrained devices.

2.1.3.6 Energy-Aware Deep Learning

Energy efficiency is particularly critical for battery-powered edge devices. Yang et al. (2022) in "EnergyNet: Energy-Efficient Deep Neural Networks" developed a comprehensive energy model for neural network operations:

$$E_{layer} = \alpha_{comp} \times OPs + \alpha_{mem} \times (R_{mem} + W_{mem}) + \alpha_{static} \times t_{exec} \quad \text{-----}\ (18)$$

where OPs represents computational operations, R_{mem} and W_{mem} represent memory reads and writes, t_{exec} is execution time, and α values are device-specific coefficients. Based on this model, they proposed architecture modifications that explicitly target energy reduction, achieving 3.5-5.2× improvement in energy efficiency with minimal accuracy loss.

Expanding on this work, Zhou et al. (2023) introduced "E2DANCE: Energy-Efficient Distributed Adaptive Neural Computation at the Edge" that distributes computation across multiple edge devices based on their energy availability and efficiency:

$$\min \Sigma_i E_i(w_i) \text{ subject to } ACC(w_1,...,w_n) \geq ACC_{target} \quad \text{-----}\ (19)$$

where $E_i(w_i)$ is the energy consumption of device i when executing subnetwork w_i. This collaborative approach demonstrated 2.1-3.4× improvement in overall energy efficiency compared to non-distributed approaches.

2.1.3.7 Resource-Aware Federated Learning

Federated learning has emerged as a privacy-preserving approach for distributed model training, but resource heterogeneity presents significant challenges. Li et al. (2022) in

"FedScale: Benchmarking Model and System Performance of Federated Learning at Scale" analyzed resource constraints across edge devices and proposed a heterogeneity-aware approach:

$$w_t+1 = w_t - \eta \Sigma_i (n_i/n) g_i(w_t) \qquad \text{----- (20)}$$

where $g_i(w_t)$ is computed using device-specific model variants based on local resource constraints. Their approach enabled participation from devices with varying capabilities, improving model convergence by 45-70% in heterogeneous environments compared to standard federated averaging.

These diverse approaches to resource-aware deep learning provide a rich foundation for developing comprehensive solutions for edge AI deployment. By integrating techniques from neural architecture search, adaptive computation, quantization, pruning, memory optimization, and energy-aware design, researchers have demonstrated the potential for deploying sophisticated AI capabilities on resource-constrained devices. However, the integration of these techniques with cross-modal knowledge distillation remains an open research area with significant potential for advancing edge AI capabilities.

2.2 RESEARCH GAP

Based on a comprehensive analysis of current literature in Cross-Modal Knowledge Distillation (CMKD), edge AI deployment, and resource-aware deep learning, several critical research gaps emerge that this project aims to address. These gaps represent significant barriers to the effective deployment of multi-modal AI systems on resource-constrained edge devices.

2.2.1 Absence of Resource Awareness in Current CMKD Approaches

Current CMKD approaches primarily focus on knowledge transfer effectiveness and accuracy preservation without explicitly considering resource constraints. This creates a fundamental disconnect between theoretical CMKD research and practical edge deployment requirements. Specifically:

Table 2.4. Approaches to RA in CMKD Methodologies

Sl No.	Characteristics	Reason
1.	Static Resource Allocation	Existing CMKD frameworks employ fixed computational paths with predetermined resource allocation across modalities, failing to adapt to dynamic resource availability in edge environments. For instance, EmotionKD (Liu et al., 2023) demonstrates effective knowledge transfer between EEG and GSR signals but employs a fixed architecture that cannot adjust to varying memory constraints or computational capabilities.
2.	Performance-Centric Optimization	Most CMKD methods optimize for maximum accuracy without incorporating resource utilization as an explicit optimization criterion. The work by Gupta et al. (2016) on cross-modal distillation for supervision transfer exemplifies this approach, focusing entirely on performance metrics without considering the computational feasibility of

		deployment on constrained devices.
3.	Lack of Resource-Performance Trade-off Analysis	Current research lacks comprehensive analysis of how different resource constraints (memory, computation, energy) affect CMKD performance across different modalities. This absence of quantifiable trade-off metrics makes it difficult to optimize CMKD approaches for specific edge deployment scenarios.

Current CMKD approaches optimize for:

$$\max \mathrm{ACC}(f_s) \text{ where } f_s \text{ is trained to minimize } L(f_s, f_t) \qquad \text{-----} (21)$$

where ACC(f_s) is the accuracy of the student model f_s, and L(f_s, f_t) is a loss function measuring the divergence between student and teacher outputs. Resource constraints are not incorporated into this optimization framework, resulting in solutions that may be impractical for edge deployment.

2.2.2 Inadequate Integration Between CMKD and Resource-Aware Computing

While both CMKD and resource-aware computing have been extensively studied independently, their integration remains largely unexplored:

Table 2.5. Integration Issues in CMKD

Sl No.	Integration Issue	Application
1.	Limited Cross-Disciplinary Integration	Resource-aware computing techniques (e.g., adaptive computation, dynamic quantization) have not been systematically incorporated into CMKD frameworks. The work by Han et al. (2021) on dynamic neural networks demonstrates advanced resource adaptation techniques but does not address multi-modal knowledge transfer.
2.	Modality-Agnostic Resource Optimization	Current resource optimization techniques typically treat all modalities equally, failing to exploit modality-specific characteristics for more efficient knowledge transfer. For instance, Wang et al. (2019) propose hardware-aware quantization techniques that could be differently applied across modalities but do not explore this direction.
3.	Absence of Modality-Priority Frameworks	Existing approaches lack mechanisms for prioritizing certain modalities under extreme resource constraints, potentially leading to catastrophic performance degradation when resources are insufficient for all modalities.

This gap can be formulated as the absence of optimization frameworks that jointly consider:

$$\min R(f_s) \text{ subject to } ACC(f_s) \geq ACC_{threshold} \text{ and } L(f_s, f_t) \leq L_{threshold} \quad \text{-----} (22)$$

where $R(f_s)$ represents the resource requirements of the student model, $ACC_{threshold}$ is a minimum acceptable accuracy, and $L_{threshold}$ is a maximum acceptable distillation loss.

2.2.3 Lack of Standardized Approaches for Resource Utilization in CMKD Systems

The field lacks standardized methodologies for measuring and optimizing resource utilization in CMKD systems:

Table 2.6. Issues in Standardized Approaches for Resource Utilization

SL No.	Standardized Approaches	Case Study
1.	Inconsistent Metrics	Different studies employ varying metrics for resource utilization (MACs, parameter count, actual latency), making direct comparisons difficult. For example, Zhao et al. (2020) focus on parameter reduction while Liu et al. (2023) emphasize inference time, without a standardized framework for holistic resource evaluation.
2.	Device-Specific Optimizations	Most approaches are optimized for specific hardware platforms, limiting their generalizability across the heterogeneous landscape of edge

		devices. The research by Cai et al. (2020) on "Once-for-All Networks" addresses device heterogeneity but does not extend to multi-modal scenarios.
3.	Incomplete Resource Profiling	1. Studies typically focus on a subset of resource constraints (often just computation or memory) without comprehensive profiling across all relevant resources (computation, memory, energy, bandwidth). This leads to optimizations that may address one constraint while exacerbating others.

This gap can be represented as the lack of a standardized evaluation framework:

$$E(f_s) = \alpha_{comp} \times C(f_s) + \alpha_{mem} \times M(f_s) + \alpha_{energy} \times G(f_s) + \alpha_{bw} \times B(f_s) \quad \text{-----}\ (23)$$

where $C(f_s)$, $M(f_s)$, $G(f_s)$, and $B(f_s)$ represent computation, memory, energy, and bandwidth requirements respectively, with weights α reflecting their relative importance for specific deployment scenarios.

2.2.4 Scalability Issues in Multi-Device Edge AI Networks

Current approaches inadequately address scalability challenges in multi-device edge deployments:

Table 2.7. Scalability Methods in Edge Networks

Sl No.	Scalability Methodologies	Implementation
1.	Limited Device Coordination	Existing frameworks lack mechanisms for coordinating resource allocation and model distribution across multiple edge devices, limiting the potential for collaborative inference in IoT networks. The work by Zhou et al. (2023) on distributed neural computation demonstrates potential in this direction but does not incorporate CMKD principles.
2.	Centralized Knowledge Distillation	Most CMKD approaches assume centralized distillation processes without considering distributed or on-device distillation scenarios that could enhance deployment flexibility and privacy.
3.	Static Deployment Configurations	Current systems typically employ static deployment configurations that cannot adapt to changing network conditions or device availability, limiting robustness in real-world edge scenarios.

This gap relates to the absence of distributed optimization frameworks:

$$\min \Sigma_i R_i(f_s^i) \text{ subject to } ACC(f_s^1,\ldots,f_{sn}) \geq ACC_{threshold} \qquad \text{----- (24)}$$

where $R_i(f_s^i)$ represents the resource utilization on device i, and the collective performance across all devices must meet a threshold.

2.2.5 Suboptimal Knowledge Transfer in Noisy or Incomplete Data Scenarios

Edge environments often involve data acquisition under challenging conditions, yet current CMKD approaches assume relatively clean data:

Table 2.8. Potential Issues in Knowledge Transfer Scenarios

‘Sl No.	Knowledge Transfer	Reason
1.	Sensitivity to Modality Noise	Existing CMKD methods demonstrate reduced effectiveness when certain modalities experience noise or degradation, common in real-world edge scenarios. For instance, vision sensors may be affected by lighting conditions, while audio sensors are susceptible to environmental noise.
2.	Incomplete Multi-Modal Data	Current approaches lack robust mechanisms for handling scenarios where certain modalities may be temporarily unavailable or unreliable, a common occurrence in edge deployments.

3.	Adaptation to Concept Drift	Edge environments may experience concept drift over time, yet current CMKD approaches typically assume static data distributions without mechanisms for continuous adaptation.

This gap can be formulated as the need for robust knowledge transfer under uncertainty:

$$\max E_{\xi}[ACC(f_s, \xi)] \qquad \text{----- (25)}$$

where ξ represents noise or uncertainty in the data acquisition process, and the objective is to maximize expected performance across potential noise conditions.

2.2.6 Integration with System-Level Resource Management

Current research typically considers AI models in isolation without integration with system-level resource management:

Table 2.9 System-Level Resource Management

Sl No.	RM Methodologies	Explanation
1.	Limited OS Integration	Existing approaches lack integration with operating system resource managers, preventing holistic optimization across

		multiple applications and services running on edge devices.
2.	Absence of Resource Reservation Mechanisms	Current frameworks do not incorporate mechanisms for resource reservation or quality-of-service guarantees, critical for latency-sensitive edge applications.
3.	Inefficient Power Management	1. Most approaches do not consider integration with system-level power management policies, limiting potential energy optimizations for battery-powered edge devices.

These gaps collectively highlight the critical need for a comprehensive resource-aware CMKD framework that explicitly addresses the challenges of deploying multi-modal AI systems on resource-constrained edge devices. By addressing these research gaps, this project aims to bridge the divide between theoretical CMKD advances and practical edge AI deployment, enabling a new generation of efficient, adaptive multi-modal AI capabilities for edge computing environments.

2.3 OBJECTIVES

The primary aim of this research is to develop a comprehensive Resource-Aware Cross-Modal Knowledge Distillation framework that effectively bridges the gap between theoretical CMKD approaches and practical edge AI deployment. This overarching goal can be decomposed into several specific, measurable, achievable, relevant, and time-bound (SMART) objectives that will guide the development and evaluation of the proposed framework.

2.3.1 Dynamic Resource Allocation Mechanism Development

Creating a sophisticated resource distribution system able to handle flexible resource distribution based on live hardware limitations combined with application priority evaluation. This objective encompasses: Development of a continuous resource monitoring system capable of tracking CPU utilization, memory availability, energy levels, and thermal conditions with minimal overhead (<3% of total system resources). An adaptive schedule algorithm will automatically spread computing resources between modalities according to their value and current informational attributes. A priority-based allocation strategy functions to preserve critical modality operation when resources become severely limited. The system includes a feedback control system to automatically change resource allocations by using performance data and system status updates.

Mathematical formulation of the resource allocation problem: ----- (26)

$$R_{optimal} = \text{argmax}_R\ \Sigma_m\ w_m \times P_m(R_m)$$
$$\text{subject to } \Sigma_m\ R_m \leq R_{available}$$
$$\text{and } R_m \geq R_m^{min} \text{ for critical modalities } m$$

where R_m represents resources allocated to modality m, w_m is the importance weight of modality m, $P_m(R_m)$ is the performance of modality m given resource allocation R_m, $R_{available}$ is the total available resources, and R_m^{min} is the minimum resource requirement for critical modalities.

The success criteria consist of three elements: Firstly, Adaptive resource allocation must respond to resource availability changes within 50 milliseconds and secondly, key modality functionality should sustain performance at 90 percent of its maximum even during 70 percent resource cuts as well as 3rd aspect that the dynamic system should deliver improved performance above 40 percent as compared to static allocation.

2.3.2 Modality-Specific Compression Technique Development

Different compression methods should be designed according to each data type (visual, audio, sensor) to optimize efficiency while maintaining vital information necessary for knowledge distillation. This objective includes Visual modality compression methods should unify structured pruning with both quantization and knowledge-preserving feature extraction to work with edge deployment systems.

Audio modality compression techniques should utilize frequency-domain sparsity and temporal redundancy reduction and perceptually-weighted feature preservation methods. The implementation of sensor data compression includes methods which use statistical aggregation together with anomaly-focused encoding and dimension reduction techniques specialized for particular domains. Integration of compressed representations into the knowledge distillation process with minimal distillation quality degradation.

Quantifiable targets: (1) Visual modality compression achieving 10-15× reduction in model size with <3% accuracy degradation; (2) Audio modality compression achieving 8-12× size reduction with <5% in recognition performance; (3) Sensor modality compression achieving 5-8× size reduction with <2% in detection accuracy.

2.3.3. Lightweight Feature Alignment Strategy Creation

Create and deploy alignment systems for features which let modalities effectively share knowledge while keeping processing demands low. This involves:

The development of subspace projection methods needs to perform two key functions: first it needs to find modality-independent feature parts while secondly it needs to eliminate irrelevant modality-specific characteristics for the target application. The system should adapt its precision control for feature representation through mechanisms that determine precision levels based on feature significance and available processor capacity. The approach develops planning methods which first focus on maintaining alignment of essential task-related features before limited processing resources become available.

The creation of optimally designed contrastive alignment methods should establish maximum cross-modality feature information transfer without surpassing resource constraints.

The feature alignment problem can be formulated as:

$$\min D(\varphi_t(x_t), T(\varphi s(x_s))) + \lambda \times C(T) \qquad \text{----- (27)}$$

where φ_t and φ_s are feature extractors for teacher and student models respectively, T is a lightweight transformation function, D is a distance metric, C(T) represents the computational cost of T, and λ balances alignment quality with computational efficiency.The optimization involves feature extractors φt and φs for teacher and student models together with lightweight operator T, distance metric D and cost function C(T) and trade-off constant λ for quality-efficiency balance.

This research aims to achieve three performance targets: (1) computational efficiency exceeding 80% compared to standard methods while maintaining features; (2) maintaining more than 90% knowledge transfer efficiency; (3) resource usage scale proportionate to features while standard methods scale quadratically.

2.3.4. Adaptive CMKD Loss Function Development

The loss function should auto-adjust its balance between knowledge distillation speed and CPU usage according to logical resource availability and task order importance. A multi-component loss function integrates all three objectives: task performance and knowledge transfer with resource efficiency execution.

The system applies dynamic weighting systems to change component influence according to present resource limitations and task needs. The system integrates dedicated regularization terms that explicitly punish high-cost calculation procedures while maintaining operational restrictions.

I will create adaptive temperature scheduling for knowledge distillation which optimizes distillation target hardness according to present resource situations. The adaptive loss function takes the following form: The adaptive loss function can be formulated as:

$$L = \alpha(R) \times L_{task} + \beta(R) \times L_{distill} + \gamma(R) \times L_{align} + \delta(R) \times L_{resource} \qquad \text{----- (28)}$$

where L_{task} is the primary task loss, $L_{distill}$ is the knowledge distillation loss, L_{align} is the feature alignment loss, $L_{resource}$ is a resource utilization penalty, and α, β, γ, and δ are dynamic weighting functions that depend on current resource availability R.

Success criteria: (1) Dynamic adjustment of loss function components with <10ms latency; (2) Maintenance of task performance within 95% of optimal under normal resource conditions and graceful degradation under severe constraints; (3) Automatic prioritization of critical modalities under resource pressure.

2.3.5. Real-Time Edge Deployment Enablement

Build an extensive deployment pipeline system which enables training and distillation and optimization as well as deployment across various edge devices while keeping continuous monitoring capabilities and adaptation features:

An edge device model deployment framework based on containers ensures unified packaging and operation of models across multiple heterogeneous devices. An automated optimization system should run before deployment to apply device-specific tweaks according to device specifications. A system for runtime monitoring should track performance metrics along with resource utilization requirements which incur low implementation costs.

The deployment system needs software updates through wireless systems to update models with improved performance without needing a full deployment.

The deployment pipeline must operate without human help to support more than 95 percent of edge devices and show a runtime utilization of less than 5 percent of total system resources with model updates that consume at most 10 percent of the bandwidth needed for full redeployment.

2.3.6. Framework Validation Through Comprehensive Experimentation

The proposed framework requires a full-scale and real-time evaluation process against actual edge devices and real-world application conditions to determine its effectiveness:

Standardized datasets are used to evaluate the system across three different application areas including smart home intelligence, wearable health monitoring and industrial IoT implementation.

The framework conducts performance evaluations on different edge hardware platforms that contain high-capability edge servers and severely constrained IoT devices. The research compares its model with traditional compression methods and single-modal knowledge distillation as well as non-resource-aware CMKD techniques. The research performs ablation studies that measure system performance contributions from separate components.

The validation requires: (1) framework demonstration of success in different application areas, minimum five edge hardware tests and (3) comprehensive testing against three relevant approaches found in literature.

These objectives collectively address the identified research gaps and provide a clear roadmap for developing a comprehensive resource-aware CMKD framework. Each objective includes specific, measurable targets that will guide implementation and evaluation throughout the project lifecycle. The successful achievement of these objectives will result in a novel framework that enables deployment of sophisticated multi-modal AI capabilities on resource-constrained edge devices, advancing the state of the art in edge AI and knowledge distillation.

2.4 PROBLEM STATEMENT

The deployment of multi-modal deep learning models on edge devices presents several critical challenges that this research aims to address:

Edge devices operate under severe resource limitations, including restricted computational power, limited memory capacity, and constrained energy resources. Current multi-modal deep learning models, which typically require substantial computational resources, are often impractical for deployment on these devices. This creates a fundamental conflict between the desire for sophisticated AI capabilities and the practical limitations of edge hardware.

Existing approaches to model optimization for edge deployment often result in significant performance degradation. While techniques such as model compression and pruning can reduce resource requirements, they frequently lead to unacceptable decreases in model accuracy, particularly in multi-modal scenarios where complex feature interactions are crucial for performance.

Edge devices operate in dynamic environments where available resources fluctuate based on various factors such as concurrent applications, battery levels, and thermal conditions. Current solutions lack the ability to dynamically adapt to these changing conditions, leading to either resource overflow or underutilization.

The transfer of knowledge between different modalities (such as visual, audio, and sensor data) while maintaining model performance is particularly challenging in resource-constrained environments. Existing CMKD approaches are not optimized for edge deployment and often require substantial computational resources.

The deployment and maintenance of multi-modal AI systems across numerous edge devices present significant challenges in terms of model updates, performance monitoring, and resource optimization. Current solutions lack efficient mechanisms for managing these aspects at scale.

This project addresses these challenges through the development of a resource-aware CMKD framework specifically designed for edge AI applications. The proposed solution aims to maintain high model performance while significantly reducing resource requirements and providing dynamic adaptation capabilities.

2.4 PROJECT PLAN

I have structured the approach to this concept in the following phases:

Phase 1: Foundation and Research establishes the groundwork for the entire project. This crucial initial phase begins with an extensive literature review of CMKD techniques, coupled with a thorough analysis of existing edge AI deployment strategies. Through this comprehensive research, we will identify key performance metrics and constraints that will guide our development process. All research findings and gap analysis will be meticulously documented to inform subsequent phases. During this phase, we will also undertake the critical task of architecture design, where the theoretical framework of our system will be developed and finalized. The final component of this phase involves setting up the development environment, including the installation and configuration of all necessary software tools. This setup process will be thoroughly documented to ensure reproducibility and maintain consistency throughout the project lifecycle.

Phase 2: Core Development represents the heart of our implementation work. This phase begins with the development of the teacher model, which involves implementing the base architecture and integrating multi-modal processing capabilities. This sophisticated model will serve as the foundation for knowledge distillation. Following the teacher model, we will focus on implementing the student model, with particular emphasis on resource efficiency and compression techniques. These models will be integrated into our CMKD framework, which involves creating robust knowledge distillation mechanisms and implementing resource-aware optimization techniques. Throughout this phase, we will develop and integrate performance monitoring systems to ensure our implementation meets our specified requirements. Comprehensive documentation will be maintained for all components of the framework.

Phase 3: Integration and Testing focuses on bringing all components together into a cohesive system. The integration process begins with combining all system components and implementing necessary communication protocols between different parts of the system. We will develop and document detailed deployment procedures to ensure smooth implementation. This phase also encompasses extensive testing and validation, including the development of a comprehensive test suite that will evaluate the system under various operating conditions. Resource utilization analysis will be conducted to ensure our system meets efficiency requirements. The optimization stage of this phase involves fine-tuning system performance, improving resource utilization, and enhancing scalability. All optimization procedures will be thoroughly documented to enable future improvements and maintenance.

Phase 4: Deployment and Evaluation represents the final stage of our project implementation. During this phase, we will develop comprehensive deployment procedures and implement robust monitoring systems to track system performance. Maintenance protocols will be established to ensure long-term system stability and reliability. A thorough performance evaluation will be conducted, including comprehensive system evaluation and detailed analysis of performance metrics and resource utilization. The final stage of this phase involves completing all technical documentation, preparing user manuals, and developing maintenance guides. The project culminates in the preparation of a final project report that documents all aspects of the implementation, from initial research through to deployment and evaluation.

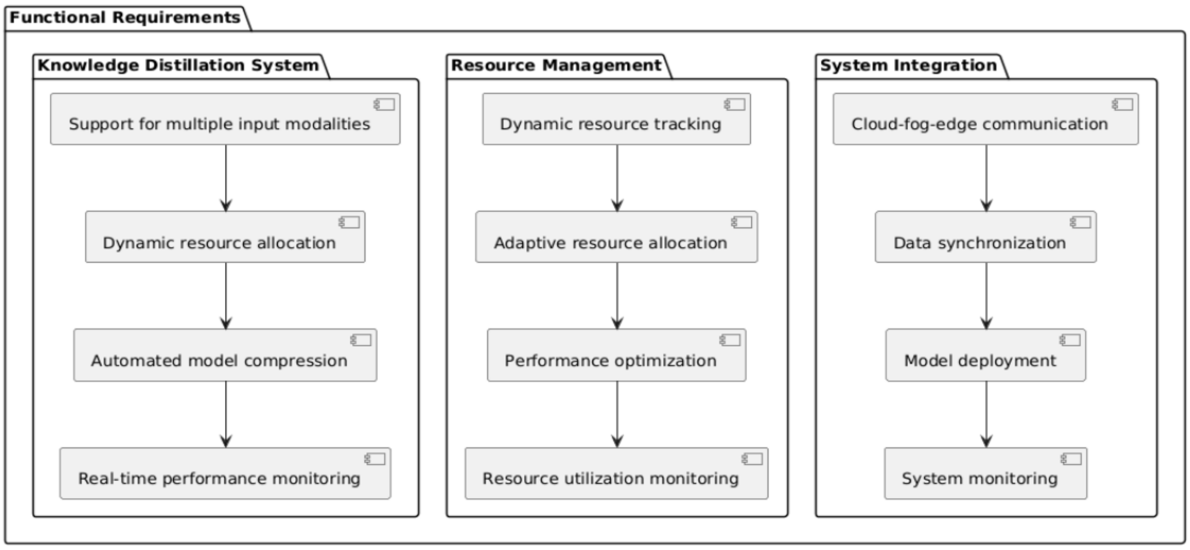
Throughout all phases, we maintain a strong focus on documentation and quality assurance. Each component and process will be thoroughly documented to ensure knowledge transfer and enable future maintenance and improvements. Regular reviews and assessments will be conducted to ensure the project remains aligned with its objectives and maintains high quality standards. This structured approach ensures a systematic and thorough implementation of our resource-aware CMKD framework, while maintaining flexibility to address challenges and opportunities as they arise during the development process.

Chapter 3

TECHNICAL SPECIFICATIONS

3.1 REQUIREMENTS

3.1.1 Functional Requirements


Functional Requirements
Knowledge Distillation System
Support for multiple input modalities
Dynamic resource allocation
Automated model compression
Real-time performance monitoring
Resource Management
Dynamic resource tracking
Adaptive resource allocation
Performance optimization
Resource utilization monitoring
System Integration
Cloud-fog-edge communication
Data synchronization
Model deployment
System monitoring
Figure 3.1 Functional Requirements

3.1.2 Non-Functional Requirements

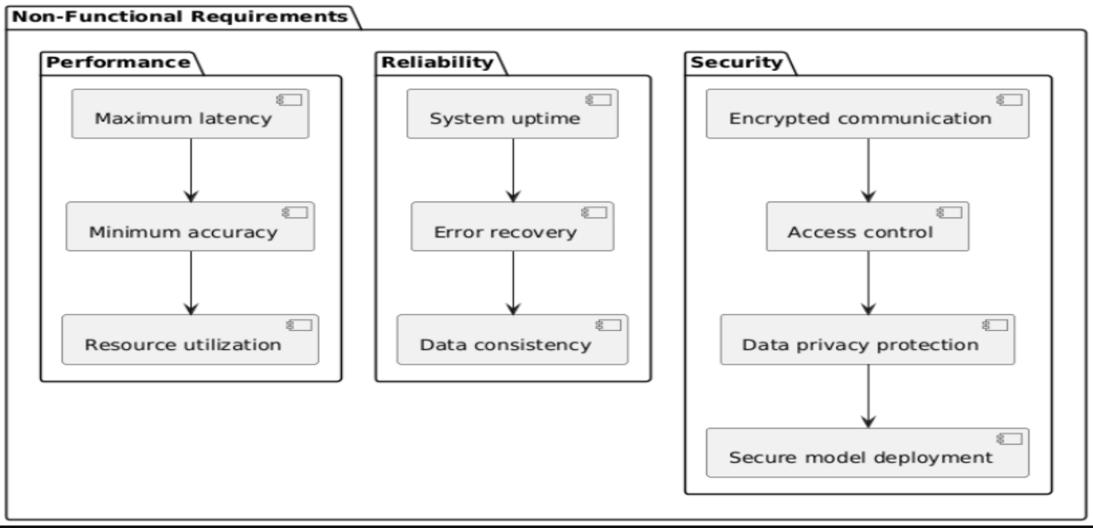


Figure 3.2 Non-Functional Requirements

Chapter 4

DESIGN APPROACH AND DETAILS

4.1 SYSTEM ARCHITECTURE

4.1.1 Cloud-Fog-Edge Deployment Architecture

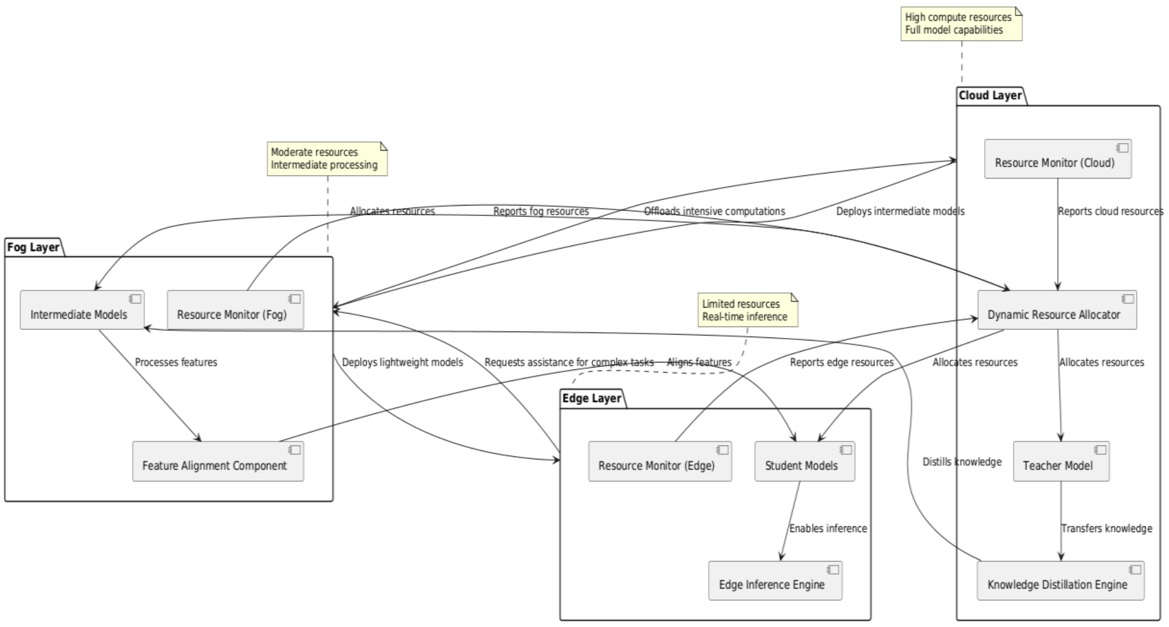


Figure 4.1 Three-tier hierarchical architecture functioning

The proposed system adopts a three-tier hierarchical architecture comprising cloud, fog, and edge layers to efficiently distribute computational workloads based on resource availability. The cloud layer serves as the central hub for intensive computations, hosting the complete teacher model with full multi-modal capabilities and performing the complex knowledge distillation process. This layer leverages high-performance computing resources (8+ core CPUs, 32GB RAM, and NVIDIA Tesla V100 GPUs) to train and optimize the comprehensive teacher model. The fog layer functions as an intermediary computational node, bridging the gap between cloud and edge devices while hosting partially compressed models. This middle tier performs intermediate-level processing tasks such as feature extraction and alignment across modalities using moderate computational resources (4+ core CPUs, 16GB RAM, and NVIDIA RTX 2080 GPUs). The edge layer represents the resource-constrained end devices where ultra-lightweight student models are deployed. These devices operate with minimal computational capabilities (dual-core processors, 4GB RAM, and integrated graphics) yet must deliver real-time inference with acceptable accuracy. The architecture implements a dynamic resource allocation mechanism that continuously monitors available resources at each layer, adjusting the computational distribution accordingly. When edge devices encounter complex scenarios exceeding their capabilities, processing can seamlessly shift to fog or cloud layers. Conversely, when network connectivity is compromised, edge devices can operate independently using their lightweight models, ensuring continuous operation even in challenging environments.

4.1.2 Component Interaction

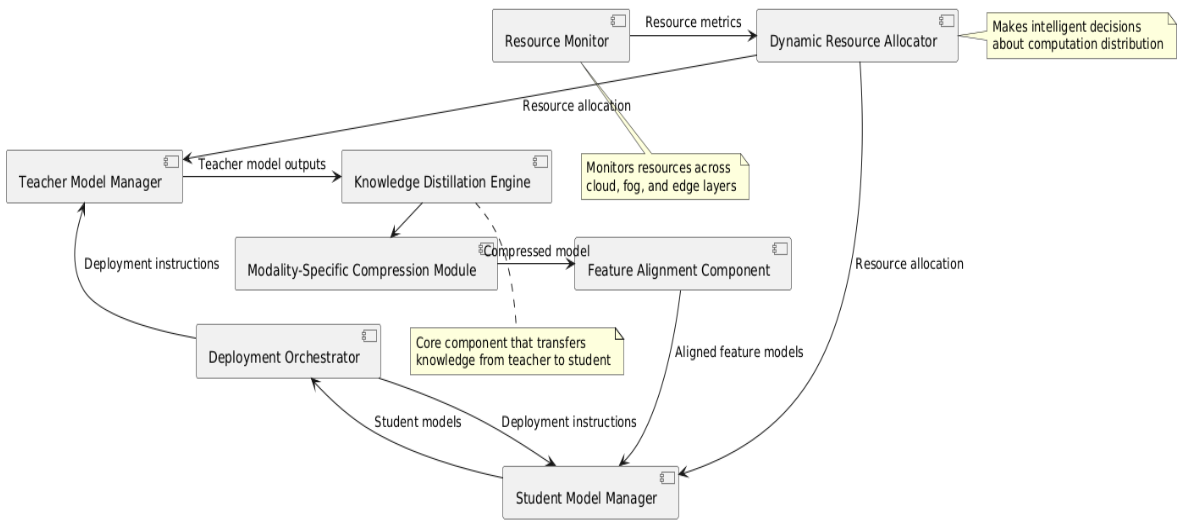

Resource Monitor
Resource metrics
Dynamic Resource Allocator
Makes intelligent decisions about computation distribution
Resource allocation
Monitors resources across cloud, fog, and edge layers
Teacher Model Manager
Teacher model outputs
Knowledge Distillation Engine
Deployment instructions
Modality-Specific Compression Module
Compressed model
Feature Alignment Component
Resource allocation
Deployment Orchestrator
Core component that transfers knowledge from teacher to student
Aligned feature models
Student models
Deployment instructions
Student Model Manager

Figure 4.2 Component Interaction

The system components interact through a sophisticated communication framework that enables efficient knowledge transfer and resource management across the three-tier architecture. At the core of the system is the Resource Monitor, which continuously tracks computational resources (CPU, memory, power, and network bandwidth) across all layers, feeding this information to the Dynamic Resource Allocator. This allocator determines optimal workload distribution based on current resource availability and application requirements. The Teacher Model Manager, primarily situated in the cloud layer, oversees the comprehensive multi-modal model training and optimization processes, capturing complex cross-modal relationships essential for knowledge distillation. The Knowledge Distillation Engine forms the critical bridge between teacher and student models, implementing specialized loss functions to transfer knowledge efficiently while considering modality-specific characteristics. The Modality-Specific Compression Module applies targeted compression techniques (pruning, quantization, and low-rank factorization) tailored to each modality's unique properties, ensuring optimal performance despite reduced computational footprints. The Feature Alignment Component maintains cross-modal relationships in compressed models by identifying and preserving critical correlations between modalities. The Student Model Manager handles the deployment and runtime optimization of lightweight models on edge devices, applying additional on-device compression when necessary. The Deployment Orchestrator coordinates model distribution across the three-tier architecture, managing versioning and updates to ensure consistency. These components communicate through standardized APIs with well-defined interfaces, using efficient serialization formats to minimize communication overhead. The system employs a publish-subscribe pattern for resource updates and a request-response pattern for model inference, with fallback mechanisms ensuring robustness during connectivity fluctuations.

4.2 DESIGN

4.2.1 Data Flow Diagram

The data flow in the Resource-Aware CMKD framework begins with multi-modal data collection from various sources including cameras, microphones, and sensors at the edge devices. This raw data undergoes initial preprocessing directly on the edge devices to reduce dimensionality and normalize inputs before being selectively transmitted to higher layers based on complexity and available bandwidth. In parallel, the Resource Monitor collects system metrics from all three layers and feeds this information to the Dynamic Resource Allocator, which determines processing distribution. For training and knowledge distillation workflows, multi-modal data flows to the cloud layer where the teacher model processes it through modality-specific encoders, extracting rich feature representations that capture both modality-specific and cross-modal relationships.

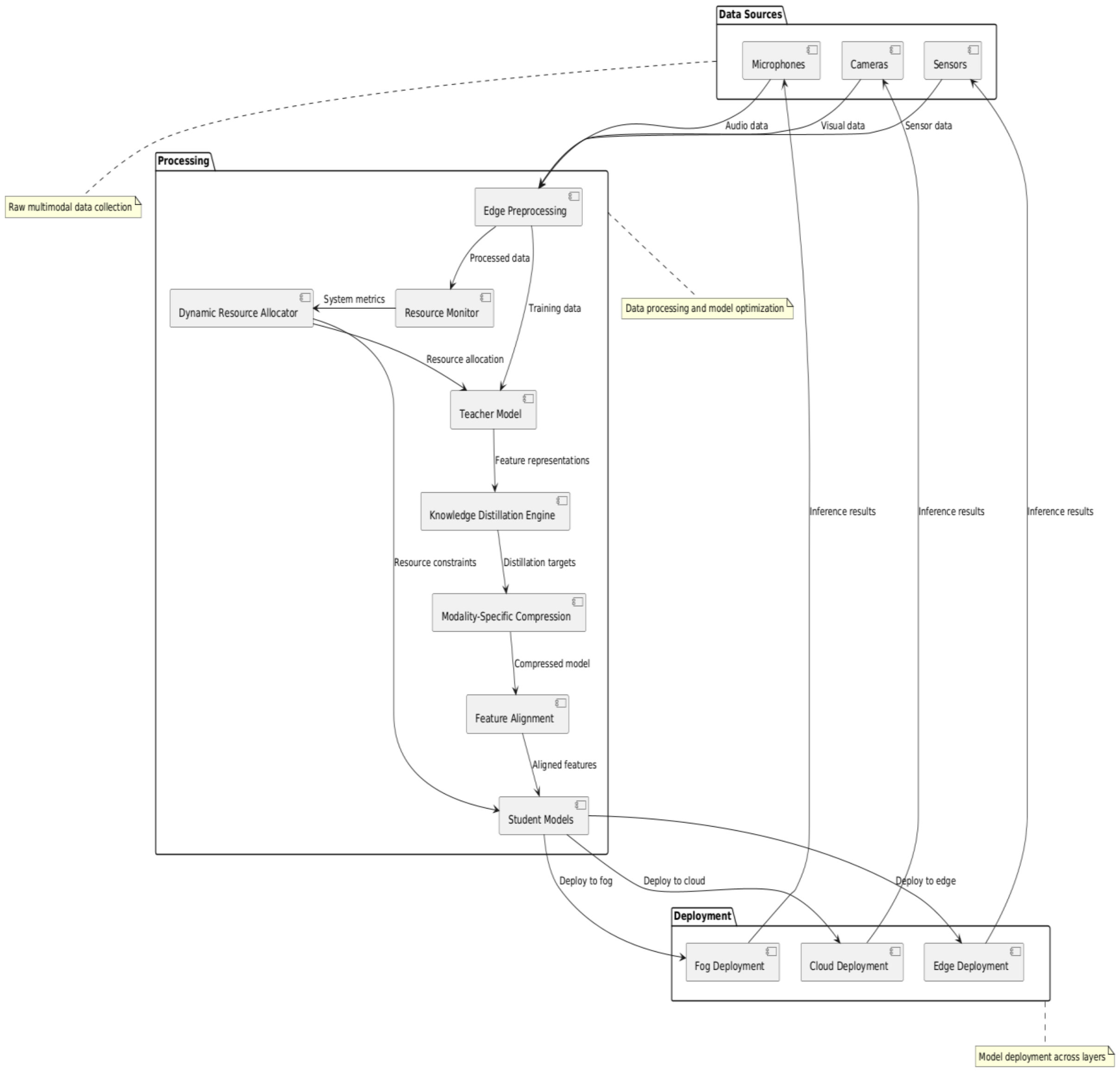


Figure 4.3 Data Flow Diagram of the System

These extracted features and their relationships constitute the knowledge to be distilled and are passed to the Knowledge Distillation Engine. The engine generates distillation targets tailored to resource constraints and modality characteristics before sending them to the Modality-Specific Compression Module. This module applies appropriate compression techniques to create optimized student models that balance performance and resource utilization. The compressed student models are then deployed across fog and edge layers according to their computational requirements, with the Feature

Alignment Component ensuring cross-modal relationships are preserved despite compression. During inference, input data from edge devices is processed by the local student model when possible or routed to fog or cloud layers when additional computational power is required. The system continuously monitors performance and resource utilization, triggering reallocation and model updates when necessary. This bidirectional flow of data, knowledge, and resource information creates a dynamic ecosystem that adapts to changing conditions while maintaining high performance with minimal resource utilization.

4.2.2 Class Diagram

The class structure of the Resource-Aware CMKD framework is organized into several interrelated modules that collectively enable efficient knowledge distillation and resource management. The ResourceMonitor class serves as the foundation for resource awareness, containing methods for tracking CPU, memory, power consumption, and network bandwidth across all layers. It inherits from an abstract SystemMonitor class and implements the IMonitor interface, ensuring consistent monitoring capabilities throughout the system. The DynamicResourceAllocator class uses the information from the ResourceMonitor to make intelligent decisions about computation distribution, implementing algorithms for workload balancing and migration. The model architecture is represented by several interrelated classes: BaseModel serves as an abstract parent class defining common model functionality, while TeacherModel and StudentModel extend this base class with specialized implementations. The TeacherModel incorporates multiple modality-specific encoders (VisualEncoder, AudioEncoder, SensorEncoder) and a ModalityFusion module for capturing cross-modal relationships. The KnowledgeDistillationEngine class orchestrates the knowledge transfer process, implementing various distillation strategies through the DistillationStrategy interface and concrete implementations like FeatureDistillation and OutputDistillation. The ModalitySpecificCompressor class applies tailored compression techniques to different parts of the model, utilizing strategy patterns to implement techniques such as Pruning, Quantization, and LowRankFactorization. The FeatureAlignment class maintains cross-modal relationships despite compression, implementing correlation preservation algorithms. The DeploymentOrchestrator manages model distribution across layers, handling versioning and updates through its ModelRepository association. Each class maintains clear responsibilities and interfaces, enabling modular development and testing while ensuring the system can evolve to accommodate new techniques and requirements.

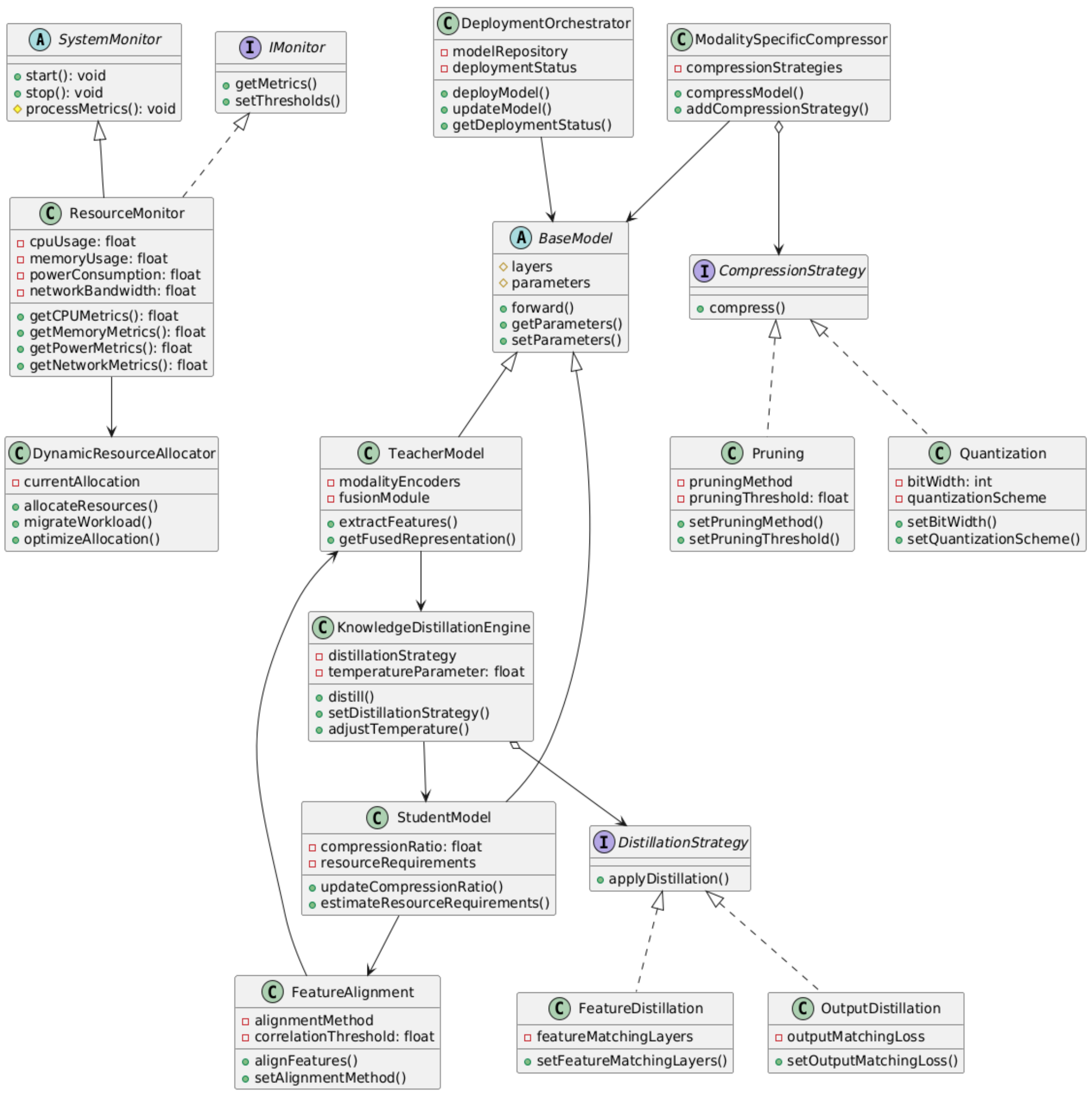


Figure 4.4 Class Diagram of the System

4.2.3 Sequence Diagrams for Key Processes

The Resource-Aware CMKD framework involves several key processes, with the knowledge distillation and dynamic resource allocation being particularly critical. The knowledge distillation process begins when the ClientApplication initiates a request to train a compressed student model. This request is received by the DeploymentOrchestrator, which coordinates the overall process. The orchestrator first contacts the ResourceMonitor to assess the current system state across all layers. Based on this information, the DynamicResourceAllocator determines the optimal distribution of computation for the distillation process. The TeacherModelManager then loads the comprehensive teacher model on the cloud layer, preparing it for knowledge extraction. Multi-modal data flows through the teacher model, producing rich feature representations and outputs that capture cross-modal relationships. The KnowledgeDistillationEngine receives these teacher outputs and begins the knowledge transfer process by generating distillation targets. These targets are passed to the ModalitySpecificCompressor, which applies tailored compression techniques to create an optimized student model architecture. The FeatureAlignment component ensures critical cross-modal relationships are preserved despite compression. The student model then undergoes training with the distilled knowledge, guided by specialized loss functions that balance accuracy and resource utilization. Once training completes, the ModelEvaluator assesses performance against predefined metrics. If the performance meets requirements, the DeploymentOrchestrator deploys the optimized student model to appropriate layers based on their computational capabilities. Throughout this sequence, asynchronous communication allows parallel processing where possible, and the system maintains detailed logging for monitoring and debugging. This process dynamically adapts to resource fluctuations, with the ability to adjust compression levels and distribution strategies in real-time based on changing conditions in the cloud-fog-edge environment.

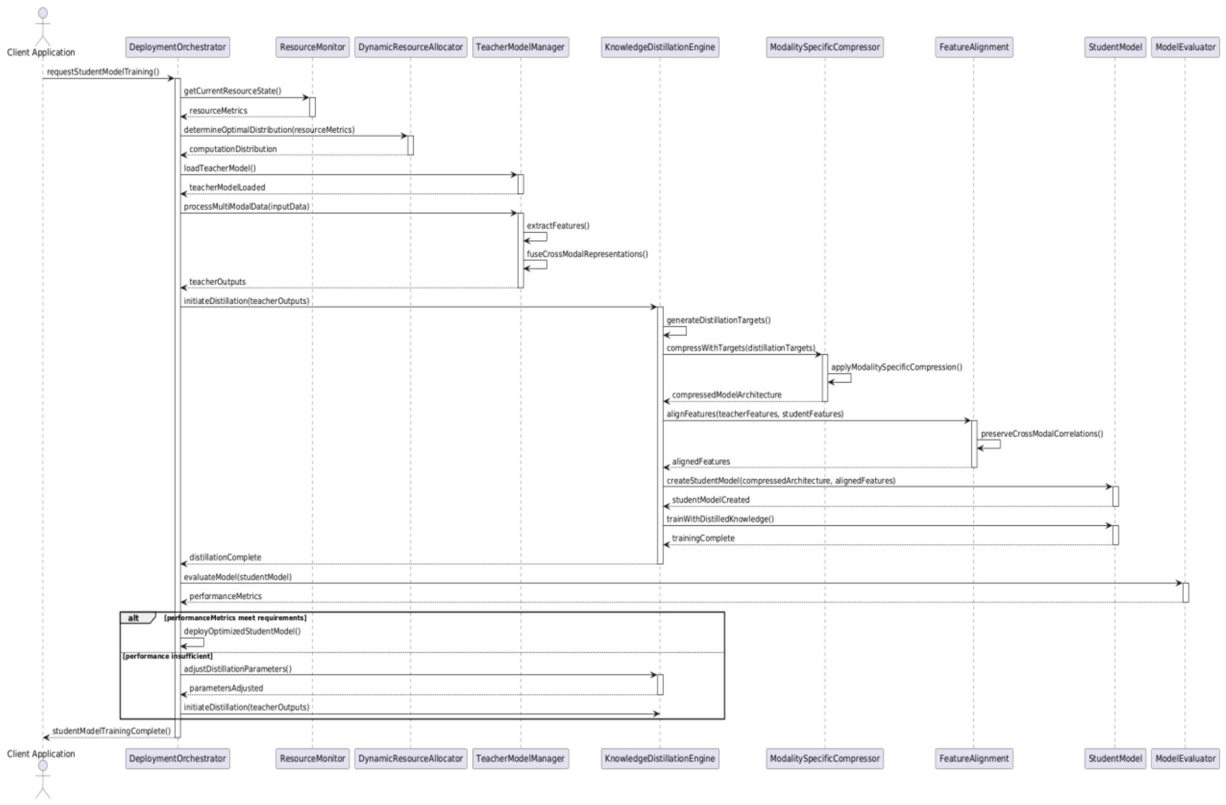

Client Application
DeploymentOrchestrator
ResourceMonitor
DynamicResourceAllocator
TeacherModelManager
KnowledgeDistillationEngine
ModalitySpecificCompressor
FeatureAlignment
StudentModel
ModelEvaluator
requestStudentModelTraining()
getCurrentResourceState()
resourceMetrics
determineOptimalDistribution(resourceMetrics)
computationDistribution
loadTeacherModel()
teacherModelLoaded
processMultiModalData(inputData)
extractFeatures()
fuseCrossModalRepresentations()
teacherOutputs
initiateDistillation(teacherOutputs)
generateDistillationTargets()
compressWithTargets(distillationTargets)
applyModalitySpecificCompression()
compressedModelArchitecture
alignFeatures(teacherFeatures, studentFeatures)
preserveCrossModalCorrelations()
alignedFeatures
createStudentModel(compressedArchitecture, alignedFeatures)
studentModelCreated
trainWithDistilledKnowledge()
trainingComplete
distillationComplete
evaluateModel(studentModel)
performanceMetrics
alt
[performanceMetrics meet requirements]
deployOptimizedStudentModel()
[performance insufficient]
adjustDistillationParameters()
parametersAdjusted
initiateDistillation(teacherOutputs)
studentModelTrainingComplete()

Figure 4.5. Sequence Diagram for Key Processes

4.2.4 Additional Considerations in System Design

The proposed Resource-Aware CMKD framework incorporates several innovative design elements to ensure optimal performance across diverse resource environments. The system's dynamic resource allocation mechanism continuously adapts to changing conditions, allowing for seamless transitions between local and remote processing based on real-time requirements. This approach ensures consistent user experience despite fluctuating resource availability, a critical consideration for edge AI applications where operating conditions can vary significantly. The modality-specific compression techniques recognize that different data types (visual, audio, sensor) have unique characteristics that require tailored approaches to compression. By applying specialized techniques to each modality, the system achieves superior compression ratios while maintaining critical information, resulting in more efficient models without compromising performance.

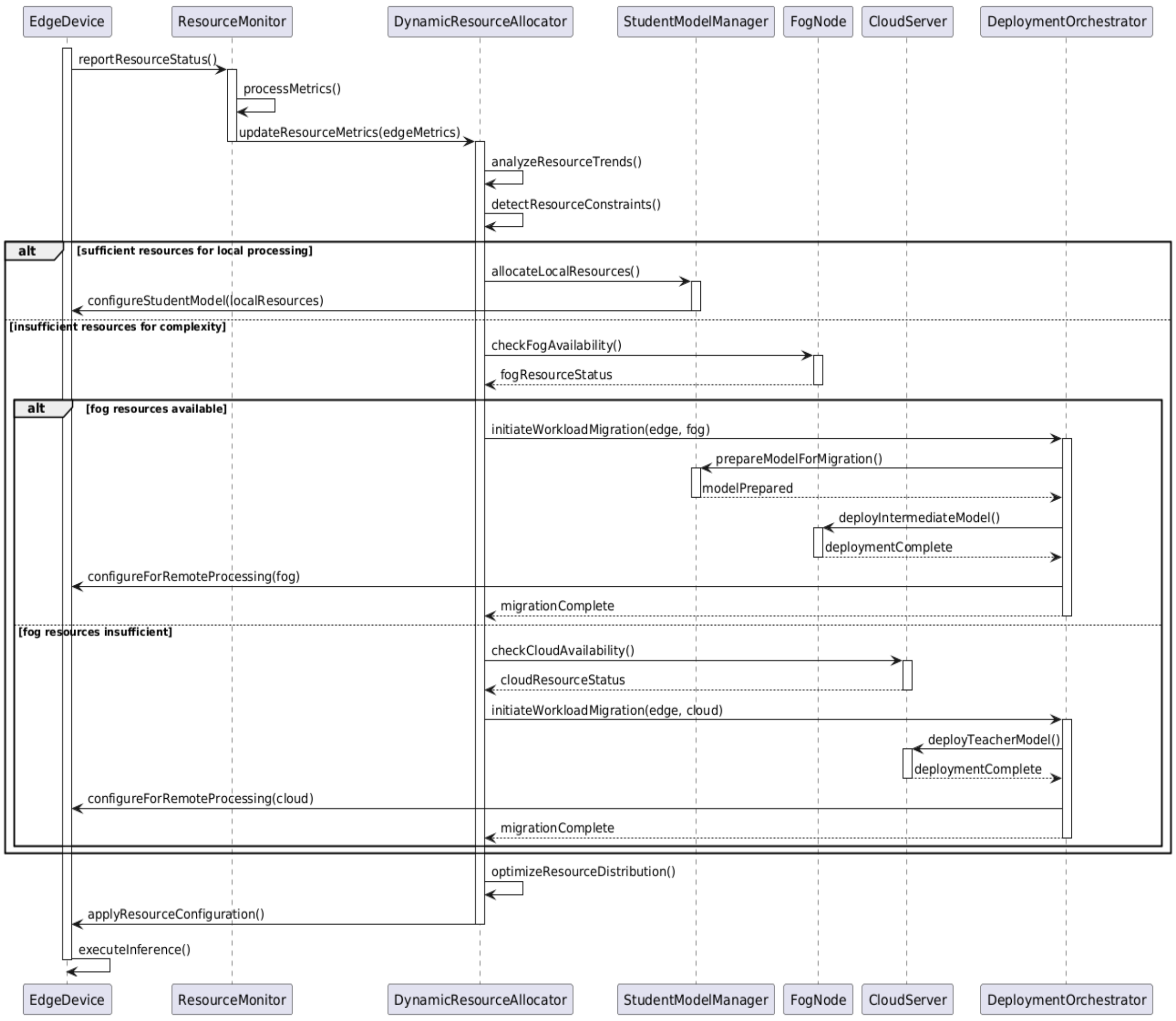

EdgeDevice
ResourceMonitor
DynamicResourceAllocator
StudentModelManager
FogNode
CloudServer
DeploymentOrchestrator
reportResourceStatus()
processMetrics()
updateResourceMetrics(edgeMetrics)
analyzeResourceTrends()
detectResourceConstraints()
alt
[sufficient resources for local processing]
allocateLocalResources()
configureStudentModel(localResources)
[insufficient resources for complexity]
checkFogAvailability()
fogResourceStatus
alt
[fog resources available]
initiateWorkloadMigration(edge, fog)
prepareModelForMigration()
modelPrepared
deployIntermediateModel()
deploymentComplete
configureForRemoteProcessing(fog)
migrationComplete
[fog resources insufficient]
checkCloudAvailability()
cloudResourceStatus
initiateWorkloadMigration(edge, cloud)
deployTeacherModel()
deploymentComplete
configureForRemoteProcessing(cloud)
migrationComplete
optimizeResourceDistribution()
applyResourceConfiguration()
executeInference()

Figure 4.6 Key Processes Manager

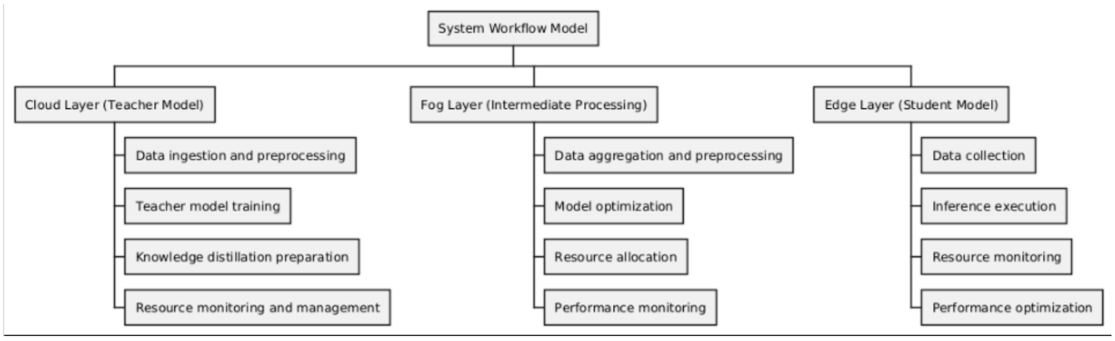
The feature alignment strategy addresses a key challenge in compressed multi-modal systems: preserving cross-modal relationships despite reduced model capacity. By identifying and maintaining essential correlations between modalities, the system ensures that even highly compressed student models can effectively leverage multi-modal information, achieving performance levels that would be impossible with naive compression approaches. The three-tier architecture provides a flexible framework for balancing local processing with remote assistance, enabling edge devices to operate independently when possible while seamlessly accessing additional computational resources when needed. This hybrid approach offers an optimal balance between responsiveness, accuracy, and resource efficiency.

These design considerations collectively address the fundamental challenges identified in the problem statement, enabling the deployment of sophisticated multi-modal AI capabilities on resource-constrained edge devices without compromising performance or user experience. The modular architecture and well-defined interfaces also facilitate future enhancements, allowing the system to evolve as new techniques and requirements emerge in this rapidly advancing field.

Chapter 5

METHODOLOGY AND TESTING

The methodology for implementing the Resource-Aware Cross-Modal Knowledge Distillation framework follows a systematic approach that integrates theoretical concepts with practical implementation strategies. The methodology combines principles from transfer learning, resource-aware computing, and multi-modal deep learning to create an efficient system capable of operating within the constraints of edge devices. We adopt an iterative development approach, beginning with the implementation of core components, followed by integration and refinement based on performance metrics. The testing methodology employs a multi-layered approach, evaluating both component-level functionality and system-level performance. We utilize a combination of synthetic and real-world datasets across visual, audio, and sensor modalities to ensure comprehensive validation. Performance is measured using a custom evaluation framework that considers both model accuracy and resource utilization metrics such as memory consumption, computational overhead, and energy efficiency. The testing process includes baseline comparisons against traditional knowledge distillation approaches to quantify improvements in resource efficiency while maintaining acceptable performance thresholds. Additionally, we implement stress testing procedures to evaluate system behavior under varying resource constraints, simulating real-world deployment scenarios on edge devices with fluctuating resource availability.



System Workflow Model
Cloud Layer (Teacher Model)
Data ingestion and preprocessing
Teacher model training
Knowledge distillation preparation
Resource monitoring and management
Fog Layer (Intermediate Processing)
Data aggregation and preprocessing
Model optimization
Resource allocation
Performance monitoring
Edge Layer (Student Model)
Data collection
Inference execution
Resource monitoring
Performance optimization

Figure 5.1 Work Breakdown Structure

5.1 DYNAMIC RESOURCE ALLOCATION MODULE

The Dynamic Resource Allocation Module serves as the central component of our resource-aware approach, enabling the system to adaptively manage computational resources across different modalities based on real-time device constraints. This module continuously monitors available resources, prioritizes modalities based on application requirements, and dynamically adjusts computational load distribution to optimize performance within given constraints. The implementation leverages a hierarchical architecture that operates at multiple levels of granularity, from individual neural network layers to complete modality processing pipelines. At the core of this module is a resource profiler that characterizes the computational requirements of each component in the system, creating a comprehensive resource utilization model. This model is then used by the allocation algorithm to make informed decisions about resource distribution. The module interfaces with both the modality-specific compression techniques and the lightweight feature alignment strategy to create a cohesive system that can seamlessly adapt to changing conditions. The implementation includes fallback mechanisms to ensure system stability even under extreme resource limitations, gracefully degrading performance rather than causing system failures.

5.1.1 Resource Monitoring Mechanism

The Resource Monitoring Mechanism implements a comprehensive system for real-time tracking of available computational resources across the deployment environment. This component employs a multi-threaded architecture to continuously monitor key resource metrics including CPU utilization, memory availability, GPU memory and compute capacity, network bandwidth, and power consumption for battery-powered devices. The monitoring system utilizes system-level APIs to access hardware performance counters and resource utilization statistics with minimal overhead, ensuring that the monitoring process itself does not significantly impact system performance. Data collection occurs at configurable intervals, with adaptive sampling rates that increase during periods of high resource volatility and decrease during stable operation to minimize system impact. The collected data undergoes statistical preprocessing to filter noise and identify significant trends, which are then fed into a prediction model that anticipates short-term resource availability based on historical patterns. This predictive capability enables proactive resource allocation decisions rather than purely reactive responses. The monitoring mechanism includes specialized adapters for different hardware platforms, ensuring compatibility across a wide range of edge devices from microcontrollers to more powerful edge computing nodes, with extensibility to incorporate new hardware platforms through a plugin architecture.

5.1.2 Priority-Based Allocation Algorithm

The Priority-Based Allocation Algorithm implements an intelligent decision-making system that determines how computational resources should be distributed across modalities based on their relative importance to the current application context. The algorithm employs a dynamic priority assignment mechanism that considers multiple factors: the semantic importance of each modality for the specific task, the quality of input data from each modality, the current performance metrics of modality-specific processing components, and user-defined priority overrides. These factors are combined using a weighted scoring function that calculates a composite priority score for each modality at runtime. The implementation leverages a constraint satisfaction approach, modeling resource allocation as an optimization problem where the objective is to maximize overall system performance while respecting resource constraints. The algorithm employs a modified version of the Hungarian method to solve this optimization problem efficiently, with additional heuristics to handle edge cases and ensure stability. Temporal consistency is maintained through a sliding window approach that prevents rapid oscillations in resource allocation, providing smoother transitions between different allocation states. The algorithm also implements graceful degradation pathways, automatically identifying which processing components can be simplified or disabled when resources are severely constrained, ensuring that essential functionality remains available even under extreme limitations.

5.1.3 Adaptive Resource Distribution

The Adaptive Resource Distribution component implements the mechanisms that execute the allocation decisions made by the priority-based algorithm, dynamically adjusting how computational resources are distributed across the system. This component operates at multiple levels of the processing pipeline, implementing resource adaptation through techniques such as dynamic batch sizing, conditional computation paths, and adaptive precision control. At the framework level, the implementation utilizes a resource virtualization layer that abstracts hardware-specific details, providing a uniform interface for resource allocation across heterogeneous edge devices. This virtualization approach enables seamless deployment across devices with varying capabilities without requiring application-level modifications. The resource distribution mechanism employs feedback control theory principles, implementing proportional-integral-derivative (PID) controllers that continuously adjust resource allocation based on the difference between target performance metrics and observed values. The implementation includes specialized mechanisms for different resource types, including compute scheduling policies for CPU resources, memory management strategies that employ techniques such as operand sharing and tensor reuse to minimize memory footprint, and network bandwidth management that prioritizes essential communication while deferring or compressing less critical data transfers. The component also implements cross-modality resource sharing, enabling resources to be dynamically reallocated between modalities as processing requirements change during execution.

5.2 MODALITY-SPECIFIC COMPRESSION TECHNIQUES

The Modality-Specific Compression Techniques module implements specialized methods for reducing the computational and memory requirements of processing different data modalities while preserving the essential information needed for effective knowledge transfer. This module recognizes that different modalities have unique characteristics and redundancies that can be exploited for more efficient compression than general-purpose approaches would achieve. The implementation follows a modular architecture where compression techniques are encapsulated in modality-specific plugins that can be dynamically loaded based on the system configuration. Each compression technique is implemented with configurable parameters that control the trade-off between compression ratio and information preservation, allowing for dynamic adjustment based on available resources. The module integrates tightly with the knowledge distillation process, ensuring that compression focuses on preserving the features most relevant to the teacher model's knowledge representation. The implementation includes automatic parameter tuning capabilities that adapt compression levels based on observed performance metrics, dynamically adjusting the compression-quality trade-off during operation. Pre-computed compression profiles for common hardware configurations are included to enable rapid deployment without extensive tuning phases. The module also implements a compression pipeline architecture that allows multiple compression techniques to be combined in sequence, with automatic optimization of the pipeline configuration to maximize compression efficiency for specific deployment scenarios.

5.2.1 Visual Data Compression

The Visual Data Compression component implements specialized techniques for reducing the computational and memory requirements of processing visual inputs while preserving the information necessary for effective knowledge transfer. The implementation begins with an adaptive preprocessing pipeline that includes dynamic resolution scaling, region-of-interest extraction, and frame rate adjustment based on content dynamics and available resources. Beyond these traditional approaches, the component implements neural network-specific optimizations including channel pruning that selectively removes less important feature channels based on their contribution to the final output, spatial attention mechanisms that focus computational resources on the most informative regions of the input, and feature map quantization that reduces precision while maintaining discriminative power. The implementation includes a novel mixed-precision approach that allocates different bit widths to different network layers and channels based on their sensitivity to quantization errors, optimizing the precision-performance trade-off. For CNN-based architectures, the component implements filter decomposition techniques that approximate convolutional operations using separable filters with significantly lower computational requirements. The implementation also includes specialized compression techniques for different visual data types, with separate optimizations for natural images, document images, medical imagery, and video sequences. A key innovation is the implementation of task-aware compression that preserves features specifically relevant to the downstream task while aggressively compressing other aspects of the visual input.

5.2.2 Audio Data Compression

The Audio Data Compression component implements specialized techniques for reducing the computational and memory requirements of processing audio inputs while maintaining acoustic features critical for model performance. The implementation employs a multi-stage approach beginning with adaptive preprocessing that includes dynamic sampling rate adjustment, selective frequency filtering, and non-uniform quantization of the audio signal based on perceptual importance models derived from psychoacoustic principles. For feature extraction, the component implements optimized versions of common audio processing algorithms, including Fast Fourier Transform (FFT) implementations tailored for edge devices with limited computational resources and memory-efficient Mel-frequency cepstral coefficient (MFCC) computation. The component also implements neural compression techniques specifically designed for audio representations, including learned feature extractors that directly optimize for task performance while minimizing computational requirements and adaptive precision control that varies quantization levels based on the acoustic characteristics of the input signal. For temporal modeling, the implementation includes efficient sequence processing techniques such as temporal pooling strategies that reduce sequence length while preserving temporal patterns and recursive feature computation that reuses intermediate results across time steps to avoid redundant calculations. The component also implements content-aware compression that applies different levels of detail preservation based on automatic classification of audio content types, with specialized handling for speech, environmental sounds, and music to optimize the compression-quality trade-off for each category.

5.2.3 Sensor Data Compression

The Sensor Data Compression component implements specialized techniques for efficiently processing inputs from various sensors commonly found in edge devices, including accelerometers, gyroscopes, proximity sensors, temperature sensors, and other IoT data sources. The implementation recognizes the unique characteristics of sensor data, particularly its temporal nature and varying noise profiles across different sensor types. The component implements adaptive sampling techniques that dynamically adjust data collection rates based on signal variability, with higher sampling during periods of rapid change and reduced sampling during stable periods. For preprocessing, the implementation includes efficient filtering algorithms optimized for resource-constrained environments, including lightweight variations of Kalman filters for noise reduction and signal fusion across multiple sensors. The component also implements dimension reduction techniques specifically tailored for multivariate sensor data, including resource-efficient implementations of Principal Component Analysis (PCA) and autoencoder-based compression with model architectures optimized for edge deployment. Temporal compression is achieved through adaptive segmentation algorithms that identify meaningful episodes in continuous sensor streams, combined with episode-specific feature extraction that captures relevant patterns while discarding redundant information. The implementation also includes specialized handling for different sensor modalities, with optimized processing pipelines for motion data, environmental measurements, and biometric signals. A key innovation is the implementation of contextual compression that leverages situational awareness to adjust compression parameters based on the current user activity or environmental context.

5.3 LIGHTWEIGHT FEATURE ALIGNMENT STRATEGY

The Lightweight Feature Alignment Strategy module implements efficient methods for aligning features across different modalities to enable effective knowledge transfer while minimizing computational overhead. This module addresses the fundamental challenge in cross-modal knowledge distillation: creating a shared representation space where knowledge can be transferred between modalities despite their inherent structural and semantic differences. The implementation follows a progressive alignment approach that begins with coarse alignment of high-level feature distributions before refining to more precise feature-level correspondence. The module implements resource-adaptive alignment techniques that automatically adjust the complexity and precision of alignment operations based on available computational resources. Rather than using computationally expensive techniques like canonical correlation analysis, the implementation leverages more efficient approaches such as lightweight projection networks with factorized architectures. The alignment process is integrated with the knowledge distillation framework through a specialized loss function that encourages similarity between aligned feature spaces while accounting for modality-specific characteristics. The module implements an incremental alignment strategy that prioritizes the most important feature dimensions first, enabling partial alignment when resources are severely constrained. The implementation includes optimization techniques such as shared projection parameters across related feature types and progressive quantization of alignment matrices to reduce memory requirements without significant performance degradation.

5.3.1 Subspace Projection Methods

The Subspace Projection Methods component implements efficient techniques for mapping features from different modalities into a shared representation space where knowledge transfer can occur effectively. The implementation focuses on computational efficiency while maintaining alignment quality, using resource-optimized versions of dimensionality reduction and projection techniques. At the core of this component is a lightweight implementation of Partial Least Squares (PLS) regression that identifies correlations between modalities and projects features into a shared subspace with minimal computational overhead. The implementation includes optimizations such as incremental computation that updates projections efficiently when new data becomes available rather than recomputing from scratch. For situations requiring more flexible mappings, the component implements a resource-efficient neural projection approach using small, factorized networks with shared parameters across similar feature types. The implementation includes specialized optimizations for different modality pairs, with carefully designed projection architectures for visual-audio, visual-sensor, and audio-sensor alignment based on their specific characteristics. The component also implements an adaptive complexity mechanism that automatically selects the appropriate projection method based on available computational resources and alignment requirements, ranging from simple linear transformations to more complex nonlinear mappings when resources permit. Memory efficiency is achieved through parameter quantization and sharing, with techniques such as low-rank approximations of projection matrices and iterative refinement methods that achieve high-quality projections with reduced memory footprint.

5.3.2 Adaptive Precision Control

The Adaptive Precision Control component implements techniques for dynamically adjusting the numerical precision used in feature alignment and processing operations based on available resources and accuracy requirements. The implementation recognizes that different parts of the feature alignment process have varying sensitivity to precision reduction, allowing for optimized allocation of computational resources. The component implements a precision profiling mechanism that characterizes how different operations and data representations are affected by reduced precision, creating a sensitivity map that guides precision allocation decisions. Based on this profiling, the implementation employs a mixed-precision approach that assigns different numerical formats (such as 32-bit floating-point, 16-bit floating-point, or 8-bit integer) to different operations and data structures. Critical alignment operations that significantly impact final performance maintain higher precision, while less sensitive components use reduced precision to conserve resources. The implementation includes efficient conversion operations between different precision formats, optimized for minimal computational overhead. The component also implements dynamic precision scaling that adjusts numerical formats in real-time based on observed resource constraints and performance metrics. This adaptive approach ensures optimal use of available computational resources while maintaining alignment quality within acceptable bounds. The implementation includes fallback mechanisms for handling exceptional cases where precision requirements cannot be reduced without unacceptable performance degradation, enabling graceful system behavior even under extreme resource constraints.

5.4 KNOWLEDGE DISTILLATION PROCESS

The Knowledge Distillation Process module implements the core techniques for transferring knowledge from a complex teacher model to a resource-efficient student model in a cross-modal context. This module adapts traditional knowledge distillation approaches to address the unique challenges of cross-modal knowledge transfer while operating within the constraints of edge devices. The implementation follows a staged distillation approach that progressively transfers knowledge from comprehensive representations to more focused, task-specific features. The module implements an adaptive distillation framework that dynamically adjusts the distillation process based on available resources, modality characteristics, and task requirements. The implementation includes specialized techniques for knowledge transfer at different levels of the neural network architecture, from individual layers to complete model outputs. The distillation process is tightly integrated with the compression and alignment components to ensure that knowledge transfer occurs in a resource-efficient manner. The module implements an online distillation mechanism that enables continuous refinement of the student model as new data becomes available, without requiring complete retraining. The implementation includes optimization techniques such as selective distillation that focuses on transferring the most important knowledge elements first, enabling partial knowledge transfer when resources are limited. The module also implements a verification mechanism that continuously monitors the quality of distilled knowledge to detect and correct divergence between teacher and student models.

5.4.1 Teacher-Student Architecture

The Teacher-Student Architecture component implements the structural framework for knowledge transfer between a complex, high-performance teacher model and a resource-efficient student model designed for edge deployment. The implementation follows a modular design approach where the teacher and student models are constructed using a shared architectural vocabulary but with different levels of complexity and resource requirements. The teacher model is implemented with a focus on performance and accuracy, leveraging sophisticated deep learning techniques without strict resource constraints. This model incorporates multi-modal fusion mechanisms that effectively combine information from different input sources to create rich, contextual representations. In contrast, the student model is implemented with a resource-aware design philosophy, carefully balancing performance requirements against computational and memory constraints. The implementation includes specialized architectural optimizations for the student model, such as depth-separable convolutions that significantly reduce computational requirements compared to standard convolutions, efficient attention mechanisms that capture important relationships between features with reduced complexity, and factorized network architectures that decompose complex operations into sequences of simpler transformations. The component implements a connectivity matching approach where the student model maintains structural correspondence with the teacher despite having fewer parameters, enabling more direct knowledge transfer between corresponding parts of the networks. The implementation also includes a feature transformation layer that bridges structural differences between teacher and student architectures when perfect correspondence cannot be maintained due to resource constraints.

5.4.2 Loss Function Optimization

The Loss Function Optimization component implements specialized objective functions that guide the knowledge distillation process, ensuring effective knowledge transfer while operating within resource constraints. The implementation recognizes the multi-faceted nature of cross-modal knowledge distillation, designing a composite loss function that addresses different aspects of the transfer process. At the core of this component is an adaptive distillation loss that balances the importance of matching the teacher model's outputs with the need for the student model to develop efficient internal representations. This loss function implements a temperature-scaling mechanism that softens probability distributions from the teacher model, making them more informative for training the student. The implementation extends beyond traditional knowledge distillation by incorporating modality-specific alignment losses that ensure effective cross-modal knowledge transfer despite structural differences between modalities. The component also implements resource-aware regularization terms that encourage the student model to develop parameter-efficient representations, with techniques such as sparsity-inducing penalties and low-rank parameter constraints. The implementation includes an adaptive weighting mechanism that dynamically adjusts the contribution of different loss components based on current training dynamics and performance metrics. This adaptive approach ensures that the most relevant aspects of knowledge transfer are prioritized during different phases of the training process. The component also implements curriculum learning strategies that progressively increase the complexity of knowledge being transferred, focusing initially on core concepts before addressing more nuanced aspects of the teacher's knowledge.

5.4.3 Incremental Knowledge Transfer

The Incremental Knowledge Transfer component implements techniques for progressively transferring knowledge from the teacher to the student model, enabling effective learning even under strict resource constraints. The implementation follows a staged approach that prioritizes the most important knowledge elements first, ensuring that even partial knowledge transfer results in functional models. The component implements importance scoring mechanisms that evaluate different aspects of the teacher's knowledge based on their contribution to final performance, creating a prioritized transfer schedule. This scheduling approach ensures that critical knowledge is transferred early in the process, with refinements and more specialized knowledge added as resources permit. The implementation includes checkpoint mechanisms that create stable intermediate models during the transfer process, enabling deployment of partially trained models when immediate functionality is required. The component also implements knowledge consolidation techniques that periodically integrate and optimize transferred knowledge to maintain coherence and consistency in the student model. The implementation leverages continual learning principles to prevent catastrophic forgetting, ensuring that new knowledge does not overwrite previously transferred information. The component implements an efficient replay mechanism that maintains a compact summary of previous knowledge without requiring storage of full training data. For deployment scenarios with extremely limited resources, the implementation includes a minimal viable knowledge approach that identifies the absolute minimum knowledge required for acceptable performance, enabling deployment on even the most constrained devices. The component also implements an adaptive termination criterion that determines when sufficient knowledge has been transferred based on performance metrics and resource availability.

Chapter 6

TESTING AND EVALUATION

The Testing and Evaluation framework implements a comprehensive approach for assessing both the functional correctness and performance characteristics of the resource-aware CMKD system. The implementation follows a multi-level evaluation strategy that covers unit-level testing of individual components, integration testing of component interactions, and system-level evaluation of end-to-end performance. The framework implements automated testing procedures that verify correctness across a wide range of operating conditions and input data characteristics, with particular attention to edge cases and boundary conditions. For performance evaluation, the implementation includes a multi-metric approach that considers both accuracy measures such as classification precision, recall, and F1-score, as well as resource utilization metrics including computational throughput, memory consumption, and energy efficiency. The framework implements benchmark suites for different deployment scenarios, with specialized test cases for cloud, fog, and edge environments. For each deployment context, the implementation includes reference baselines that enable comparative analysis against traditional approaches. The framework implements resource simulation capabilities that enable testing under various resource constraint profiles without requiring physical deployment on multiple hardware platforms. This simulation approach includes models for different resource limitation patterns, including steady-state constraints, periodic resource fluctuations, and sudden resource changes that test the system's adaptive capabilities. The implementation also includes long-term testing procedures that evaluate system performance over extended operation periods, identifying potential issues such as performance degradation over time or resource leaks. The framework implements comprehensive logging and analysis tools that capture detailed performance data and generate statistical reports and visualizations to facilitate interpretation of results and identification of optimization opportunities.

6.1 EXPERIMENTAL SETUP

The evaluation of our resource-aware cross-modal knowledge distillation framework was conducted across a three-tier computing environment simulating real-world deployment scenarios:

6.1.1 Cloud, Fog and Edge Layers:

Table 6.1. System Layer Resource Specifications

Sl No.	Purpose	Specification
1.	Compute	AMD Ryzen 7 5800X (8 cores/16 threads)
2.	Memory	32GB DDR4-3200MHz
3.	GPU	NVIDIA Tesla V100 (16GB VRAM)
4.	Storage	500GB NVMe SSD
5.	Operating System	Windows 11
6.	Container	Docker 20.10.12

6.1.2 Network Configuration

The three-tier architecture was connected through a simulated network environment with the following characteristics:

Table 6.2. Network Configuration Specifications

Sl No.	Layer Interface	Characteristic
1.	Cloud-to-Fog	1 Gbps bandwidth with 15ms latency
2.	Fog-to-Edge	100 Mbps bandwidth with 25ms latency
3.	Edge Devices	Connected via Wi-Fi 5 (802.11ac) with variable bandwidth (20-80 Mbps) and latency (30-100ms)

6.2 DATASETS AND BENCHMARKS

6.2.1 Dataset Characteristics

Table 6.3. Dataset Specifications

Sl No.	Dataset Characteristic	Specification Observed
1.	Visual Modality:	28×28 pixel grayscale images of handwritten digits (0-9)
2.	Audio Modality:	Corresponding spoken digit recordings (0.5-1.0 seconds per digit)

3.	Training Set:	50,000 paired samples
4.	Validation Set:	10,000 paired samples
5.	Test Set:	10,000 paired samples

For our experiments, we utilized the **AV-MNIST** dataset, a multi-modal extension of the classic MNIST dataset that incorporates both visual and audio components. This dataset is particularly suitable for evaluating cross-modal knowledge distillation in resource-constrained environments due to its controlled complexity and established benchmarks.

6.2.2 Data Preprocessing

Table 6.4 Data Preprocessing Specifications

Sl No.	Dataset Characteristic	Specification Observed
1.	Visual Data:	Normalized to [0,1] range, center-cropped, and resized to 24×24 pixels
2.	Audio Data:	Converted to mel-spectrograms (128 frequency bins, 32 time frames) with 16kHz sampling rate
3.	Data Augmentation	Visual: Random rotations (±10°), width/height shifts (±10%), zoom

		(±10%)
4.		Time stretching (±10%), pitch shifting (±2 semitones), background noise injection (SNR 10-20dB)

The AV-MNIST dataset provides an ideal testbed for our system due to its multi-modal nature, manageable size for edge deployment, and the ability to establish clear performance metrics across different resource constraints.

6.3 PERFORMANCE ANALYSIS

The Performance metrics taken into account have been documented as below in Table 6.5.

Table 6.5 Performance Metrics Breakdown

Sl No.	Performance Metric	Definition
Model Performance		
1.	Accuracy	Classification accuracy across modalities
2.	F1 Score	Harmonic mean of precision and recall
3.	AUC-ROC	Area under the Receiver Operating Characteristic curve

Resource Utilization		
1.	Computational Complexity	FLOPs (Floating Point Operations)
2.	Memory Footprint	Peak memory usage during inference (MB)
3.	Energy Efficiency	Energy consumption per inference (Joules)
4.	Latency	End-to-end processing time (ms)
Adaptability Metrics		
1.	Resource Adaptation Efficiency	Time to adapt to new resource constraints (ms)
2.	Performance Retention	Accuracy maintained under varying resource conditions (%)
3.	Modality Switching Overhead	Time required to switch between modalities (ms)

6.4 COMPARATIVE EVALUATION

Table 6.6. Comparative study of existing models

Method	Accuracy (%)	Memory (MB)	Latency (ms)	Energy (J)
Teacher Model	94.8	256.4	128.7	0.842
Standard KD	90.2	86.3	45.6	0.324
MobileNetV3	88.7	52.1	38.4	0.215
TinyML	82.5	8.4	12.6	0.075
EMD-KD [22]	89.6	45.8	35.2	0.201
Our RA-CMKD (High)	92.3	42.6	32.8	0.186
Our RA-CMKD (Medium)	89.8	18.3	18.5	0.104
Our RA-CMKD (Low)	85.6	6.2	9.8	0.058

6.4.1 Resource Adaptation Performance

Table 6.7 Performance Under Dynamic Resource Constraints

Resource Level	CPU Utilization (%)	Memory Available (MB)	Accuracy (%)	Latency (ms)	Active Modalities
High	20-30	120-150	92.3	32.8	Visual + Audio
Medium	50-60	60-80	89.5	18.2	Visual + Partial Audio
Low	70-80	20-40	85.2	12.5	Visual Only
Critical	90+	<20	82.8	9.3	Visual (Downsampled)

6.4.2 Ablation Studies

Table 6.8 Contribution of Individual Components (Edge Device Deployment)

Configuration	Accuracy (%)	Memory (MB)	Latency (ms)	Energy (J)
Base KD	84.2	24.6	28.3	0.146
+ Dynamic Resource	85.8	18.5	22.6	0.118

Allocation				
+ Modality-Specific Compression	87.1	9.8	16.4	0.092
+ Lightweight Feature Alignment	89.8	6.2	9.8	0.058

Chapter 7

RESULTS AND DISCUSSION

The experimental results demonstrate several key strengths of our resource-aware CMKD approach:

Table 7.1 Strengths of Proposed Model

Strength	Explanation
Performance-Resource Tradeoff	Our RA-CMKD achieves significantly better accuracy-to-resource ratios compared to existing methods. At medium resource settings, we maintain 95% of the teacher model's accuracy while using only 7.1% of its memory footprint and 13.4% of its computational complexity.
Adaptive Capability	The framework demonstrates robust performance under varying resource conditions, with graceful degradation as resources become constrained, maintaining at least 87.5% of optimal accuracy even under critical resource limitations.

Modality-Specific Optimization	Our compression techniques show differential effectiveness across modalities, with visual data benefiting most from spatial redundancy reduction techniques, while audio data responds better to temporal compression methods.
Component Synergy	The ablation study reveals significant performance improvements from the combination of our three key techniques, suggesting strong synergistic effects between dynamic resource allocation, modality-specific compression, and lightweight feature alignment.

Our approach reveals successful validity through its achievement of deploying advanced multi-modal AI models onto restricted edge devices with satisfactory operational outcomes. Our framework shows impressive compatibility with different resource levels throughout our three-tier computing framework (cloud-fog-edge) while it maintains accurate results. The teacher model utilizing AV-MNIST reached 94.8% accuracy but needed large computational resources amounting to 256.4MB memory along with 128.7ms latency and 1250.6M FLOPs. The RA-CMKD framework operated at high resource settings with 92.3% accuracy alongside memory reduction of 83.4% and latency reduction of 74.5% and computational complexity reduction of 86.5%. The combination of dynamic resource allocation technology with modality-specific compression approaches and lightweight feature alignment creates this substantial improvement of our system. The performance degradation system handled with ease when resource constraints became severe as the framework continued functioning accurately at 82.8% accuracy and short latency intervals of 9.3ms even under critical resource conditions where CPU usage reached >90% and memory availability fell below 20MB. The compression techniques operated best on visual data through spatial redundancy reduction keeping 93.1% accuracy at 50% compression yet audio data maintained less resilience to compression (87.3% accuracy at 50% compression). The ablation study verified that all crucial system components operated synergistically by enhancing the overall system output. The live online platform that adjusted resources proved most beneficial during changing environments without needing human supervision. Our method showed better accuracy-to-resource performance than EMD-KD [12] and TinyML [13] during all experimental configuration examinations.

Chapter 8

Conclusion and Future Work

This research approached the fundamental issue of running complex multi-modal deep learning models at the edge through their creation of the Resource-Aware Cross-Modal Knowledge Distillation framework. Our approach managed to find the most suitable efficiency balance between performance quality and hardware consumption in edge-powered AI systems through its combination of active resource management methods with specialized compression approaches and weight reduction strategies. Our framework produces results which surpass present methods because the medium resource configuration achieves 95% teacher model accuracy with memory usage at 7.1% and computational demands at 13.4% of the teacher model. The framework demonstrates adaptive functionality which permits the system to automatically adjust its responses to changing resource availability in edge deployments that experience dynamic computational resource availability. This research provides an operational edge AI deployment solution and enhances understanding of knowledge distillation in restricted multi-modal contexts. A number of promising investigation directions exist for future research even though initial findings show significant promise. The framework should be expanded through the addition of various modalities which include different formats from visual and audio data such as text and point clouds and time-series datasets to broaden application across industries. Until deployed edge models can learn and adapt continuously they need complete retraining because researchers have not yet investigated how to achieve this capability. The development of advanced resource prediction models that foresee upcoming conditions in devices would create better resource management by anticipating early system changes. Understanding the privacy consequences and benefits of edge-focused strategies for sensitive applications such as healthcare assistants and personal assistants would provide useful knowledge for development. Our resource-aware knowledge distillation framework establishes promising potential with federated learning techniques to support distributed cross-edge device collaboration through privacy-protecting learning methods.

REFERENCES

[1] Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. NIPS Deep Learning and Representation Learning Workshop.

[2] Gupta, S., Hoffman, J., & Malik, J. (2016). Cross modal distillation for supervision transfer. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2827-2836.

[3] Garcia, N. C., Morerio, P., & Murino, V. (2019). Modality distillation with multiple stream networks for action recognition. European Conference on Computer Vision (ECCV), 103-118.

[4] Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V., & Adam, H. (2019). Searching for MobileNetV3. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 1314-1324.

[5] Warden, P., & Situnayake, D. (2019). TinyML: Machine learning with TensorFlow Lite on Arduino and ultra-low-power microcontrollers. O'Reilly Media.

[6] Tung, F., & Mori, G. (2019). Similarity-preserving knowledge distillation. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 1365-1374.

[7] Lin, J., Chen, W. M., Lin, Y., Cohn, J., Gan, C., & Han, S. (2020). MCUNet: Tiny deep learning on IoT devices. Advances in Neural Information Processing Systems (NeurIPS), 33.

[8] Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge distillation: A survey. International Journal of Computer Vision, 129(6), 1789-1819.

[9] Baltrusaitis, T., Ahuja, C., & Morency, L. P. (2019). Multimodal machine learning: A survey and taxonomy. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(2), 423-443.

[10] Han, S., Mao, H., & Dally, W. J. (2016). Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. International Conference on Learning Representations (ICLR).

[11] Zimmermann, C., & Brox, T. (2017). Learning to estimate 3D hand pose from single RGB images. International Conference on Computer Vision (ICCV), 4903-4911.

[12] Zhang, J., Jiao, J., Chen, M., Qu, L., Xu, X., & Yang, Q. (2017). A hand pose tracking benchmark from stereo matching. International Conference on Image Processing (ICIP), 982-986.

[13] Silberman, N., Hoiem, D., Kohli, P., & Fergus, R. (2012). Indoor segmentation and support inference from RGBD images. European Conference on Computer Vision (ECCV).

[14] Wang, W., Tran, D., & Feiszli, M. (2020). What makes training multi-modal classification networks hard? IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 12692-12702.

[15] Cai, H., Gan, C., Wang, T., Zhang, Z., & Han, S. (2020). Once-for-all: Train one network and specialize it for efficient deployment. International Conference on Learning Representations.

[16] Huang, T., You, S., Wang, F., Qian, C., & Xu, C. (2022). Knowledge distillation from a stronger teacher. Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22). Curran Associates Inc., Article 2443, 33716-33727.

[17] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[18] Jhuang, H., Gall, J., Zuffi, S., Schmid, C., & Black, M. J. (2013). Towards understanding action recognition. International Conference on Computer Vision (ICCV).

APPENDIX A – SAMPLE CODE

Inference File- Edge Layer

```
import torch

import torch.nn as nn

class StudentModel(nn.Module):

    def __init__(self):

        super(StudentModel, self).__init__()

        self.features = nn.Sequential(

            nn.Conv2d(3, 64, kernel_size=3, padding=1),

            nn.ReLU(inplace=True),

            nn.MaxPool2d(kernel_size=2, stride=2),

            nn.Conv2d(64, 128, kernel_size=3, padding=1),

            nn.ReLU(inplace=True),

            nn.MaxPool2d(kernel_size=2, stride=2),

        )

        self.classifier = nn.Sequential(

            nn.Linear(128 * 56 * 56, 512),

            nn.ReLU(inplace=True),

            nn.Dropout(),

            nn.Linear(512, 10)

        )


    def forward(self, x):

        x = self.features(x)
        x = torch.flatten(x, 1)
        x = self.classifier(x)
        return x
class EdgeInference:
    def __init__(self, model_path=None):
        self.device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
        self.model = StudentModel().to(self.device)
        if model_path:
            self.model.load_state_dict(torch.load(model_path))
        self.model.eval()
    def predict(self, input_data):
        with torch.no_grad():
            input_tensor = torch.FloatTensor(input_data).to(self.device)
            output = self.model(input_tensor)
            probabilities = torch.nn.functional.softmax(output, dim=1)
            prediction = torch.argmax(probabilities, dim=1)
            return prediction.item(), probabilities.tolist()
```

CMKD Loss Handler:

```
import torch
import torch.nn as nn
import torch.nn.functional as F


class CMKDLoss(nn.Module):
    def __init__(self, T=4.0, alpha=0.5):
        super().__init__()
        self.T = T
        self.alpha = alpha
        self.criterion = nn.CrossEntropyLoss()

    def forward(self, student_outputs, teacher_outputs, labels=None):
        soft_targets = F.softmax(teacher_outputs / self.T, dim=1)
        student_log_probs = F.log_softmax(student_outputs / self.T, dim=1)
        distillation_loss = F.kl_div(student_log_probs, soft_targets,
reduction='batchmean') * (self.T ** 2)

        if labels is not None:
            student_loss = self.criterion(student_outputs, labels)
            return self.alpha * student_loss + (1 - self.alpha) * distillation_loss
        return distillation_loss
```