Fatima, Haneen (2026) *Internet of mirrors for personalised services: from conceptual vision to intelligent orchestration of a distributed IoT ecosystem.* PhD thesis.

https://theses.gla.ac.uk/86340/

Copyright and moral rights for this work are retained by the author

A copy can be downloaded for personal non-commercial research or study, without prior permission or charge

This work cannot be reproduced or quoted extensively from without first obtaining permission from the author

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given

Enlighten: Theses
https://theses.gla.ac.uk
research-enlighten@glasgow.ac.uk

Internet of Mirrors for Personalised Services: From Conceptual Vision to Intelligent Orchestration of a Distributed IoT Ecosystem

Haneen Fatima

Submitted in fulfilment of the requirements for the
Degree of Doctor of Philosophy

James Watt School of Engineering
College of Science and Engineering
University of Glasgow



September 2026

Abstract

Smart mirrors have emerged as interactive platforms for personalised health, beauty, and assistive services, but existing systems predominantly operate as standalone or application-specific devices. This thesis introduces the **Internet of Mirrors (IoM)** as an interconnected Internet of Things (IoT) ecosystem in which smart mirrors, users, professionals, service providers, and distributed computational resources operate collectively. Its novelty lies at the ecosystem level, providing a unified framework for coordinating heterogeneous mirror-based services, stakeholders, and computational resources.

The research first defines the conceptual and architectural foundations of the IoM through a stakeholder-oriented Consumer, Professional, and Hub hierarchy and corresponding functional layers. Practical feasibility is demonstrated through an automated dental smile-analysis application integrated into a smart-mirror interface, achieving 81.6% classification accuracy for excessive gingival-display detection.

The thesis then evaluates IoM application execution across heterogeneous computational infrastructure. Edge–cloud experiments and a physical multi-tier testbed show that performance depends jointly on processing capability, communication overhead, network conditions, workload characteristics, and concurrent demand. Consumer-only execution produced end-to-end latency of approximately 1.5 s, while higher-tier offloading reduced this to approximately 0.8 s and lowered Consumer-tier resource utilisation.

To support increasingly diverse workloads, a Context-Aware Task Distribution (CATD) framework is developed using task requirements, computational capability, network quality, geographic proximity, and data sensitivity. Across bandwidth degradation, node failure, and increased workload conditions, CATD maintained higher task-completion rates than the evaluated single-factor baselines, demonstrating the value of integrated contextual assessment.

Finally, the thesis extends coordination beyond the IoM through a reinforcement-learning-based orchestration mechanism for interaction with the Internet of Sounds (IoS). The formulation incorporates ecosystem origin, modality, hardware compatibility, system state, and application context to coordinate visual and audio workloads across shared resources. Evaluation shows lower p95 latency and higher deadline satisfaction than cloud offloading for Live-Interactive tasks, while action masking and task deferral maintain feasibility under modality-dependent resource scarcity.

Collectively, the thesis progresses the IoM from a conceptual smart-mirror ecosystem to an experimentally and computationally evaluated distributed architecture.

Acknowledgements

To Allah belongs all praise and gratitude. Upon Him is my reliance, to Him is my commitment, and from Him comes all strength, preservation, and success. By His grace alone I have been sustained throughout this journey. Through His beloved Messenger, Muhammad (pbuh), a mercy to all creation, I have found the faith, strength, and perseverance to endure its difficulties. I am also profoundly grateful to my spiritual guide, Shaykh-ul-Islam Dr Muhammad Tahir-ul-Qadri, whose teachings have deepened this connection and remained a constant source of inspiration.

I am deeply indebted to my supervisors, Dr Lina Mohjazi and Dr Ahmad Taha, for giving me the opportunity to undertake this PhD and for their unwavering support. Their passion for academic excellence and research integrity has profoundly shaped me as a researcher. Their continued guidance, mentorship and valuable insights have broadened my scholarly horizons. It has been a genuine honour to work alongside them and establish the first IoM research team together. The time spent working and learning with them has marked one of the most formative chapters of my life, and one that I will always look back on with great appreciation.

I also extend my sincere thanks to my co-supervisor, Prof. Muhammad Ali Imran, whose encouragement first set me on the path towards this PhD. His guidance, mentorship and support in establishing the foundations of this project have been instrumental in shaping the research, and I remain greatly appreciative of his confidence in me.

I am especially grateful to my beloved parents, whose unwavering love and encouragement have supported me throughout my life. They instilled in me the value of education and encouraged my ambition of completing a PhD. This achievement is as much theirs as mine, and I hope it reflects their sacrifices and the dreams they entrusted to me.

I am sincerely thankful to my younger sister, *Kaneez Fatima*, whose support has meant more than she may ever realise. Despite the difference in our ages, she understood my challenges with remarkable maturity and was always willing to listen in difficult moments. Her relentless support brought comfort and perspective when I needed it most.

I am grateful to my friends for their encouragement and presence, and to everyone who played a part, however small, in bringing this research to fruition.

Finally, I would like to thank my husband, *Shahzaib*, who has understood the demands of this PhD more closely than anyone. His patience, support, and unwavering presence have anchored me throughout these years, and the life we have built together remains one of the most meaningful outcomes of this journey.

For my beloved parents, Shahid and Tasnim,

to whom I owe everything.

University of Glasgow

College of Science & Engineering

Statement of Originality

Name:

Registration Number:

I certify that the thesis presented here for examination for a PhD degree of the University of Glasgow is solely my own work other than where I have clearly indicated that it is the work of others (in which case the extent of any work carried out jointly by me and any other person is clearly identified in it) and that the thesis has not been edited by a third party beyond what is permitted by the University's PGR Code of Practice.

The copyright of this thesis rests with the author. No quotation from it is permitted without full acknowledgement.

I declare that the thesis does not include work forming part of a thesis presented successfully for another degree.

I declare that this thesis has been produced in accordance with the University of Glasgow's Code of Good Practice in Research.

I acknowledge that if any issues are raised regarding good research practice based on review of the thesis, the examination may be postponed pending the outcome of any investigation of the issues.

Signature: _____

Date: _____

Contents

Abstract i

Acknowledgements ii

Statement of Originality iv

List of Tables x

List of Figures xi

List of Abbreviations xiv

List of Symbols xvi

1 Introduction 1

1.1 Background and Motivation 1

1.1.1 Internet of Things and Assistive Technologies 1

1.1.2 The Shift Towards Personalised Services 2

1.1.3 Smart Mirrors as an Emerging Interface 3

1.1.4 The Internet of Mirrors (IoM) as a Unified Connected Platform 4

1.2 Research Scope 5

1.3 Problem Statement 5

1.3.1 Conceptualising the IoM Ecosystem Vision and Architecture . . 6

1.3.2 Realising the IoM Across Distributed Infrastructure 6

1.3.3 Scaling the IoM for Diverse Computational Demands 6

1.3.4 Enabling Intelligent IoM Operation Within the Broader IoT Ecosystem 7

1.4 Aims, Research Questions and Objectives 7

1.4.1 Aims 7

1.4.2 Research Questions 7

1.4.3 Objectives 8

1.5 Contributions and Research Outcomes 9
1.5.1 Contributions 9
1.5.2 Research Outcomes 10
1.5.3 Wider Research Impact 11
1.6 Thesis Outline 12

2 The IoM: Concept, Architecture, and Vision 14
2.1 The IoM Concept and Ecosystem 14
2.1.1 Smart Mirrors 16
2.1.2 Connectivity 18
2.1.3 Applications and Services 18
2.1.4 Novel Features 19
2.2 IoM Architecture and Stakeholders 21
2.2.1 Stakeholders and Interactions 22
2.2.2 IoM Node Hierarchy 23
2.3 IoM Use Cases and Application Domains 24
2.3.1 Use Case Scenarios 24
2.3.2 Unique Advantages 25
2.4 Summary 30

3 Background and Literature Review 31
3.1 Evolution of Smart Mirror Platforms 31
3.2 Sensing, Computer Vision, and Personalisation Technologies 35
3.2.1 Multimodal Sensing and Data Heterogeneity 36
3.2.2 Computer Vision and Personalisation 38
3.3 Connectivity and Computing Architectures for IoT Systems 39
3.3.1 Wireless Connectivity for IoT 40
3.3.2 The Computing Continuum 41
3.4 Task Distribution and Adaptive Orchestration in Distributed IoT Systems 44
3.4.1 Task Offloading and Context-Aware Scheduling 44
3.4.2 RL for Task Offloading and Service Orchestration 45
3.5 Summary and Research Gap 47

4 Proof-of-Concept and System-Level Evaluation of the IoM as a Distributed System 50
4.1 Introduction 50
4.2 Representative IoM Use Case: Automated Dental Smile Analysis 51

4.2.1 Clinical Motivation and Application Requirements 52
4.2.2 Methodology 54
4.2.3 Use Case Evaluation and Discussion 58
4.3 Initial IoM Performance Evaluation: Computational Placement Between Edge and Cloud 61
4.3.1 System Architecture 63
4.3.2 Methodology 65
4.3.3 Results 67
4.3.4 Discussion and Implications 70
4.4 Physical Testbed Evaluation: Computational Placement Across the IoM Tier Hierarchy 72
4.4.1 System Design 73
4.4.2 Experimental Setup 76
4.4.3 Results and Discussion 79
4.5 Summary 85

5 Context-Aware Task Distribution (CATD) for IoM Systems 87
5.1 Introduction 87
5.2 System Architecture and Model 90
5.2.1 IoM Network Architecture 90
5.2.2 Node-Tier Capability Model 92
5.2.3 Task Characterisation and Profiles 93
5.3 CATD Framework Design 95
5.3.1 Scoring System Design 95
5.3.2 CATD Algorithm 97
5.4 Methodology and Simulation Design 100
5.4.1 Simulation Environment and Implementation 100
5.4.2 Performance Metrics 104
5.4.3 Baseline Algorithms 104
5.4.4 Experimental Testing Framework 105
5.5 Results and Discussion 108
5.5.1 Category 1: Algorithm Validation 108
5.5.2 Category 2: Performance Scaling and Capacity Analysis 111
5.5.3 Category 3: System Resilience and Fault Tolerance 113
5.5.4 Category 4: Context-Awareness and Intelligent Adaptation . . . 115
5.5.5 Broader Implications 117

5.6 Summary 117

6 Bridging the IoM and the Internet of Sounds (IoS): An RL-Driven Multimodal Task Orchestration Approach 119

6.1 Introduction 119
6.2 System Architecture and Model 121
6.2.1 IoM Architecture 121
6.2.2 Orchestration Layer 122
6.2.3 IoS as an External Ecosystem 124
6.3 Problem Formulation 124
6.3.1 Task Profiles 125
6.3.2 Representative Use Case 126
6.3.3 Modality-Hardware Capability and State Space 126
6.3.4 Context Mode Determination 128
6.3.5 Action Space and Constraint Masking 129
6.3.6 Packet Loss and End-to-End Latency 130
6.3.7 Reward Function 132
6.3.8 Optimisation Objective 134
6.4 Methodology and Simulation Design 134
6.4.1 Training Algorithm 134
6.4.2 Episode Structure 136
6.4.3 Simulation Setup 136
6.4.4 Baseline 137
6.4.5 Evaluation Design 138
6.5 Results and Discussion 138
6.5.1 Validation 138
6.5.2 Load vs. Baseline 139
6.5.3 Modality-Heterogeneity Sensitivity 141
6.6 Summary 142

7 Conclusion and Future Work 144

7.1 Reflection on the Research Objectives 144
7.2 Future Work 147
7.2.1 Physical Testbed Expansion for Multi-User IoM Deployments . 147
7.2.2 Diverse Use Case Development and Validation 147
7.2.3 Multimodal Representation and Cross-Ecosystem Interoperability 148

7.2.4 Privacy Preservation and Security Protocols for IoM Deployments 149

Bibliography **150**

List of Tables

2-1 High-level comparison of the IoM with existing technology approaches at interface and ecosystem levels. 27

2-2 IoM features mapped to accessibility requirements. 29

3-1 Commercially available smart mirrors and their features. 34

3-2 Characteristics of computing paradigms within the IoT computing continuum. 43

3-3 Progression of task distribution and adaptive scheduling approaches. . . 44

3-4 Summary and comparison of the state of the art, identified research gaps, and positioning of the Internet of Mirrors. 49

4-1 Summary of computational placement strategies. 77

4-2 Experimental design summary for the physical IoM testbed evaluation. 78

5-1 Methodological comparison of CATD with representative task-distribution and scheduling approaches. 88

5-2 Task profile categories and characteristics. 93

5-3 Testing Setup and Configuration 106

5-4 Task completion rates (%) across dynamic operational phases. 116

6-1 Task Profile Categories and Characteristics 125

6-2 Use-Case Pipeline Mapping 127

6-3 Context Modes 129

6-4 Per-Mode Reward Weights 133

List of Figures

1-1 A summary of the thesis structure, mapping the chapters and publications to the IoM roadmap. 13

2-1 Possible interactions between the IoM and other ecosystems under the IoT framework. 15

2-2 Overall IoM ecosystem. 16

2-3 Functional relationship between the physical smart mirror interface, its sensing and processing components, and wider IoM services. 17

2-4 Multidisciplinary context of the IoM. 21

2-5 Interaction possibilities between consumers, professionals and service providers. 22

2-6 Visualisation of IoM use cases and possible stakeholder interactions. . . 25

2-7 Comparison of user interaction overhead between the IoM, standalone smart mirrors and current assistive technologies. 29

3-1 A typical standalone IoT-enabled smart mirror schematic. 33

4-1 Presence of excessive gingival display in (a) before orthodontic treatment and an ideal lip line in (b) following treatment. The blue line indicates the location of the lip line. 52

4-2 Examples of the principal smile-arc configurations: (a) ideal, (b) flat, and (c) reversed or inverted. The blue line follows the curvature of the lower border of the maxillary teeth, while the orange line illustrates deviation from this curvature in the flat and reversed cases. 52

4-3 Diagram of the dental smile analysis pipeline from data collection to preprocessing to CNN training. 56

4-4 UI layout for an assistive diagnostic application. 57

4-5 Image-acquisition hardware and visual feedback used in the dental smile-analysis interface. 57

4-6 Training and validation accuracy of the CNN model. 59

4-7 Training and validation loss of the CNN model. 59

4-8 Box plots of the test accuracy obtained using models trained on real, AI-generated, and combined datasets. 59
4-9 Confusion matrix for a representative run of the combined-data CNN model. 60
4-10 Edge computing architecture for dental smile analysis. 64
4-11 Cloud computing architecture for dental smile analysis. 65
4-12 Latency component breakdown for edge and cloud computing systems. 67
4-13 Latency component comparison for edge vs cloud. 69
4-14 Comparison of image-classification latency and additional network latency for the edge and cloud configurations. 70
4-15 Computational placement strategies across the IoM tier hierarchy. . . . 75
4-16 Physical IoM testbed configuration showing consumer, professional, and hub nodes 78
4-17 End-to-end latency decomposed by pipeline stage and network technology across the four computational placement strategies. 79
4-18 Network transfer overhead decomposed by inter-tier hop direction for each placement strategy. 81
4-19 Combined Resource utilisation (%) across IoM tiers per placement strategy. 82
4-20 Scalability analysis under concurrent consumer load. 83

5-1 IoM system architecture showing the integration of its conceptual layers with the three-tier physical node hierarchy. 91
5-2 IoM topology showing the spatial distribution of Hub, Professional, and Consumer nodes and the possible cross-tier interactions. 91
5-3 CATD Algorithm Flowchart. 99
5-4 Simulation Framework Architecture and Module Interactions 101
5-5 Distribution of task allocation percentages across different node types for CATD and baseline algorithms. 109
5-6 Allocation consistency and performance variance across different seeds. 110
5-7 Task completion rate against normalised system load. 111
5-8 Task completion rate versus resource-intensive workload percentage. . . 112
5-9 Resilience analysis under varying node failure conditions. 113
5-10 Performance analysis in response to bandwidth degradation. 115
5-11 Data Sensitivity and Locality Analysis 116

6-1 IoM system architecture, showing the tier hierarchy and functional layers, the Intelligent Orchestration Layer, and its connection to the co-deployed IoS and the broader IoX vision. 122

6-2 Simulation architecture, showing the environment interface, constraint and reward modules, and the agent-training loop. 137

6-3 Assigned-tier distribution by task profile under the orchestrated condition at medium load, grouped by ecosystem of origin. 139

6-4 Task completion rate versus load condition, orchestrated system compared against the cloud-offload baseline. 140

6-5 Latency and reliability for Live-Interactive tasks versus load condition: p95 latency, deadline satisfaction, and packet-loss compliance. 140

6-6 Task completion and defer rate versus audio-provisioning probability, at fixed medium load. 141

List of Abbreviations

5G	Fifth-Generation
AI	Artificial Intelligence
AM	Ambient Monitoring
AR	Augmented Reality
BB	Background-Batch
CATD	Context-Aware Task Distribution
CNN	Convolutional Neural Network
CPE	Customer Premises Equipment
CPU	Central Processing Unit
CV	Computer Vision
DL	Deep Learning
DQN	Deep Q-Network
E2E	End-to-End
GFLOPS	Giga Floating-Point Operations Per Second
GPU	Graphics Processing Unit
IoM	Internet of Mirrors
IoMT	Internet of Medical Things
IoMusT	Internet of Musical Things
IoS	Internet of Sounds
IoT	Internet of Things
IoV	Internet of Vehicles
IoX	Internet of X

LI	Live-Interactive
LPWAN	Low-Power Wide-Area Network
ML	Machine Learning
NTP	Network Time Protocol
RL	Reinforcement Learning
RR	Round Robin
SD	Standard Deviation
UI	User Interface

List of Symbols

$\mathcal{A}$	Action set.
$\mathcal{A}(s_t)$	Feasible action set in state s_t.
a_t	Action selected at decision step t.
a^*	Greedy next action selected by the online Q-network.
Access$(\cdot)$	Function defining the destination tiers accessible from a task's entry tier.
$\mathcal{B}$	Experience replay buffer used for Double-DQN training.
$B(w)$	Bandwidth associated with network path or link w.
$b(w)$	Currently available bandwidth of link w.
Bernoulli(p_{aud})	Bernoulli distribution with success probability p.
C	Set of Consumer-tier nodes.
$Cap_r(n)$	Capacity of node n in resource dimension r.
C_{lat}	Upper clamp applied to the latency-ratio reward term.
$D(t)$	End-to-end deadline associated with task t.
$d(Src(t), n)$	Euclidean distance between the source of task t and candidate node n.
$Dst(t)$	Delivery destination of task t.
defer	Action representing deferral of a task.
$e(t)$	Ecosystem of origin of task t.
$\mathbb{E}_\pi[\cdot]$	Expectation under policy π.
H	Set of Hub-tier nodes.
i	Context-mode index.

$\mathbf{1}[\cdot]$	Indicator function, equal to 1 when its condition is true and 0 otherwise.
K	Decision horizon, i.e. the number of task-allocation decisions in one bounded reinforcement-learning episode.
k	Reward-component index.
$L(w)$	Latency associated with network path/link w.
$\overline{L}_{sys}$	Average system latency from task generation to completion.
$Load(n)$	Current processing load of node n.
$Load(n, t')$	Processing load of node n at simulation time t'.
$Load(w)$	Current traffic load carried by link w.
$local$	Distance threshold within which a candidate node is considered local to a task source.
$\ell(t)$	End-to-end latency experienced by task t.
$\ell_{uplink}(t)$	Uplink-latency component of task t.
$\ell_{route}(t)$	Routing-latency component of task t.
$\ell_{proc}(t)$	Processing-latency component of task t.
$\ell_{downlink}(t)$	Downlink-latency component of task t.
M	Set of task modalities.
$m(t)$	Modality of task t.
mode(t)	Context mode associated with task t.
N	Set of all IoM processing nodes, $N = H \cup P \cup C$.
$N_{acc}(t)$	Set of nodes permitted by the stakeholder-tier access constraint for task t.
$N_m(t)$	Set of nodes capable of processing the modality of task t.
$N_{\mathbf{tier}}$	Set of nodes belonging to a specified IoM tier.
$NQ(w, t)$	Network-quality score for task t over path w.
$n(t)$	Processing node assigned to task t.
$\oplus$	Concatenation of successive segments of a task's end-to-end service path.

P	Set of Professional-tier nodes.
p_{aud}	Probability that a Professional- or Hub-tier node is provisioned with audio-processing capability.
$PC(t)$	Performance-requirement profile of task t.
$PC(t).b$	Bandwidth requirement of task t.
$PC(t).l$	Latency requirement of task t.
$PL(w, t)$	Packet-loss probability for task t on link w.
$PL(t)$	End-to-end packet-loss probability for task t.
Path(t)	End-to-end service path followed by task t.
$q(n)$	Queue depth of node n.
$Q(s, a; \theta)$	Action-value function parameterised by network weights θ.
r	Resource-dimension index, $r \in \{c, m\}$, denoting compute and memory.
r_t	Reward obtained at decision step t.
$R(n, t)$	Resource-capability score of node n for task t.
$R(w)$	Reliability of network link w.
$RC_r(t)$	Requirement of task t in resource dimension r.
$R_k(t)$	Reward component k for task t.
$R_{lat}(t)$	Latency component of the orchestration reward.
$R_{dead}(t)$	Deadline-satisfaction component of the orchestration reward.
$R_{eff}(t)$	Resource-efficiency component of the orchestration reward.
$R_{loss}(t)$	Packet-loss component of the orchestration reward.
$Rel(w)$	End-to-end reliability of network path w.
rank(n)	Rank of node n according to its IoM processing tier.
s_t	System state observed at decision step t.
s_{t+1}	System state following the action taken at decision step t.
$SC(t)$	Data-sensitivity level associated with task t.

$sensitive$	Multiplicative CATD score factor applied to sensitive tasks assigned locally.
$Src(t)$	Source location or originating endpoint of task t.
$Score(t, n, w)$	CATD assignment score for placing task t on node n via path w.
t	Representative task index.
t'	Continuous simulation-time variable.
T	Total simulation time.
$\text{tier}(n)$	IoM tier membership of node n.
$u(n)$	Utilisation of node n.
$U_{\text{link}}(w, t)$	Clamped utilisation of link w while handling task t.
$U_{\mathbf{tier}}$	Time-weighted average computational resource utilisation of an IoM node tier.
W	Set of network links.
w	Network link or instantiated network path, as specified by the relevant formulation.
$w(t, n)$	Network path between the source of task t and candidate node n.
$w_k(t)$	Reward weight applied to component k for task t.
$w_k^{(i)}$	Weight assigned to reward component k under context mode i.
y	Double-DQN target value used to update the online Q-network.
$\alpha(t)$	Indicator of whether task t is generated during an active session.
$\chi(t)$	Trigger type associated with task t.
ε	Exploration probability used by the ε-greedy action-selection policy.
η	Task completion rate, defined as the percentage of generated tasks successfully completed.
γ	Reinforcement-learning discount factor.
κ	CATD tier-widening retry counter.
$\kappa_{\max}$	Maximum number of CATD tier-widening allocation attempts.
λ	Task arrival rate.

λ_{rel}	Relative task-arrival load.
λ_{loss}	Packet-loss penalty coefficient.
$\mu_{\mathbf{sys}}$	Aggregate service rate of the simulated IoM system.
$\phi(n, m)$	Binary processing-capability function indicating whether node n supports modality m.
π	Task-orchestration policy.
π^*	Optimal task-orchestration policy.
ρ	Normalised system load.
$\sigma_r(t)$	IoM entry node associated with task t.
τ_H	Compute-requirement threshold for Hub-tier processing.
τ_P	Compute-requirement threshold for Professional-tier processing.
θ_m	Packet-loss threshold for modality m.
θ	Parameters of the online Q-network.
θ^-	Parameters of the target Q-network.
$\psi(\text{mode})$	Deadline-miss penalty magnitude associated with a context mode.
$\psi_{defer}(\text{mode})$	Deferral penalty magnitude associated with a context mode.
ξ	Congestion-sensitivity exponent used in the packet-loss model.

Chapter 1

Introduction

1.1 Background and Motivation

1.1.1 Internet of Things and Assistive Technologies

The emergence of the Internet of Things (IoT) has supported a shift towards increased interconnectivity and automation across sectors including healthcare, transportation, and smart cities [1] [2]. The increasing number of Things, comprising internet-connected embedded systems, has created opportunities for everyday physical objects to exchange information and operate collectively towards shared goals such as improved energy efficiency, enhanced health monitoring, and more accessible living environments [3] [4] [5]. As connected devices have become more widely deployed across homes, workplaces, and industrial environments, specialised ecosystems have emerged under the IoT umbrella, such as the Internet of Vehicles (IoV) [6], the Internet of Medical Things (IoMT) [7], the Internet of Sounds (IoS) [8] and the Internet of Musical Things (IoMusT) [9]. These developments illustrate a progression from individually connected devices towards ecosystems in which multiple devices, services, and stakeholders interact through shared communication and computing infrastructure.

This progression is particularly relevant to healthcare and assisted living, where connected technologies increasingly support users with diverse requirements and abilities. Approximately 1.3 billion people worldwide live with disabilities, and access to IoT-enabled healthcare and assistive technologies remains uneven across this population [10]. Many digital health systems have been developed without inclusive design principles from the outset, resulting in interfaces and interaction mechanisms that remain inaccessible to some users [11]. Accessibility is also affected by the cognitive effort required to interpret information and interact with digital systems [12][13]. Where users are required to manage several independent assistive devices, each with its own interface and interaction pattern, frequent context switching and fragmented information can increase this burden [14] [15]. Assistive technologies therefore need to account not only for the functionality delivered to the user but also for how that functionality is

accessed and coordinated across different services.

Supporting multiple assistive services within connected environments introduces corresponding requirements at the system level. Existing IoT-enabled healthcare systems continue to face challenges associated with interoperability, data security, and reliable real-time transmission [16]. These challenges become increasingly relevant when services depend on data exchange between heterogeneous devices or when information must move between users and remote computing infrastructure. In such environments, the presence of individually capable smart devices does not by itself provide an integrated service. Reliable connectivity and interoperability are required to allow devices and services to exchange information, coordinate their operation, and make distributed resources available where they are needed [17]. This distinction between standalone smart technologies and interconnected systems highlights the need for continued development of interoperable architectures and reliable communication mechanisms to translate advances in IoT into integrated and accessible assistive environments in which multiple services can operate around the same user [18].

These requirements extend across healthcare, assisted living, and related personalised service domains. Elective and specialised healthcare services in areas such as dentistry [19] [20], dermatology, and optometry [21] provide examples of domains in which user-specific information and interaction with professionals can increasingly be supported through connected technologies [17]. Related developments in consumer-facing beauty services provide a further motivating context in which digital technologies are being used to support individualised interaction and service delivery [22]. Together, these domains provide a diverse application space in which connected technologies must support different users, services, and interaction requirements.

1.1.2 The Shift Towards Personalised Services

Across these application domains, there has been a broader shift towards services that respond to the characteristics, preferences, and needs of individual users. Within healthcare, IoT-enabled systems increasingly support personalised approaches through applications such as remote monitoring, individualised treatment strategies, and data-driven service delivery [23]. A similar trend is evident in consumer-facing beauty services, often termed beauty tech, where the integration of technologies like IoT and Augmented Reality (AR) is used to provide services such as personalised product recommendations and cosmetic colour customisers [24]. This demand for personalisation is highlighted in an Accenture survey, which found that 75% of consumers are more likely to make a purchase when recommendations are tailored to them [25]. A further industry report found that 77% of consumers had chosen, recommended, or paid more for a brand that offered a personalised service or experience [26]. Together, these developments demonstrate a wider movement away from generic service delivery towards services that respond more closely to individual users.

However, the technologies supporting this personalisation demand have often developed as separate tools designed around individual functions. Within beauty tech, examples remain limited to information delivery and product customisation, with services such as portable skin analysers, cosmetic curation apps, and skin analysis apps [27] [22]. A similar pattern is evident in smart healthcare, where personalised capabilities are often delivered through separate applications and services developed for individual functions or stages of care [28]. As a result, advances in individual technologies do not necessarily translate into a unified service experience in which information and functionality can be shared across different user needs and contexts.

1.1.3 Smart Mirrors as an Emerging Interface

The need for a multi-purpose platform is particularly relevant given the evident demand for personalisation and the visual nature of services across the cosmetics and beauty industry and the elective healthcare sector. An existing smart system that has the potential to support these services is a smart mirror. A smart mirror refers to a user-facing intelligent interface that retains the reflective function of a conventional mirror while augmenting it with digital display and interaction capabilities. Depending on the application, sensing and processing functions may be embedded within the mirror itself or provided through connected external devices [29]. Their common features include the visualisation of multimedia and contextual information alongside interaction mechanisms such as voice and touch. In recent advancements, smart mirrors have been explored across a range of user-centred applications, including virtual makeup and personalised beauty services [30], real-time skin assessment [31], health monitoring and personalised medicine [32], contactless vital-sign estimation [33], and smart-home and environmental interaction [34]. Their growing commercial presence further demonstrates interest in smart mirrors as an emerging interface across consumer, retail, and healthcare settings [35].

Despite the growing range of smart mirror applications, current smart mirror systems are limited to standalone systems. Although individual smart mirror devices may incorporate data from external sensors, there remains limited consideration of data exchange and coordination between smart mirrors themselves or with other IoT infrastructure. This limitation is reinforced by the only study identified in the literature that systematically investigated the limitations of existing smart mirror technologies through expert interviews and surveys across the relevant application sectors [36]. The study reported three key limitations, namely insufficient personalisation, limited data exchange between smart devices and the smart mirror, and limited support for advanced real-time visualisation such as augmented reality try-on. Correspondingly, representative smart mirror academic implementations remain centred on the functionality of an individual mirror, including IoT-enabled, mobile-programmable, and modular platforms [37; 38; 39]. The limitation is therefore not simply a lack of connectivity within an in-

dividual mirror, but the absence of coordinated multi-mirror operation and shared services and resources across a wider connected environment. Together, these limitations reinforce the broader challenges mentioned in 1.1.1 and 1.1.2, including the demand for more personalised service delivery, fragmentation across existing technologies, and limited interoperability between connected systems. Extending smart mirrors beyond standalone operation could therefore allow information and functionality to be shared across devices and services, supporting an interoperable and multi-purpose interface centred around the needs of the user.

1.1.4 The Internet of Mirrors (IoM) as a Unified Connected Platform

The challenges identified across the preceding subsections collectively highlight the need to move beyond fragmented, standalone technologies towards a more integrated approach to personalised and assistive service delivery. Such an approach needs to accommodate users with diverse requirements, support multiple services through a common user-facing interface, and enable information and functionality to be shared rather than remaining restricted to individual devices or applications. The visual and interactive capabilities of smart mirrors make them well suited to act as this interface, but their current standalone operation limits the extent to which they can support services across different devices, users, and contexts. Addressing these limitations therefore requires more than incremental or hardware-level improvements to individual smart mirrors and instead requires consideration of how these devices can operate collectively.

To respond to these converging challenges and opportunities, this thesis introduces a unified vision for an interconnected system of smart mirrors, coined as the Internet of Mirrors (IoM), under the wider IoT framework. Central to this vision is the interoperability and interconnectivity of smart mirrors within an ecosystem, distinguishing the IoM from existing standalone systems and providing a basis for data exchange, distributed processing, and responsive personalised services. In this way, the proposed vision moves beyond device-level connectivity towards ecosystem-level coordination, allowing mirrors to exchange data and services and access distributed computational resources through a common architecture.

The role of the mirror within this concept extends beyond that of a display device. As a familiar and routinely used user-facing interface, the mirror naturally co-locates the user with visualisation, sensing, and interaction, while its fixed presence supports hands-free access and repeated interaction within everyday environments. Consequently, the IoM is centred on smart mirrors because they provide the recurring user-facing interface through which distributed devices, computational resources, and services can be brought together. The IoM therefore brings together the visual and user-centred characteristics of the smart mirror with the connectivity and computing

capabilities of IoT, through which otherwise fragmented devices, information, and services could be brought together around the needs of the user. Grounded in the health, beauty, and assistive service domains discussed throughout this section, the IoM is envisioned as a common platform through which these services could be integrated while smart mirrors operate as part of the wider connected environment. Its position within the broader IoT landscape also provides a basis for interaction with other connected devices, services, and IoT ecosystems, extending the vision to assisted living and wider smart environments.

While this vision encompasses a broad range of possible services and smart environments, the specific aspects investigated within this thesis are defined in the following section.

1.2 Research Scope

As the IoM is introduced here as a new concept, the establishment of its foundations forms the basis of the research. Therefore, the research scope of this thesis is centred on the IoM as an interconnected ecosystem and on the foundational, architectural, and operational considerations required to support its wider vision.

At the foundational level, the scope encompasses the structure of the ecosystem, its constituent elements, stakeholders, and the requirements arising from the users and services it is intended to support.

At the architectural and operational levels, the scope extends to the infrastructure through which IoM services are delivered and the system-level considerations associated with computation, communication, data heterogeneity, scalability, and interaction with other IoT ecosystems. These aspects are considered at system level, in relation to the differing requirements that may arise across users, applications, and deployment environments.

The application scope is grounded in the personalised and assistive service domains that motivate the IoM, but it is bounded to representative use cases that provide concrete workloads through which the implications of these requirements on the underlying ecosystem can be examined.

1.3 Problem Statement

Within the research scope defined, realising the IoM as an interconnected ecosystem introduces a set of related challenges that span from its conceptual foundations to its operation within the wider IoT landscape. These challenges concern the conceptualisation of an ecosystem vision and architecture, its realisation across distributed infrastructure, its ability to accommodate increasingly diverse computational demands as it scales, and its intelligent operation alongside other IoT ecosystems. The following

subsections define each of these problems.

1.3.1 Conceptualising the IoM Ecosystem Vision and Architecture

The first challenge lies in establishing what an interconnected ecosystem of smart mirrors should comprise, how it should be structured, and who it should serve. No existing framework conceptualises smart mirrors as an interconnected ecosystem capable of supporting personalised and assistive services across health, beauty, and related domains in a unified manner. Consequently, the constituent elements, stakeholder roles, architectural relationships, and user-focused requirements of such an ecosystem remain undefined. These requirements must account for users of diverse abilities and needs, as well as the differing service and deployment contexts across the envisioned application landscape. This conceptual foundation is needed so that subsequent technical formulations have a coherent frame of reference to build upon.

1.3.2 Realising the IoM Across Distributed Infrastructure

Translating the IoM ecosystem vision and architecture into practical operation across distributed infrastructure introduces a different set of system-level considerations. The diversity of services envisioned for the IoM may require sensing, processing, and visualisation to be distributed across smart mirrors and remote computing resources, while the computational capacity and communication characteristics of these resources may differ substantially. The placement of computation across this infrastructure therefore introduces trade-offs between processing capability, communication delay, network overhead, and service responsiveness, which may vary according to the workload and deployment conditions. Understanding these trade-offs is necessary to determine how the IoM architecture can support its intended services under practical operating conditions.

1.3.3 Scaling the IoM for Diverse Computational Demands

A further consequent challenge arises when considering the operation of the IoM beyond a single user or application workload. Increasing the number and diversity of users and applications introduces concurrent computational demands that may differ in processing complexity, latency sensitivity, data characteristics, and resource requirements. At the same time, the computing nodes available across the distributed infrastructure may differ in processing capability, connectivity, and resource utilisation. Consequently, a computation placement that is suitable for one task or operating condition may not remain suitable as workload characteristics and resource availability change. Establishing the operability of the IoM under these varying conditions therefore requires

consideration of how heterogeneous computational demands can be coordinated across the ecosystem while maintaining responsive service delivery and effective utilisation of the available resources.

1.3.4 Enabling Intelligent IoM Operation Within the Broader IoT Ecosystem

Extending the operation of the IoM beyond its own ecosystem introduces an additional coordination problem at the boundary with other IoT ecosystems. As the IoM is positioned within the wider IoT landscape, interaction with other sub-ecosystems introduces the possibility of services and tasks with different modalities, computational requirements, hardware dependencies, and performance constraints. While interoperability provides the basis for information and services to be exchanged across ecosystem boundaries, such interaction also introduces the problem of coordinating heterogeneous workloads across resources that may differ in capability and suitability. Establishing the operability of the IoM within this broader environment therefore requires consideration of how both native IoM services and externally originating services can be coordinated while accounting for differences in task requirements, resource capabilities, and changing system conditions.

1.4 Aims, Research Questions and Objectives

1.4.1 Aims

This thesis aims to establish the conceptual, architectural, and operational foundations of the IoM as an interconnected IoT ecosystem capable of supporting personalised and assistive services across diverse users, applications, and connected environments.

1.4.2 Research Questions

The following research questions have been formulated to guide this thesis:

Q_1: What are the requirements and considerations for realising an interconnected ecosystem of smart mirrors that supports personalised and assistive services for diverse user populations?

Q_2: What performance implications and trade-offs arise when IoM services are realised across distributed, multi-tier infrastructure under varying operating conditions?

Q_3: How can the IoM accommodate diverse and concurrent computational demands across heterogeneous computing resources as the number and variety of users and applications increase?

Q_4: How can the IoM be positioned as an intelligent and interoperable member of the broader IoT landscape while accommodating heterogeneous services and resource requirements across its own infrastructure and external IoT ecosystems?

1.4.3 Objectives

In light of the research questions above, the objectives of this thesis are as follows:

O_1: Establish the conceptual and architectural foundations of the IoM as an interconnected ecosystem supporting personalised and assistive services. This involves identifying the requirements and considerations for the IoM, defining its core components, architectural layers, and stakeholder roles, and establishing the user-centred requirements and representative use cases needed to support populations with diverse needs and abilities. The outcome of this objective establishes the conceptual and architectural foundation against which the remaining objectives are formulated.

O_2: Establish the feasibility of the IoM architecture and quantify its performance under progressively realistic conditions, addressed in three stages:

a) Develop and validate a representative proof-of-concept use case, demonstrating that the IoM can deliver an assistive service through the smart mirror interface.

b) Evaluate the latency implications of edge and cloud computation placement for IoM applications, identifying the contribution of computation and communication to service performance.

c) Evaluate computational placement across the IoM infrastructure using a physical testbed under real wireless network conditions, quantifying the performance bottlenecks and trade-offs that arise under different deployment conditions.

O_3: Develop and evaluate a context-aware task distribution framework to coordinate diverse application demands and heterogeneous node capabilities across the IoM's tiered infrastructure when operating beyond a single user or application workload. This should consider multiple task, node, and system criteria when distributing computational workloads, with the objective of maintaining responsive service delivery and efficient resource utilisation under varying operating conditions. This includes benchmarking the proposed framework against existing task offloading and scheduling mechanisms in the wider edge-cloud continuum, which do not address the specific multi-tier structure and multi-criteria demands presented by the IoM.

O_4: Develop an adaptive, learning-based orchestration mechanism that extends task coordination beyond the IoM's own boundaries, enabling the ecosystem to reason over cross-ecosystem constraints including hardware availability, modality, and

latency requirements when coordinating with external IoT sub-frameworks such as the Internet of Sounds. This includes evaluating the proposed mechanism against a fixed offloading approach to assess its effect on preserving the IoM's own operating capacity while meeting external task requirements, to demonstrate the benefit of adaptive orchestration as the IoM engages with the broader IoT landscape.

1.5 Contributions and Research Outcomes

1.5.1 Contributions

The major contributions of this thesis, aligned with the objectives outlined in Section 1.4.3, are summarised below. The corresponding publications are provided in Section 1.5.2.

- A unified architectural vision for the IoM is conceptualised, defining its core components, connectivity layer, and stakeholder roles, alongside a set of demonstratory use cases illustrating how the ecosystem serves users with diverse needs and abilities across health, beauty, and assisted living contexts. The outcome of this contribution is published in [J1] and [J2].
- A representative proof-of-concept use case demonstrates the IoM's capability to deliver an assistive dental smile analysis service through the smart mirror interface, establishing the feasibility of a single IoM application. As part of this use case, a dedicated dentist-labelled dental smile dataset was developed to address the absence of a suitable publicly available dataset for the intended gingival-display classification task. The dataset combines manually curated real images and AI-generated images and supports the development and evaluation of the proposed CNN-based analysis. The outcomes of this contribution are published in [J3] and [C1].
- A primary system-level latency analysis of edge versus cloud computation placement for IoM applications identifies where performance bottlenecks arise within the architecture, establishing an initial understanding of the trade-offs involved in distributing computation across the ecosystem. The outcome of this study was published in [C2].
- A physical testbed evaluation spanning the IoM's node tiers under real wireless network conditions validates these placement trade-offs beyond two endpoints, establishing the practical feasibility and performance limits of the IoM architecture under realistic deployment conditions. The outcome of this contribution is published in [C3].
- A Context-Aware Task Distribution (CATD) framework developed to coordinate the diverse computational demands of applications and node capabilities as the

IoM scales, distributing tasks across its tiered infrastructure according to multiple criteria simultaneously. The framework is benchmarked against existing task offloading and scheduling mechanisms in the wider edge-cloud continuum, demonstrating its comparative effectiveness for the multi-tier structure of the IoM. The outcome of this contribution is currently under review, as indicated in [J4].

- An adaptive, Reinforcement Learning (RL) based orchestration mechanism is developed to extend task coordination beyond the IoM's own boundaries, reasoning over cross-ecosystem constraints including hardware availability, modality, and latency requirements when coordinating with the Internet of Sounds. The mechanism is evaluated against a fixed offloading approach, demonstrating its benefit in preserving the IoM's own operating capacity while meeting external tasks' requirements. The outcome of this contribution is accepted for publication, as indicated in [C4].

1.5.2 Research Outcomes

The outcomes of this thesis have resulted in the following publications:

Journals:

[J1] **H. Fatima**, M. A. Imran, A. Taha, and L. Mohjazi, "Internet-of-Mirrors (IoM) for connected healthcare and beauty: A prospective vision," *Internet of Things*, vol. 28, p. 101415, Dec. 2024, doi: 10.1016/j.iot.2024.101415. Available: `https://www.sciencedirect.com/science/article/pii/S2542660524003561`

[J2] **H. Fatima**, W. S. Alharbi, A. Taha, M. A. Imran, and L. Mohjazi, "Enabling Inclusive and Accessible Living Through the Internet of Mirrors (IoM)," *IEEE Internet of Things Magazine*, pp. 1–9, 2025, doi: 10.1109/MIOT.2025.3584283. Available: `https://ieeexplore.ieee.org/document/11059764`

[J3] M. Baidachna, **H. Fatima**, R. Omran, N. Ghadban, M. A. Imran, A. Taha, and L. Mohjazi, "Mirror, Mirror on the Wall: Automating Dental Smile Analysis with AI in Smart Mirrors," *Computing&AI Connect*, vol. 1, no. 1, pp. 1–10, Jul. 2024, publisher: Scifiniti, doi: 10.69709/CAIC.2024.194138. Available: `https://scifiniti.com/3006-4163/1/2024.0002`

[J4] **H. Fatima**, L. Mohjazi, M. A. Imran, and A. Taha, "A Context-Aware Task Distribution (CATD) Framework for Internet of Mirrors (IoM) Applications," Feb. 18, 2026, submitted to *IEEE Internet of Things Journal* (under review). TechRxiv. doi: 10.36227/techrxiv.177138798.86374620/v1. Available: `https://www.techrxiv.org/doi/full/10.36227/techrxiv.177138798.86374620/v1`

Conference Proceedings:

[C1] M. Baidachna, **H. Fatima**, R. Omran, M. A. Imran, A. Taha, and L. Mohjazi, "Mirror, Mirror on the Wall: Automating Dental Smile Analysis in Smart Mirrors with CNN and Diffusion Model," in *ICML'24 Workshop ML for Life and Material Science: From Theory to Industry Applications*, Jun. 2024. Available: `https://openreview.net/forum?id=ZrlFczMbVh`

[C2] **H. Fatima**, M. A. Imran, L. Mohjazi, and A. Taha, "A Latency Analysis of Edge vs. Cloud Computing for Internet-of-Mirrors (IoM) Applications," in *1st 2024 IEEE Middle East Conference on Communications and Networking*, , Abu Dhabi, United Arab Emirates, 2024, pp. 440-445. Available: `https://ieeexplore.ieee.org/document/10882135`

[C3] **H. Fatima**, M. A. Imran, A. Taha, and L. Mohjazi, "Toward Real-Time Mirrors Intelligence: System-Level Latency and Computation Evaluation in Internet of Mirrors (IoM)," in *2026 32nd International Conference on Telecommunications (ICT)*, Thessaloniki, Greece, 2026, pp. 272-277. Available: `https://ieeexplore.ieee.org/document/11594722`

[C4] **H. Fatima**, L. Mohjazi, M. A. Imran, and A. Taha, "Bridging the Internet of Mirrors (IoM) and Internet of Sounds (IoS): An RL-Driven Multimodal Task Orchestration Approach," in *7th IEEE International Symposium on the Internet of Sounds (IEEE IS2 2026)*, Cannes, France, 28 - 30 October 2026, (Accepted for Publication) Available: `https://eprints.gla.ac.uk/393156/`

1.5.3 Wider Research Impact

Beyond the publication outputs listed above, the research undertaken within this thesis has contributed to wider academic engagement and the continued development of the IoM as an emerging research direction.

- The IoM research led to a dedicated invited IEEE Internet of Things Distinguished Webinar, entitled "The Internet of Mirrors: Reimagining Beauty and Healthcare Through Connected Intelligence", delivered as part of the IEEE IoT webinar programme in March 2026. The webinar disseminated the IoM vision and its development towards connected intelligence for personalised health and beauty services to the wider IoT research community. The webinar recording is available at `https://dx.doi.org/10.17023/bkp0-mg12`.

- The research has also contributed to the establishment of the first dedicated workshop on the IoM, "The Internet of Mirrors: Connected Intelligence for Healthcare, Beauty, and Immersive Digital Experiences", organised as part of the IEEE World Forum on Internet of Things (WF-IoT) 2026.

- The development of the IoM prototype and supporting research activities attracted additional funding, including £2,000 of Chancellor's Fund support and £2,500 of divisional funding. The research direction also supported an EPSRC summer internship.

- The IoM use case of dental smile-analysis research contributed to the University of Glasgow MOOC *Generative AI for Healthcare Students and Professionals* on Coursera, where it forms part of the clinical applications module covering generative-AI-supported data curation, CNN-based dental smile analysis, and implementation within an IoM smart mirror. This is available on: `https://www.coursera.org/learn/generative-ai-for-healthcare-students-and-professionals/home`

- The establishment of the IoM research direction has further supported the recruitment of a new PhD researcher, extending the research beyond the scope and duration of this thesis and contributing to the continued development of the IoM.

1.6 Thesis Outline

The remainder of this thesis is structured to address the research questions and objectives established in this chapter. Chapter 2 establishes the conceptual and architectural foundations of the IoM, defining the ecosystem, its core components, stakeholders, architecture, and application domains. Chapter 3 reviews the relevant literature on smart mirrors and enabling technologies, task offloading in IoT systems, and RL for IoT orchestration, situating the technical challenges associated with realising and operating the IoM within the existing research landscape. Chapter 4 investigates the practical realisation and performance implications of the IoM across distributed computing infrastructure. This is examined through three stages, beginning with a representative proof-of-concept use case, followed by a latency analysis of edge versus cloud computation placement, and concluding with a physical testbed evaluation of computation across the IoM hierarchy under real wireless network conditions. Chapter 5 considers the operation of the IoM beyond a single user or application workload and addresses the coordination of diverse computational demands across heterogeneous computing resources. A Context-Aware Task Distribution framework is developed and evaluated against representative task offloading and scheduling mechanisms. Chapter 6 extends this coordination beyond the boundaries of the IoM to consider its intelligent and interoperable operation within the broader IoT landscape. An adaptive RL-based orchestration mechanism is developed and evaluated for coordinating services across the IoM and the Internet of Sounds. Finally, Chapter 7 concludes the thesis by summarising the findings and contributions and outlining directions for future research.

Fig. 1-1 summarises this progression, illustrating how the contributions map to each chapter and the thesis as a whole.

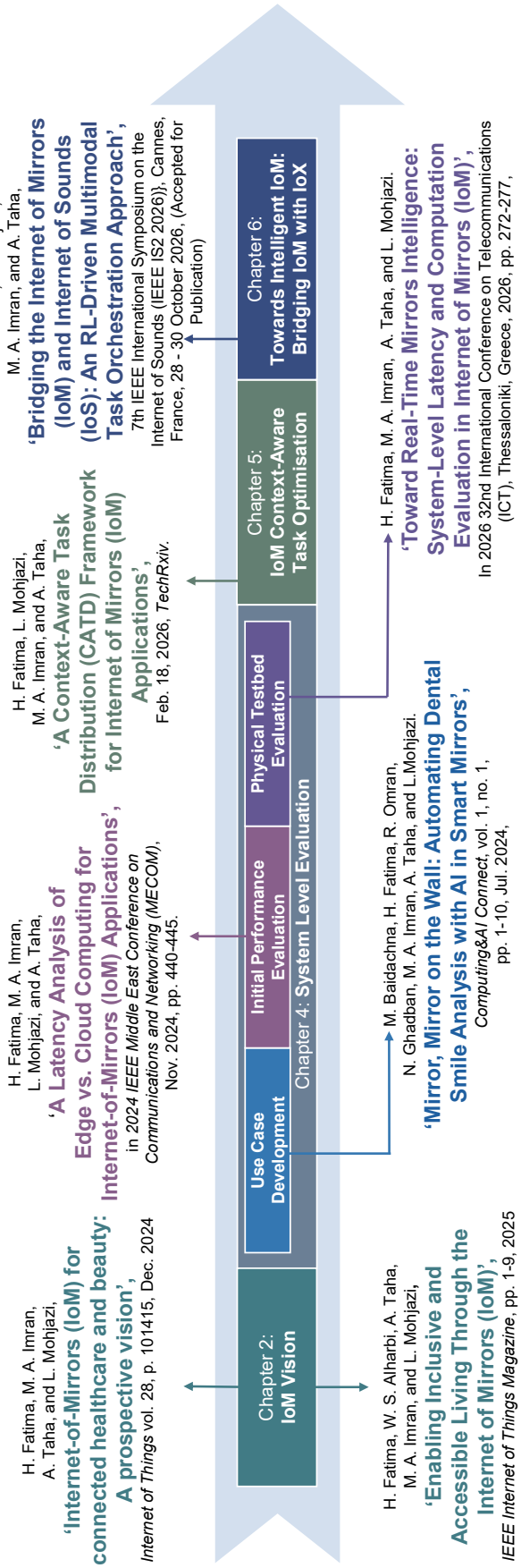


Figure 1-1: A summary of the thesis structure, mapping the chapters and publications to the IoM roadmap.

Chapter 2

The IoM: Concept, Architecture, and Vision

This chapter establishes the conceptual and architectural foundations of the IoM as an interconnected ecosystem, building on the initial IoM vision reported in [J1] and its subsequent consideration of accessibility and inclusive living in [J2]. It addresses Q_1 and meets O_1 by defining what constitutes the IoM ecosystem, who it serves, and the requirements and interactions that shape its structure. As the IoM is introduced as a new concept, the chapter first defines the ecosystem and its constituent elements, followed by its stakeholders and node hierarchy, and then considers representative use cases through which the wider vision can be illustrated. Together, these elements establish the reference framework for the IoM, while the enabling technologies and related research landscape required to support its realisation are examined later in Chapter 3. The framework established here subsequently provides the architectural basis for the technical investigation and operation of the IoM throughout the remainder of this thesis.

2.1 The IoM Concept and Ecosystem

The IoM is a novel concept that can be considered an ecosystem or a subfield of the IoT. The vision for this ecosystem is for it to interact with other ecosystems and smart systems such as the IoV, the IoMusT, the IoS or the IoMT, as well as with the Things within these systems. Fig. 2-1 visualises the possible interactions between IoT ecosystems, where Internet of X (IoX) represents an X number of other possible ecosystems within the IoT framework.

As a system, the IoM is defined as,

the integration of smart mirror interfaces and related or embedded sensors over a connected network that can support personalised services and user metric visualisation based on sensed data from interactions between the user and smart mirror or interactions between the smart mirrors themselves.

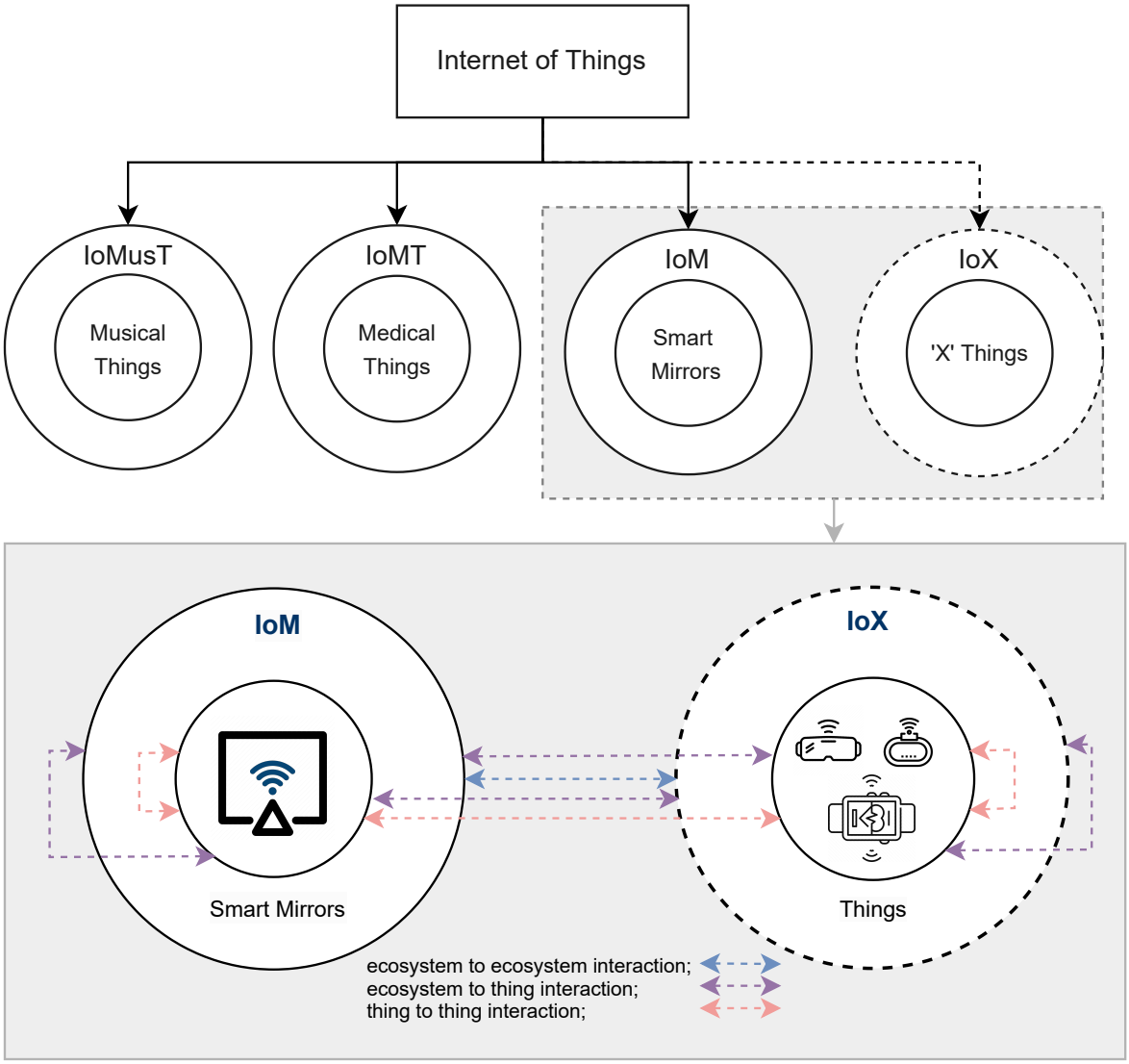


Figure 2-1: Possible interactions between the IoM and other ecosystems under the IoT framework.

In this context a smart mirror is considered to be a mirror that can sense, visualise and interact with the user and with other components of the IoM ecosystem. The ecosystem of these smart mirrors involves collective elements such as computing, storage, data exchange and resource management, all of which are garnered by unified protocols and standards.

Considering IoT ecosystems in general [40; 41; 42], the IoM can have diverse users, depending on the application and scope. At its fundamental level the IoM ecosystem involves the users, such as consumers, patients, and cosmetic and health professionals, alongside the smart mirror interfaces themselves. From an interconnectivity perspective this system can be categorised into three elements, which are detailed in the subsections below.

The elements of the IoM can also be described in terms of layers similar to a general IoT layer structure [1], where the smart mirrors act as the sensing and transmission layer and the connectivity of the system can be considered as the network and communication

layer. The services component corresponds to an application layer. The overall IoM ecosystem with its components and layers is depicted in Fig. 2-2.

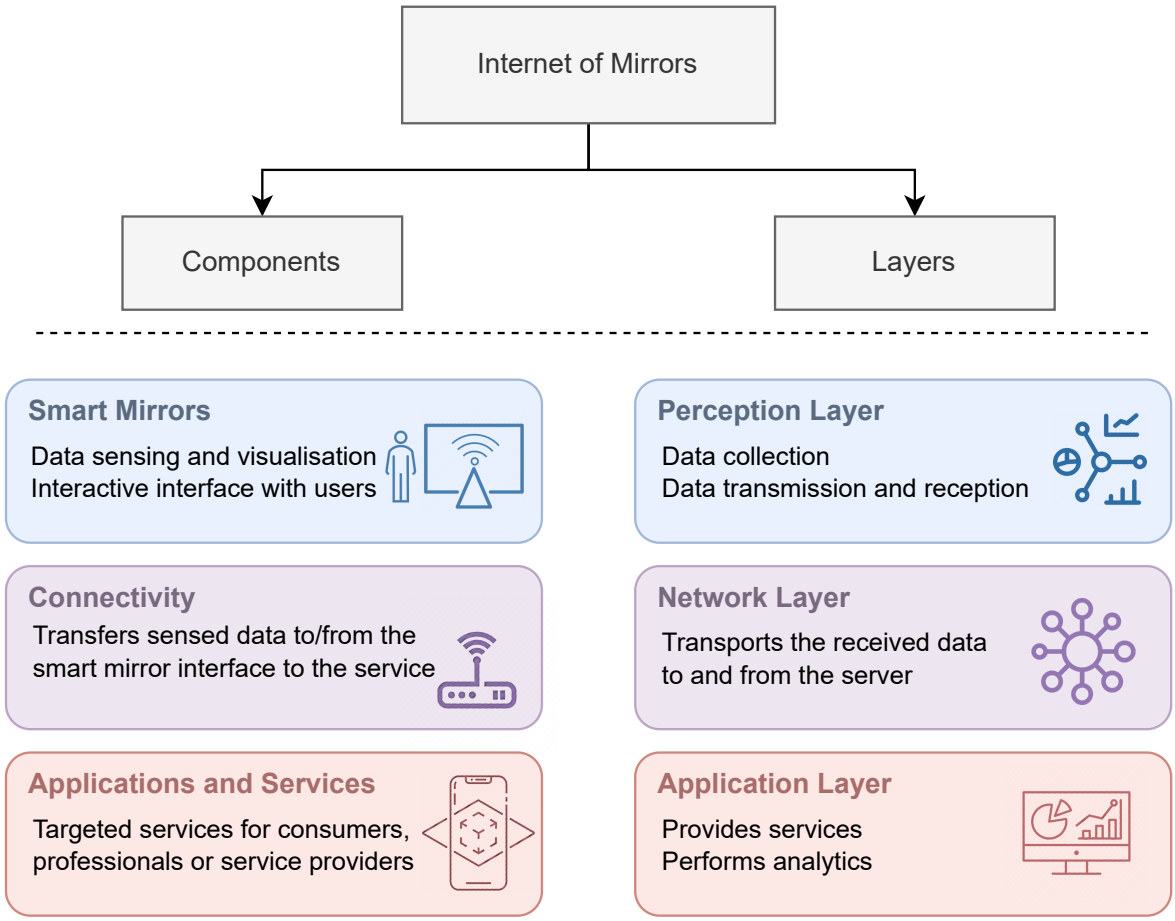


Figure 2-2: Overall IoM ecosystem.

2.1.1 Smart Mirrors

The envisioned concept of the IoM is for a number of smart mirror devices, together with their sensed data, to be interconnected and to have access to other IoT devices. Smart mirrors within the IoM can therefore be considered a subclass of Things, inheriting characteristics such as sensing, actuation, data collection and analysis, communication, and network connectivity.

These capabilities can operate at different levels within a smart system. Within this thesis, the term intelligent is used in a functional sense to describe technologies that incorporate capabilities such as sensing, processing, communication, interaction, or adaptation, rather than to denote a particular implementation technology. An intelligent sensor represents a component-level technology primarily responsible for acquiring information from the user or surrounding environment, potentially with additional local processing or communication capabilities. An intelligent surface instead refers more broadly to a physical surface augmented with sensing, interaction, actuation, or computational functionality. A smart mirror operates at a higher system level, combining

the functional role of a mirror with a user-facing interface and the sensing, processing, communication, and interaction capabilities required by its application.

A smart mirror within the IoM is therefore considered a complete system component rather than an individual sensor or intelligent surface. Its implementation may range from a connected interface relying on external sensing or computational resources to a more self-contained system with embedded sensors and processing capabilities. These concepts are complementary within the wider framework: intelligent sensors may form part of, or provide data to, a smart mirror, while intelligent surface functionality may be incorporated within the mirror interface or its surrounding environment. Regardless of its implementation, an IoM smart mirror must be capable of communicating with the relevant local or remote network to exchange data and access wider services and computational resources.

More specifically, the physical smart mirror provides the primary user-facing interface of an IoM node, combining the reflective function of a conventional mirror with the digital visualisation and interaction capabilities required by the service. Depending on the implementation, sensing and processing components may be embedded within the mirror assembly or provided through connected external devices and resources. In Fig. 2-3, the user interface therefore represents the physical mirror-facing element through which the user views information and interacts with the system, while the input and processing components provide the sensing and computational functions that support this interaction. The IoT gateway connects this physical interface and its associated components to the wider IoM network and remote services. A representative physical configuration of a standalone smart mirror, including its reflective surface, display, sensing, and embedded processing components, is presented later in Fig. 3-1.

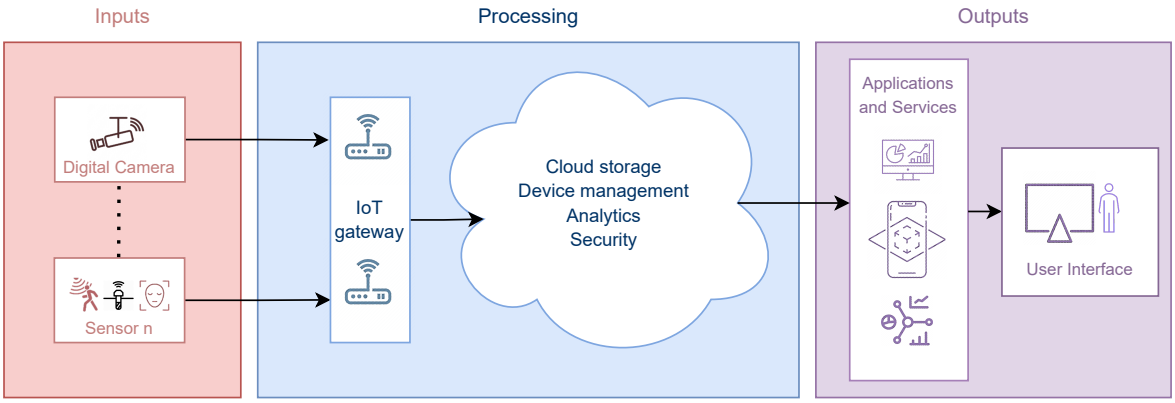


Figure 2-3: Functional relationship between the physical smart mirror interface, its sensing and processing components, and wider IoM services.

With the increasing interest in existing smart mirrors [43], the integration of the envisioned IoM could enable many new opportunities not only for the cosmetics and health industries but also for other domains such as smart homes and smart education. The visualisation aspect of the smart mirror interface also has the potential to transform user accessibility for consumers and patients by offering a more enhanced

and innovative health and beauty experience.

2.1.2 Connectivity

The connectivity of the IoM ecosystem is fundamental to its operation and development. To support the envisioned multi-directional communication between smart mirrors and from smart mirror to user, appropriate network protocols and standards must be considered. Depending on the application and domain, different local and remote wireless communication protocols can be deployed. The connectivity of the IoM would facilitate the processing of computationally heavy commands and requests to offload the computation from the embedded micro-controller in the smart mirror. Although the applications of the IoM in the cosmetics industry are not time critical in comparison to health applications, real-time use cases such as remote consultancy would still require the network to fulfil low latency, high reliability and synchronisation requirements.

The connectivity element of the IoM would not only enable interconnection between IoM smart mirrors and devices but could also facilitate additional services to support the user experience. For example, Fifth generation, or 5G, connectivity would allow smart mirrors to access and store data through optimised cloud services, to deliver high quality services and analytics through natively integrated applications, and to act as wireless access points through which other devices in the vicinity could themselves connect. Low-power wide-area networks (LPWANs) would instead support the persistent, low-bandwidth functions surrounding the mirror, such as tracking the location of personal assets, monitoring and adjusting energy consumption in real time, communicating with smart home systems including locks, thermostats and security systems, and allowing users to monitor mirror usage and performance remotely for maintenance purposes. These examples illustrate the breadth of what the connectivity layer would need to accommodate in operation, spanning latency-critical service delivery and continuous background communication.

2.1.3 Applications and Services

Whether the application domain for the IoM is assistive health, beauty or living, there are several services that can be developed from the sensed and communicated data from the smart mirrors and their connectivity features. These services can target various stakeholders including consumers, professionals and service providers, whose roles within the ecosystem and interactions with one another are detailed in Section 2.2. To establish an efficient interactive interface of the smart mirrors, these services will need to consider real-time computation and requirements.

2.1.4 Novel Features

The novelty of the IoM does not lie in introducing a new sensing modality, artificial intelligence technique, communication protocol, or individual smart mirror function. Rather, its conceptual contribution lies in treating smart mirrors as interoperable user-facing nodes of a shared IoT ecosystem, through which services, data, stakeholders, and distributed computational resources can be coordinated across different environments. This extends the smart mirror from an application-specific standalone device to an interface within a wider system whose capabilities arise from interconnection and collective operation. The following features are therefore presented as distinguishing characteristics of the IoM framework and as capabilities enabled or strengthened by this ecosystem-level organisation that establish the IoM's unique value proposition:

1. **Unified Health and Beauty Platform:** Unlike existing solutions that typically focus on single-purpose or single-domain services, IoM provides a comprehensive platform that recognises and leverages the interconnection between these domains. This integrated approach enables more holistic and personalised recommendations that consider both health and aesthetic factors.

2. **Specialised Interconnected Ecosystem:** The IoM creates a specialised network of smart mirrors designed specifically for health and beauty applications. This mirror-centric approach allows for unique capabilities such as real-time visual analysis, augmented reality overlays for beauty treatments, and a consistent User Interface (UI) across different locations (home, clinics, retail spaces). The interconnected nature of these mirrors enables a seamless experience as users move between different environments, with their personalised data and preferences following them.

3. **Continuous, Non-invasive Monitoring:** By leveraging advanced computer vision and AI technologies, the IoM framework complements existing wearable sensors by offering additional non-invasive monitoring capabilities. This approach can augment data from wearables, potentially increasing overall user compliance and providing a more comprehensive dataset for analysis. The combination of wearable data and IoM-derived information offers a richer, more nuanced picture of user health and beauty metrics.

4. **Context-aware Personalisation:** The IoM system integrates visual data with environmental factors and historical user information to provide highly personalised, context-aware recommendations. This multi-faceted approach to data analysis enables more accurate and relevant suggestions than current standalone systems.

5. **Seamless Professional Integration:** IoM facilitates easier integration of professional expertise into daily routines. It can serve as a platform for remote

consultations with healthcare providers or beauty professionals, bridging the gap between at-home care and professional services.

6. **Adaptive Learning:** Through its interconnected nature, the IoM system can learn from aggregated, anonymised data across its network. This enables the system to continuously improve its recommendations and adapt to changing trends and user needs.

These novel features address the limitations of current fragmented solutions in health and beauty technologies identified in Chapter 1. By providing a unified, interconnected platform, the IoM aims to offer a more comprehensive, personalised, and effective approach to health and beauty care. While these features present significant potential, realising them involves addressing several research challenges. The seamless interactions and data exchange across the IoM ecosystem necessitate innovations across multiple technological domains, including advanced communication and network technologies, robust security and privacy measures, enhanced interoperability protocols, standardisation efforts, and scalable AI and machine learning algorithms. Addressing these challenges forms a core motivation for the IoM vision, driving forward research and innovation across multiple disciplines.

Figure 2-4 organises this multidisciplinary context into three interrelated levels. At the application and service level, the IoM is motivated by the convergence of assistive health, beauty technology, and assisted living, where services share requirements for personalised, visual, and interactive support. These service requirements shape the interface and intelligence capabilities needed by the ecosystem, including smart mirror platforms, multimodal sensing, computer vision and personalisation. In turn, these capabilities depend on an underlying computational and system foundation through which the IoM can operate as a distributed ecosystem, encompassing wireless connectivity, the computing continuum, context-aware task distribution, and adaptive orchestration. The relationships are bidirectional where service needs define interface requirements, while the capabilities of the interface enable the services that can be delivered. Similarly, application and interface demands inform the computational mechanisms and objectives required at the system level, while the system foundation enables their practical realisation.

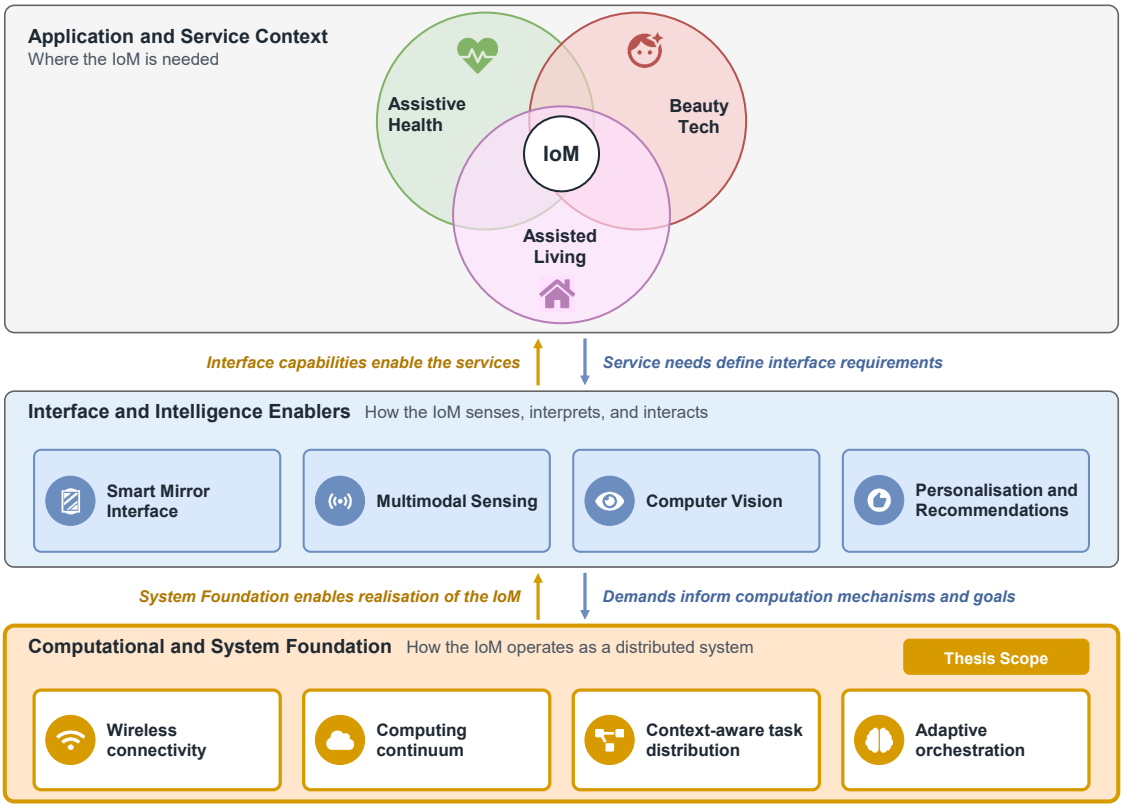


Figure 2-4: Multidisciplinary context of the IoM.

Although the wider IoM vision spans all three levels, the primary technical focus of this thesis lies within the computational and system foundation. The conceptual and architectural definition established in this chapter provides the framework for this investigation, while the subsequent technical contributions progressively examine distributed execution across the IoM infrastructure, context-aware task distribution, and adaptive orchestration across ecosystem boundaries. The application and interface levels therefore provide the motivating and enabling context within which these system-level contributions are developed and evaluated.

2.2 IoM Architecture and Stakeholders

Section 2.1 defined the elements of the IoM ecosystem and the layers through which they operate. This section describes who the ecosystem serves and how it is structured to serve them, first by identifying the stakeholders and the interactions between them, and then by introducing the node hierarchy through which those interactions are realised.

2.2.1 Stakeholders and Interactions

The realisation of the IoM would allow many everyday activities and aspects of human life to be reimagined with respect to a technological ecosystem of interactions between its stakeholders. These stakeholders are:

- **Consumers:** individuals such as customers and patients who interact with the IoM to receive personalised services and recommendations.
- **Professionals:** specialists such as dermatologists, dentists, makeup artists and other practitioners who provide expertise and services through the IoM platform.
- **Service Providers:** entities such as businesses, retail stores, clinics and healthcare providers that offer products and services integrated into the IoM.

The services described in Section 2.1.3 are delivered through interactions between these stakeholders, and the interconnectivity of the IoM allows those interactions to take place either co-located or remotely. A consumer may receive a service directly at a smart mirror in their home, at a mirror situated in a clinic or retail environment alongside the professional delivering it, or through a mirror that connects them to a professional at another location entirely. The same applies to interactions between professionals and service providers, where expertise and data can be shared across sites rather than being confined to the location at which they were generated. Fig. 2-5 shows the possible remote and co-located interactions between the different stakeholders in the IoM ecosystem, while the communication technologies supporting these interactions depend on the requirements and deployment context of the corresponding service.

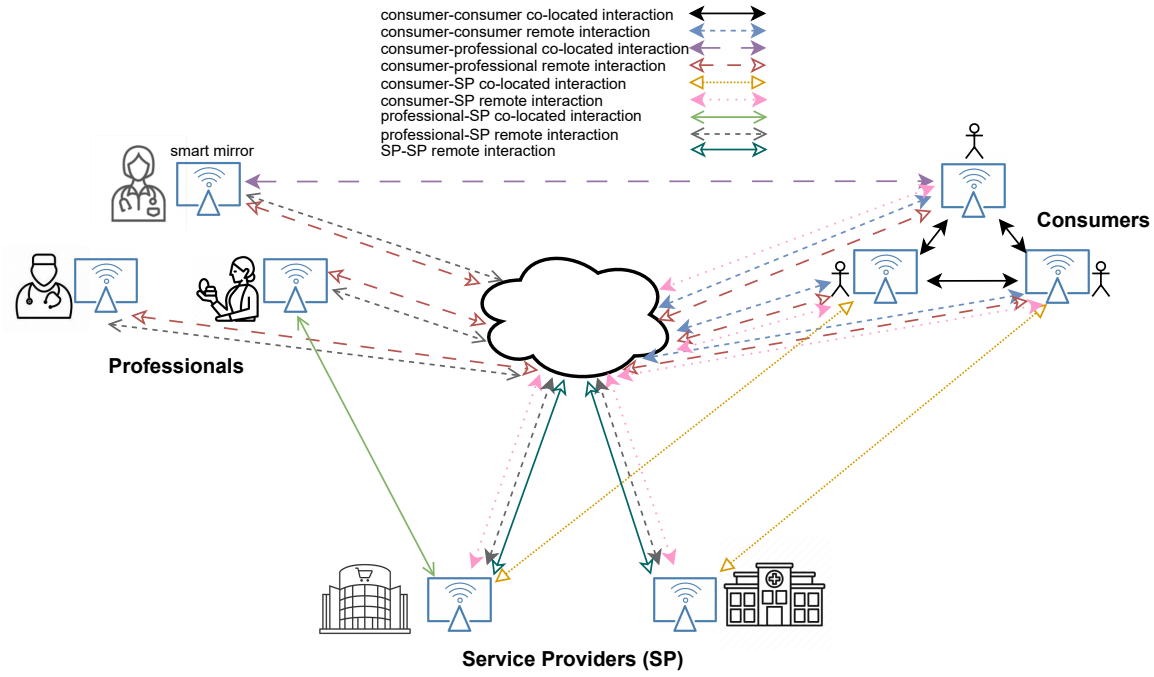


Figure 2-5: Interaction possibilities between consumers, professionals and service providers.

The interaction types shown in Fig. 2-5 represent stakeholder and service relationships rather than prescribing a specific communication technology. Their realisation depends on factors such as communication range, data requirements, latency constraints, and deployment environment. Co-located interactions may be supported through local wireless connectivity such as Wi-Fi, while remote interactions require access to wider-area network infrastructure, including cellular connectivity such as 5G. Lower-data-rate background functions may instead be supported through low-power wide-area connectivity. The suitability of these wireless technologies for different IoT and IoM service requirements is considered in greater detail in Section 3.3.1.

2.2.2 IoM Node Hierarchy

The stakeholder interactions shown in Fig. 2-5 take place across environments that differ considerably in scale and purpose. A smart mirror in a private home, one in a dermatology clinic, and the infrastructure of a hospital or large retailer are all part of the same ecosystem, yet each serves a different set of users in a different setting. Tiered computing architectures are conventionally defined by computational capability, with the users served by each tier following from where those resources reside. The IoM is structured in the opposite direction. Its architecture is derived from the stakeholders of the ecosystem, with each tier defined by the users whose interactions it supports and the environment in which those interactions take place. Computational capability follows from this, as the resources realistically available to each of these environments differ considerably. The IoM is therefore organised as a hierarchy of three node tiers, each corresponding to a deployment context within the ecosystem and to the stakeholder interactions that context supports:

- **Consumer nodes** form the lowest tier and are intended for widespread deployment in homes, small retail locations and personal service environments. They provide the point at which consumers interact directly with the ecosystem, supporting the rendering of the smart mirror interface, local sensing, lightweight processing and communication with the remaining tiers. This tier corresponds to the consumer-to-consumer interactions in Fig. 2-5, where a service is delivered to a user within their environment.

- **Professional nodes** occupy the intermediate tier and are positioned in professional service environments such as dermatology clinics, dental practices, beauty salons, and pharmacies. They support services that exceed what a consumer node can deliver independently but that remain local to the point of care. This tier corresponds to the consumer-to-professional interactions in Fig. 2-5, whether the professional is co-located with the consumer or delivering the service remotely.

- **Hub nodes** form the highest tier and represent institutional infrastructure deployed by service providers, such as that of a hospital, a large retailer or cen-

tralised service platform. They support the most demanding services in the ecosystem, including analysis across many users and the retention of records over extended periods. This tier corresponds to the service provider interactions in Fig. 2-5, where services are offered at institutional scale.

Although each tier is associated with a particular deployment context, the interconnectivity of the IoM means that nodes are not restricted to serving only the stakeholders situated alongside them. A service initiated at a consumer node may be delivered with the involvement of a professional or hub node elsewhere in the ecosystem, which is what allows the remote interactions in Fig. 2-5 to be supported. The tiers therefore describe where users and capability are situated within the ecosystem rather than where a given service must be executed.

This node hierarchy is the structural expression of the stakeholder model that precedes it, and together the two define the ecosystem architecture of the IoM. Since the hierarchy is derived from the ecosystem's users rather than from its hardware, where a service is executed within it is not solely a matter of computational efficiency, and carries consequences for service quality, data locality, and the experience of the user receiving it. The tiers introduced here therefore provide the architectural basis through which the ecosystem is subsequently deployed, evaluated, and reasoned about throughout this thesis.

2.3 IoM Use Cases and Application Domains

Sections 2.1 and 2.2 defined the IoM as a system and established the architecture through which it operates. This section demonstrates the ecosystem in operation, first through a set of use case scenarios spanning its application domains, and then through an examination of the advantages the IoM offers relative to the devices and services currently available to its users.

2.3.1 Use Case Scenarios

Building on the stakeholder interactions defined in Section 2.2.2, five hypothetical use case scenarios are presented below to illustrate what those interactions could involve in practice and what their implications would be. Together they span the health, beauty and assisted living domains and the range of environments in which the IoM is envisioned to operate, from the residential setting to professional service and retail environments. Following on from Fig. 2-5, a visualisation of these scenarios with the relevant stakeholders and the possible interactions between them is presented in Fig. 2-6.

The use cases presented in this section are prospective illustrations of how the proposed architecture could support interactions across the IoM ecosystem, they are not

individually implemented or experimentally validated within this thesis. The technical evaluation is instead bounded to the representative implementations and computational scenarios examined in Chapters 4–6.

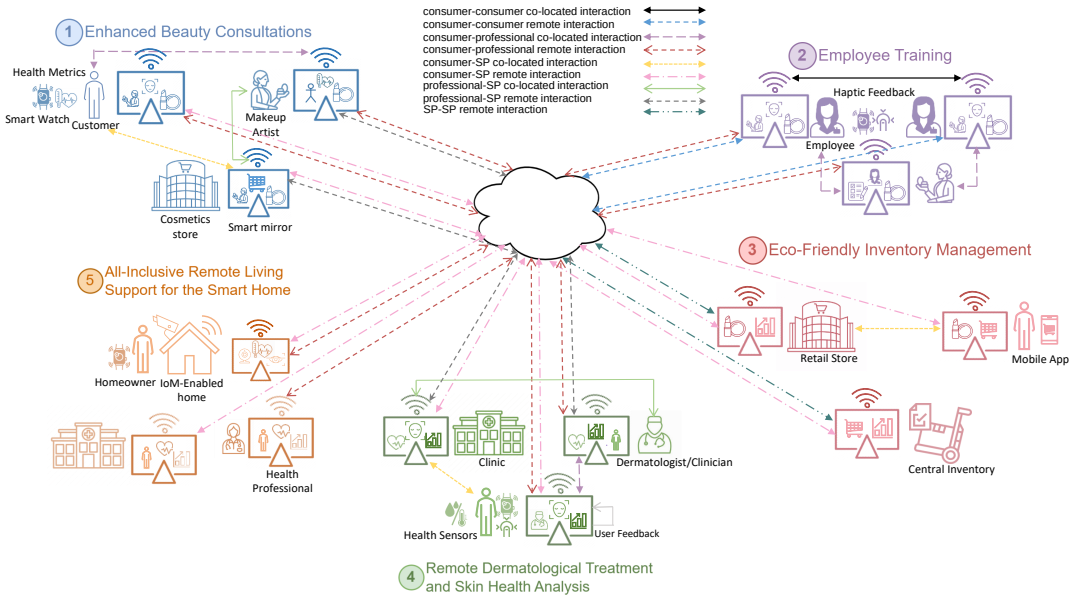


Figure 2-6: Visualisation of IoM use cases and possible stakeholder interactions.

Realising the Scenarios

The scenarios presented above depict the IoM as it is envisioned rather than as it currently exists. Each of them assumes capabilities that have yet to be established within a connected ecosystem of mirrors, including the accuracy of the visual analysis performed at the mirror interface, the reliability with which services can be delivered in real time across the network, and the coordination required when several such services operate concurrently across multiple users and locations. These assumptions cannot be resolved within any individual scenario, as they concern the ecosystem that would have to support all of them. The remainder of this thesis therefore addresses the foundational steps through which they can begin to be established, demonstrating that an IoM service can be delivered end to end, quantifying the conditions under which it can be delivered responsively, and developing the mechanisms through which concurrent and cross-ecosystem services can be coordinated. The scenarios described here remain the longer-term vision toward which that work builds, and the further capabilities they would ultimately demand are returned to in Chapter 7.

2.3.2 Unique Advantages

While the scenarios presented above share similarities with services offered by existing technologies, the IoM offers advantages that set it apart. The IoM is not designed to

replace smartphones, tablets or other smart devices, but rather to complement them by providing a specialised visual interface that integrates into daily life. In health and beauty contexts this interface supports comprehensive monitoring, real-time full-body analysis and personalised recommendations through an always-on display, extending the range of services that can be delivered beyond what handheld devices can provide.

The key differentiators of the IoM include its passive integration into daily routines, requiring minimal active engagement from users, and its role as a unified platform for health and beauty services that maintains data consistency and interoperability across them. Unlike fragmented applications and services, the IoM provides a centralised ecosystem with a large, always-on visual interface offering full-body visualisation and interaction, which is particularly suited to health and beauty applications. This design allows for hands-free operation, making it applicable to scenarios where manual interaction with a device is impractical.

Two properties of the interface follow from this design and are central to the advantages the IoM offers. The first is continuous and non-invasive monitoring. By leveraging computer vision and analysis capabilities at the mirror, the IoM complements existing wearable sensors by offering monitoring that requires nothing to be worn or charged. This can augment data collected from wearables, supporting user compliance and providing a more comprehensive dataset for analysis, and the combination of wearable data and mirror-derived information offers a richer picture of user health and appearance than either provides alone. The second is the integration of professional expertise into daily routines. The IoM can serve as a platform for remote consultations with healthcare providers or beauty professionals, and the consistency of the interface across deployments means that the visual data available to a professional does not vary with the device a given user happens to own. Its fixed position further allows for consistent, long-term monitoring of user health and appearance, enabling more accurate tracking of changes over time than is possible with handheld or intermittently worn devices.

The significance of the mirror within the IoM is therefore primarily functional rather than dependent on a particular physical implementation. The mirror provides a familiar interaction point that is already integrated into everyday activities involving appearance, self-care, health observation, and personal routines. Its typically fixed position and large visual surface support repeatable user interaction, hands-free access, and persistent availability, while sensing and processing capabilities may either be embedded locally or provided through connected external resources. These characteristics allow the mirror to act as a stable user-facing boundary to a wider distributed system.

Table 2-1 provides a high-level comparison between the IoM and representative existing technology approaches, distinguishing between capabilities associated with the smart mirror interface and those arising from its operation within an interconnected ecosystem.

Table 2-1: High-level comparison of the IoM with existing technology approaches at interface and ecosystem levels.

Feature	IoM	**Standalone smart mirrors** [29; 36]	**Wearable and mobile systems** [44; 21]	**General connected/IoT platforms** [41; 42]
Interface-level capabilities				
Full-body, hands-free analysis	Persistent visual interface supporting full-body analysis and interaction.	Supported by some systems, but confined to the individual mirror or application.	Primarily dependent on handheld or body-worn devices.	No inherent mirror-based interface for full-body visual interaction.
Passive, continuous monitoring	Repeated non-invasive monitoring with minimal active user intervention.	Possible where sensing is integrated, but generally limited to the individual mirror.	Strong continuous sensing, but typically requires a device to be worn, carried, or maintained.	Supports distributed sensing, but not through a consistent user-facing interface.
Professional-grade imaging	Fixed positioning can support repeatable imaging under more consistent acquisition conditions.	Specialised imaging can be integrated, but remains local to the mirror or application.	Image quality varies with device hardware, positioning, and environmental conditions.	Can process visual data, but does not inherently standardise image acquisition.
Ecosystem-level capabilities				
Multi-user, shared experience	Supports users and services across interconnected mirrors and locations.	Primarily centred on an individual standalone mirror.	Primarily centred on individual users and personal devices.	Supports multiple users and devices, but without a common mirror-based interface.
Contextual recommendations	Combines data across mirrors, connected devices, services, and user contexts.	Generally based on information available to the individual mirror.	Personalisation may rely on data distributed across separate applications and devices.	Can integrate heterogeneous contextual data across connected systems.
Seamless professional integration	Supports shared services and information across consumer, professional, and service-provider environments.	Remote services may be supported, but continuity across independently deployed mirrors is limited.	Supports telehealth and remote communication through personal devices.	Supports remote services and shared infrastructure, but without a consistent mirror-based interface.

Representative Example: Concurrent Support for Users with Accessibility Needs

The advantages set out above are most apparent where a user depends on several services at once, since it is the coordination between services rather than any individual service that distinguishes the IoM from the alternatives available to that user. Users with accessibility needs are a case in point, as the difficulty they face is frequently not the absence of a suitable technology but the burden of operating several of them together.

Consider a user who requires various forms of assistance throughout the day, including health monitoring, medication reminders, environmental control and remote communication. This assistance is currently delivered through a fragmented landscape of wearable devices, specialised mobile applications and smart home technologies, each operating independently with its own interface and interaction requirements [16]. As identified in Chapter 1, this fragmentation imposes a burden that grows with the number of services a user depends on, through frequent context switching, information that must be reconciled manually across systems, and a learning demand that is repeated for every device [15]. The IoM would instead coordinate these services within a single ecosystem, delivering them through a consistent visual interface that adapts to the user rather than requiring the user to adapt to each service in turn.

The extent of this burden can be expressed as user interaction overhead, meaning the number of manual actions and interventions required to access and manage services over the course of a day. This includes switching between devices, navigating unfamiliar interfaces and repeating the same information across systems. For users with accessibility needs, reducing these interactions is essential rather than merely convenient. The IoM addresses this through four properties of the ecosystem:

- **Unified access:** a single interface for all services, reducing context switching.
- **Intelligent coordination:** automatic integration between services without manual intervention.
- **Adaptive support:** context-aware assistance that anticipates user needs.
- **Seamless experience:** consistent interaction methods across all services.

Fig. 2-7 compares how user interaction overhead varies with increasing service complexity across these three approaches. As the number of concurrent services grows, the integrated approach of the IoM requires only modest increases in user interaction, owing to its unified interface and coordination between services. Both conventional assistive technologies and standalone smart mirrors instead show steeper increases in interaction requirements, as each additional service is accompanied by an additional interface to learn and operate. The figure is an illustrative conceptual model rather

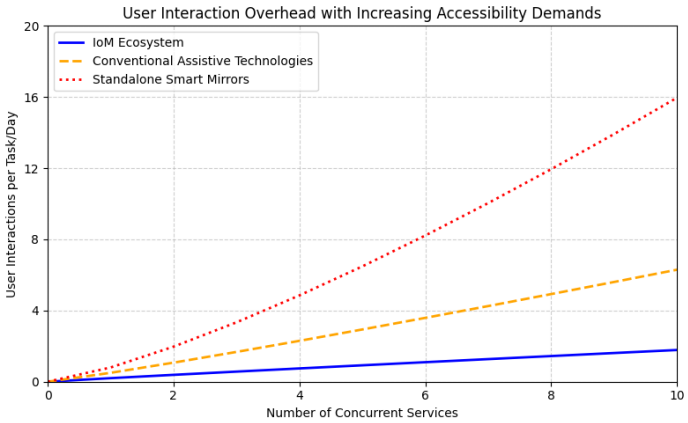


Figure 2-7: Comparison of user interaction overhead between the IoM, standalone smart mirrors and current assistive technologies.

than a representation of empirical user-study measurements. Its relative trends are informed by literature on the interaction and cognitive burden associated with fragmented interfaces and task switching [12; 14; 15], together with the assumption that each additional service introduces fewer new user interactions within a common coordinated interface than across separate devices or interfaces, the plotted values represent comparative trends rather than measured magnitudes.

The interface itself is what allows this coordination to accommodate different users. Because the IoM would support voice, gesture, visual and haptic interaction through the same surface, the mode of interaction can be matched to the user without altering the services delivered through it. Table 2-2 maps a range of accessibility needs to the IoM features that would support them, illustrating how a single platform can serve requirements that are currently met, where they are met at all, by separate and unconnected devices.

Table 2-2: IoM features mapped to accessibility requirements.

Accessibility Need	**IoM Feature**
Visual impairment	Voice interaction, audio feedback, high-contrast displays
Mobility limitations	Gesture recognition, automated environment control
Cognitive support	Adaptive interfaces, simplified interactions, memory aids
Communication support	Multi-modal interaction, real-time translation, visual aids
Sensory processing	Adjustable display settings, ambient awareness, customisable feedback

What distinguishes the IoM in this context is not the addition of specialised accessibility features, as many of those listed exist in some form already, but that they are available within one ecosystem and can operate together. Accessibility is therefore a consequence of how the ecosystem is structured rather than a set of provisions added to it, which follows directly from the user-centric basis of the architecture described in Section 2.2.2. The same holds for the advantages in Table 2-1 more generally. Each is a consequence of an IoM node being a fixed, shared, always-on interface within a connected ecosystem,

features which are not centrally available to any existing standalone smart mirror.

2.4 Summary

In this chapter, the IoM is established as a defined ecosystem, with its constituent elements, layer structure, stakeholders and node hierarchy specified, and its intended services demonstrated through a set of use case scenarios. The main outcome is a reference frame that shapes the remainder of this thesis. The architecture is derived from the users and deployment contexts of the ecosystem rather than being defined solely by computational capability. Consequently, the Consumer, Professional, and Hub tiers represent different interaction and service environments as well as different levels of available resources. Computational placement within this hierarchy therefore affects not only processing efficiency, but also the path through which a service is delivered, the network and resources involved, and the context in which the user accesses that service. Alongside this, it is established that the advantages of the IoM follow from the interconnection of mirrors rather than from the capability of any individual mirror, meaning the ecosystem must be evaluated as a whole rather than through its components. Realising this ecosystem at scale places demands on the underlying infrastructure, requiring low-latency and reliable communication, the coordination of concurrent services with competing requirements, and interoperation with neighbouring IoT ecosystems. Moving forward, the thesis turns to the technologies through which these demands could be met, reviewing the literature on smart mirrors and their enabling technologies, task offloading and scheduling, and RL for orchestration in light of the ecosystem defined here.

Chapter 3

Background and Literature Review

Building on the IoM vision and architecture established in Chapter 2, this chapter reviews the state of the art in the technologies and research relevant to realising the proposed ecosystem. The chapter critically evaluates existing work to determine how far current approaches address the requirements of the IoM and where important limitations remain. These gaps provide the research basis for the system-level challenges investigated in the subsequent chapters, while the reviewed technologies establish the technical foundations on which the IoM can operate and position the contributions of this thesis within the wider IoT research landscape.

Section 3.1 first examines the evolution of smart mirror platforms, considering the application domains and capabilities that have shaped their development and the limitations that remain within existing standalone systems. Section 3.2 then broadens the review to the sensing, computer vision, and personalisation technologies to investigate the applicability and limitations of existing paradigms to assistive health and beauty applications envisioned within the IoM.

The review subsequently moves from application-level capabilities to the wider system technologies required to support them. Section 3.3 examines the connectivity and computing architectures relevant to IoT systems and considers their capabilities and limitations across different service requirements. Building on this infrastructure, Section 3.4 reviews approaches for distributing and coordinating computational tasks, progressing from conventional and context-aware scheduling to RL-based mechanisms for adaptive operation. Finally, Section 3.5 synthesises the findings of the literature review and identifies the research gaps that motivate the subsequent technical investigation of the IoM.

3.1 Evolution of Smart Mirror Platforms

Smart mirrors were introduced in Section 1.1.3 as an emerging visual interface relevant to the motivation for the IoM. This section examines their development in greater depth to establish the capabilities represented within the current smart mirror landscape. A

smart mirror generally refers to a conventional mirror augmented with digital display, sensing, computing, or interaction capabilities [29]. Early work in the late 2000s and early 2010s explored augmented mirror interfaces based on live camera feeds, through which digital content could be incorporated into the user's reflected image [45]. Similar work combined live video with infrared-based tracking to support virtual makeup visualisation [46]. These early systems established the use of the mirror as an interactive visual interface rather than solely as a reflective surface.

The development of physical smart mirror prototypes became more prominent during the following decade. One of the earliest widely referenced implementations was Michael Teeuw's "Mirror, Mirror on the Wall" project introduced in 2014, which began as a personal proof-of-concept rather than a formally sponsored research project. The objective was to demonstrate that a conventional mirror could be augmented with a digital information display using a semi-transparent mirror, monitor, and Raspberry Pi. The project subsequently attracted a large maker community and evolved into the open-source MagicMirror^2 platform, providing a modular basis for later smart mirror implementations [47]. This general hardware arrangement subsequently became common within academic prototypes, with Raspberry Pi platforms frequently used as the embedded computing unit [38] and Arduino-based implementations also considered [48]. Many systems focused on comparatively lightweight information services, including weather, time, calendars, news, and multimedia information delivered through web services or external devices [49; 50; 51; 52]. This implementation pattern remains evident in more recent work. In [53], a Raspberry Pi-based mirror provides weather, news, time, and note-taking services for public and educational environments, while another recent implementation similarly uses a network-connected Raspberry Pi to display information including time, weather, temperature, humidity, and news [54].

Despite variation in the services implemented, much of this literature adopts a broadly similar underlying platform structure. A digital display and embedded computing device are positioned behind a reflective surface and combined with sensing, networking, or interaction components according to the requirements of the application. A typical standalone IoT-enabled smart mirror configuration is illustrated in Fig. 3-1. This provides the basic platform upon which the wider capabilities considered throughout the smart mirror literature have progressively been developed.

The capabilities incorporated into this platform have progressively expanded beyond basic information display. Infrared sensing has been used for user presence detection [34], while gesture recognition has enabled touchless interaction and control [55]. Wi-Fi connectivity has supported access to networked information and services [56], with mobile synchronisation enabling information exchange between the mirror and a user's smartphone [57]. Software extensibility has also been considered through plugin-based smart mirror platforms [39] and the integration of web application programming interfaces [58]. More recent work extends these capabilities towards increasingly sophisticated processing and interaction. Speech recognition has been executed directly

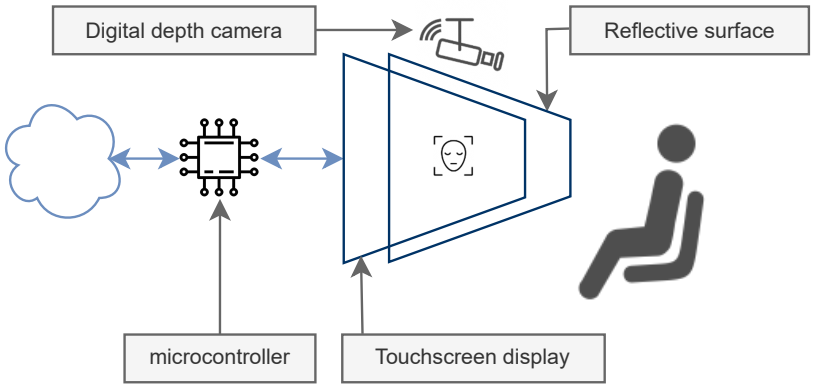


Figure 3-1: A typical standalone IoT-enabled smart mirror schematic.

on a smart mirror edge device using a Convolutional Neural Network (CNN) and long short-term memory model [59], reducing reliance on remote speech-processing services. Augmented reality interaction has also been explored through interface elements that can be manipulated using the user's reflected image [60]. A further recent system combines voice and gesture interaction, local edge artificial intelligence, non-invasive health monitoring, and privacy-preserving mechanisms within the same smart mirror platform [61]. The increase in functionality also introduces corresponding resource requirements, with the activation of different smart mirror modules shown to affect the energy consumption of Raspberry Pi-based implementations [62]. Smart mirror development has therefore progressed from relatively simple information-display systems towards platforms incorporating sensing, intelligent processing, multimodal interaction, and increasingly complex application functions.

This expansion in capability is also reflected in the commercial smart mirror landscape. Products have been developed across beauty, personal care, retail, assisted living, and smart-home applications, demonstrating wider interest in the mirror as an interactive consumer interface [35]. Table 3-1 presents representative commercial smart mirrors identified within this landscape. The range of functions includes skin analysis, product recommendation, virtual try-on, voice interaction, lighting control, and integration with other consumer technologies, illustrating the breadth of services that have been incorporated within individual mirror platforms.

Academic research demonstrates a similarly broad expansion in the application domains supported by smart mirrors. Assisted living and smart-home applications include programmable interfaces for ambient IoT environments [37] and systems incorporating intrusion detection and human monitoring [77]. Conversational interaction has been incorporated into smart mirrors to support elderly users [78], while movement detection has been used to improve pose estimation for older populations [79]. Related applications in fitness and wellbeing include automatic exercise recognition and repetition counting [80], as well as workplace health promotion through personalised physical activity information and wearable integration [81]. More recent work extends this use

Table 3-1: Commercially available smart mirrors and their features.

Name	**Features**
Sephora Mirror [63]	QR code generation for in-store and online makeup, skincare and fragrance recommendations
HiMirror [64]	Adjustable lighting, skin analysis, product recommendation, cosmetics tutorial
MemoMi [65]	Image capturing and storage, AR eyewear and clothes, smartphone and social network sharing
Charlotte Tilbury MagicMirror [66]	Augmented makeup, importing images
Shiseido Digital Counseling Mirror [67]	Product information, application instructions, skin check, smartphone compatibility
Panasonic New Magic Mirror [68]	Skin condition identification, virtual try-on
Modiface [69]	Product recommendation, virtual try-on
Lumini Kiosk V2 [70]	Skin analysis, product recommendation, skin comparison with other users
Icon AI Z Mirror [71]	Adjustable lighting, skin analysis, product recommendation, virtual try-on
Morror [72]	Adjustable lighting, cosmetics tutorial, weather information
Simplehuman Sensor Mirror Pro [73]	Smart lighting
Kohler Verdera Voice Lighted Mirror [74]	Voice activation integrated with Alexa and Google Assistant
iHome Reflect Smart Vanity Mirror [75]	Lighting and speaker control with smartphone
Capstone Connected Home Smart [76]	Integration with other smart-home devices, built-in speaker, voice activation

of the mirror as an interactive wellbeing interface by combining emotion recognition, gesture interaction, and generative feedback to support workplace emotion regulation [82].

Healthcare represents another established application area. Smart mirrors have been developed for general health monitoring [83], while the combination of sensing, visualisation, and biomedical data has been proposed as a basis for personalised medicine [32]. The capabilities demonstrated within this domain have continued to become more specialised. Real-time computer vision has been used for acne lesion detection and tracking of skin condition in a system intended for smart mirror deployment [31]. Contactless physiological monitoring has also progressed, with remote photoplethysmography integrated into a smart mirror to estimate vital parameters from facial video [33]. These examples show how the mirror platform has progressed from displaying health information towards supporting the acquisition and analysis of user-specific physiological and visual data.

Personalisation and visual analysis are similarly prominent across beauty and consumer

facing applications. Facial and affective recognition have been used to identify user state for health-related applications [84], deliver personalised information according to detected emotion [85], and adapt environmental lighting according to facial expression [86]. Facial and voice authentication have also been incorporated to support personalised and secure interaction [87], while facial recognition has been combined with personalised information recommendation [88]. Within beauty applications, intelligent makeup recommendation and synthesis have been demonstrated through smart mirror interfaces [89], while skin-type identification has been used to provide individualised skin analysis [90]. More recent systems extend these functions by combining facial analysis, skin-health assessment, virtual enhancement, fitness visualisation, and smart-home interaction within a single mirror platform [91]. User-centred smart mirror research for retail has also identified intelligent analysis, real-time guidance, and adaptive lighting as important design requirements [92]. These developments continue the established movement towards increasingly personalised and interactive mirror-based services rather than introducing a separate application direction.

Overall, the smart mirror literature demonstrates a clear progression from basic augmented displays and information services towards increasingly interactive and computationally capable platforms operating across health, assisted living, fitness, beauty, retail, and wider personal wellbeing applications. As established in Section 1.1.3, these advances remain centred on increasingly capable individual smart mirror systems rather than coordinated operation between smart mirrors as part of a wider IoT ecosystem. The significance of the state of the art considered here is therefore the increasing breadth and sophistication of the services that smart mirror platforms can support. These services depend on a wider range of sensing, computer vision, and personalisation technologies and introduce differing data modalities, processing requirements, and interaction demands. Section 3.2 therefore broadens the review beyond smart mirror platforms themselves to examine the enabling technologies underlying these diverse assistive applications.

3.2 Sensing, Computer Vision, and Personalisation Technologies

The application landscape examined in Section 3.1 demonstrates that increasingly sophisticated smart mirror services depend on capabilities extending beyond the mirror interface itself. Applications across health monitoring, assisted living, personal care, visual augmentation, and personalised interaction differ in the information they require, the mechanisms through which that information is acquired, and the processing needed to transform it into a user-facing service. This section critically reviews the sensing, computer vision, and personalisation technologies that underpin these capabilities to establish the extent to which they are supported by the current state of the art and

the requirements that emerge when they are considered collectively within an interconnected IoM environment. The review therefore considers not only the capabilities demonstrated by individual technologies, but also differences in sensing hardware, data modalities, analytical operations, and contextual dependencies that become significant when diverse services are supported through a common ecosystem.

3.2.1 Multimodal Sensing and Data Heterogeneity

The sensing technologies relevant to these services encompass conventional and specialised imaging [93], non-contact physiological sensing [94], radio-frequency sensing [95], and wearable technologies [44]. Visual imaging is particularly prominent in personalised health and appearance-related applications, where classification, feature extraction, and segmentation have been used to identify characteristics including skin texture, colour, lesions, spots, and wrinkles [96; 97]. Machine learning-based image analysis has also been applied to melanoma and wider skin-condition detection [98; 99], while more recent work has incorporated real-time acne lesion detection and skin-condition tracking into a system intended for smart mirror deployment [31]. These applications demonstrate the broad utility of conventional visual imagery, but the information available from RGB cameras is not sufficient for all services. Multispectral imaging has instead been investigated for assessing skin pigmentation and lesions [100; 101], while capacitive imaging has been used for skin hydration analysis [102] and specialised devices such as wearable skin sensors and dermatoscopes provide further forms of dermatological information [103]. The state of the art therefore does not indicate a single sensing configuration capable of supporting the breadth of envisioned services. The appropriate sensing mechanism remains application-dependent, introducing differences in the hardware required at the point of interaction and in the form of data subsequently available for analysis.

Non-contact physiological sensing further broadens the information that can be acquired during normal interaction with a user-facing interface. Photoplethysmography uses variations in reflected light to derive physiological information including heart rate and respiratory activity [104], while related measurements can be obtained remotely by analysing colour variations within facial video [94]. Subsequent work has sought to improve the extraction of physiological signals from noise using machine learning [105], and more recent smart mirror research has demonstrated real-time estimation of vital parameters within a mirror-based healthcare system [33]. Recent reviews of remote photoplethysmography further highlight its potential for camera-based vital-sign estimation while identifying persistent challenges associated with motion artefacts, illumination variation, image quality, and generalisation across users and environments [106; 107]. These characteristics make camera-based sensing particularly compatible with smart mirror platforms, since the same visual interface can support both image-based analysis and physiological monitoring, but also introduce sensitivity to acquisi-

tion conditions and additional processing requirements.

Infrared thermography provides an alternative non-contact modality by measuring emitted infrared radiation rather than visible-light information. Thermal patterns have been investigated for dermatological assessment and skin cancer detection [108; 109], while infrared thermography has also been applied to automated respiratory monitoring [110; 111]. This provides access to thermal and physiological information that is not directly available through conventional RGB imagery and can therefore complement camera-based analysis. However, thermographic measurements remain sensitive to acquisition methodology, environmental conditions, device calibration, and measurement protocol [112; 113]. Infrared sensing therefore requires dedicated imaging hardware and more controlled acquisition procedures than conventional visual sensing.

Radio-frequency sensing provides a further mechanism for unobtrusive monitoring in healthcare and assisted living [95]. Radar-based systems can detect small movements associated with respiration and cardiac activity [114], while related RF techniques have been investigated for cardiovascular and cardio-pulmonary monitoring [115]. More recent reviews demonstrate continued development of millimetre-wave, ultra-wideband, and other microwave sensing techniques for contactless vital-sign monitoring, including under dynamic conditions [116; 117]. Unlike optical and thermal sensing, RF-based methods derive measurements from propagated radio signals and are therefore not dependent on direct visual image acquisition. This can provide advantages in conditions where visual sensing is unsuitable, but introduces specialised RF hardware, signal-processing requirements, and sensitivity to factors such as body movement and environmental interference.

These approaches therefore share the objective of obtaining physiological information without conventional contact-based measurement, but differ considerably in sensing mechanism, hardware, data representation, and processing requirements. Recent multimodal sensing reviews similarly emphasise that camera, radio, and other non-contact modalities provide complementary rather than interchangeable information, with fusion approaches increasingly considered to improve robustness across monitoring tasks [118; 119]. Camera-based methods integrate naturally with the visual smart mirror interface and can exploit sensing hardware already common to the platform, but their performance remains dependent on factors such as illumination, viewpoint, motion, and image quality. Infrared and RF sensing extend the range of physiological information that can be acquired beyond conventional visual imaging, but introduce specialised hardware, distinct signal representations, calibration requirements, and modality-specific processing pipelines. For the IoM, this means that multimodal sensing cannot be treated as a uniform input source; the ecosystem must accommodate differences in acquisition conditions, data formats, computational workloads, and communication requirements while allowing heterogeneous sensing information to be combined within a common service framework.

3.2.2 Computer Vision and Personalisation

Once user information has been acquired, computer vision provides a principal mechanism through which visual sensing data can be interpreted for personalised and interactive services. The state of the art encompasses classification and feature extraction [120], object detection and segmentation [31], and visual augmentation and synthesis [121]. These methods have been applied to skin analysis, including acne classification according to lesion type [122] and severity estimation from user-acquired imagery [123]. Similar techniques have also been applied within elective healthcare, including automated dental smile analysis using image classification and segmentation [20]. Although these applications share visual data as an input, the analytical operations required to produce their outputs differ. Classification, detection, segmentation, and feature extraction represent distinct processing pipelines and may operate on different forms and volumes of visual information. Their common use of computer vision therefore does not imply equivalent execution requirements when several such applications are supported within the same computational environment.

Computer vision additionally supports interactive visualisation through augmented reality. Early augmented mirror systems combined live imagery and tracking to superimpose virtual makeup onto the user [46], while subsequent systems introduced more interactive forms of virtual cosmetic application [124] and improved colour reproduction using convolutional neural networks [121]. Personalisation has been extended by incorporating characteristics including skin tone, attire, hair, and facial features when generating makeup recommendations [125], while geometric properties of the face have also been used to customise makeup styles [126]. Similar combinations of visual analysis and augmentation have been investigated beyond cosmetics, including body measurement for apparel selection [127], augmented fitness guidance [128], and three-dimensional product tracking for virtual visualisation [129]. These studies demonstrate a progression from static visual analysis towards services in which acquisition, analysis, and visual feedback occur as part of an interactive process. When considered alongside simpler classification or recommendation applications, this introduces further variation in the responsiveness and processing characteristics associated with different user-facing services.

Recommendation systems provide a further stage of personalisation by relating detected user characteristics to an individualised output. Deep learning has been used to identify makeup from facial imagery [130], while eye-shape classification has been combined with real-time eyeliner guidance [131] and facial imagery has been linked with product information to generate personalised cosmetic recommendations [132]. More advanced approaches expand the basis of the recommendation beyond visual features alone. Professional knowledge has been combined with machine learning-based facial classification for hair and makeup recommendations [133], while occasion has been considered alongside user appearance to generate context-specific recommendations and

visual previews [134]. Three-dimensional facial representations provide further information about individual facial geometry [135], and user feedback can be incorporated to refine subsequent recommendations according to previous preferences and responses [136]. Social context has similarly been considered within recommendation systems [137]. The literature therefore demonstrates a progression from personalisation based principally on individually detected characteristics towards approaches combining visual information with contextual, professional, historical, and social data.

This progression increases the range of information on which a personalised service may depend. While some applications can derive an output directly from locally acquired imagery, others require information from multiple sources or previous interactions before a recommendation can be generated. Personalisation therefore introduces a system-level consideration in addition to an application-level capability, as the information required to execute a service may be distributed across different devices, data sources, or computing resources. The resulting applications also differ in whether processing is performed once on an individual image, continuously on a visual stream, or interactively as the user receives and responds to real-time feedback. These differences become increasingly important when the applications are no longer considered as isolated implementations but as services operating concurrently within a shared connected environment.

Overall, the reviewed literature establishes that many of the sensing, analytical, visualisation, and personalisation capabilities required by the IoM are individually well developed. The research issue relevant to this thesis is therefore not the absence of these techniques, but the heterogeneity that emerges when their capabilities are considered collectively within an interconnected ecosystem. The state of the art spans embedded and external sensing, visual and non-visual modalities, different analytical pipelines, interactive and non-interactive processing, and personalised services dependent on varying amounts of contextual information. These differences introduce corresponding variation in data, processing, communication, and resource requirements, particularly as multiple services and users are supported within a shared environment. The question consequently shifts from whether the individual enabling technologies exist to how their heterogeneous requirements can be supported through a common connected and distributed infrastructure.

3.3 Connectivity and Computing Architectures for IoT Systems

The technologies reviewed in Section 3.2 demonstrate that information supporting personalised services may originate from the smart mirror itself, external sensors, wearable devices, or other connected systems. Enabling these components to operate collectively requires communication between distributed data sources and access to comput-

ing resources through which the resulting services can be supported. The wider IoT landscape provides several wireless communication technologies and distributed computing paradigms for this purpose, each offering different characteristics according to the application and deployment environment [138] [139]. This section reviews these two aspects of the supporting infrastructure, beginning with wireless connectivity and subsequently considering the development of cloud, edge, and fog computing towards the wider computing continuum.

3.3.1 Wireless Connectivity for IoT

IoT environments encompass devices operating across different physical ranges, data rates, power constraints, and service requirements, resulting in the use of several complementary wireless communication technologies [138]. Local environments such as smart homes commonly employ wireless local area network technologies including Wi-Fi, while wider-area healthcare and mobile IoT services may rely on cellular or other long-range communication mechanisms [139; 140]. Low-power wide-area networks provide an alternative for applications prioritising extended coverage and low energy consumption [141], while wireless sensor and local area networks support communication across more localised connected environments [142]. The appropriate communication mechanism is therefore dependent on the characteristics and location of the devices being connected rather than on a single network technology applicable across all IoT services.

The variation between IoT applications is also evident in their data communication requirements. A survey of IoT connectivity technologies in [143] distinguishes applications according to factors including data rate and whether interaction is predominantly machine-oriented or human-oriented. Applications involving rich multimedia, continuous sensing, or interactive visual information occupy a different communication profile from low-rate monitoring and telemetry services. This distinction is relevant to the service landscape examined in Section 3.1 and Section 3.2, where information may range from individual sensor measurements to images and continuous video streams.

Cellular networks have consequently received substantial attention for IoT applications requiring higher data rates or responsive interaction. 5G extends the available wireless capacity and support communication characteristics associated with low-latency and high-data-rate services [141]. Millimetre-wave communication provides access to wider channel bandwidths within 5G systems, supporting high data-rate transmission while introducing its own propagation and deployment considerations [144]. The relevance of 5G to interactive IoT services has also been demonstrated in IoT sub-ecosystems. In networked musical interaction, for example, latency and reliability have been experimentally evaluated over a 5G-enabled Internet of Musical Things environment [145], illustrating the importance of communication performance where distributed users interact in real time.

More recent empirical work demonstrates that communication performance also varies according to the network environment and operating conditions. A comparison of private 5G and Wi-Fi under concurrent industrial traffic found that 5G maintained lower latency and fewer latency outliers under the evaluated load conditions [146]. Experimental evaluation of multimedia communication on a live 5G edge platform similarly demonstrates that the performance of interactive augmented reality services is influenced by both network and communication protocol characteristics [147]. These findings reinforce that network capability alone does not fully characterise the communication experienced by an application.

The availability of multiple radio access technologies introduces further variability. Multi-radio edge-computing research has considered the simultaneous use of Wi-Fi, Wi-Fi Direct, and 5G, where differences in throughput and delay across the available links affect how traffic can be distributed between them [148]. The results demonstrate that inappropriate allocation across heterogeneous radio technologies can reduce throughput and increase service delay, while informed utilisation of their differing capabilities can improve communication performance [148]. Wireless connectivity in distributed IoT systems should therefore be viewed as a heterogeneous and condition-dependent resource rather than as a fixed transport layer.

The state of the art consequently provides a broad range of wireless technologies capable of connecting devices across local and wider environments. However, the communication characteristics available to a service depend on factors including network type, deployment range, traffic conditions, and the data being transferred. Connectivity therefore provides the mechanism through which distributed IoT components can exchange information, but does not by itself determine how or where that information should be processed. This consideration has become increasingly important as IoT architectures have moved away from reliance on centralised cloud computing towards the use of computing resources distributed throughout the network.

3.3.2 The Computing Continuum

Cloud computing has traditionally provided IoT systems with access to centralised computing and storage resources beyond those available on individual end devices [149]. This approach allows data collected from distributed devices to be transferred to remote infrastructure for storage, analysis, and service delivery. However, dependence on remote computing also requires information to traverse the network between the data source and the cloud, making communication an inherent component of service delivery. The increasing scale and responsiveness requirements of IoT applications have therefore contributed to the development of computing paradigms that position resources closer to where data is generated [150].

Edge computing places computation and storage resources at or near the network edge, where the data is actually generated, reducing the physical and network separation be-

tween end devices and processing resources [151]. In [150], proximity to mobile devices and sensors is identified as a defining characteristic through which edge computing can support responsive services, IoT scalability, and local policy enforcement. This makes edge resources particularly relevant where data is generated continuously or where interaction with a remote cloud would otherwise introduce additional communication dependency.

Fog computing extends this distributed model by introducing intermediate computing resources between end devices and central cloud infrastructure [152]. Rather than limiting processing to either a local device or a remote data centre, fog architectures provide additional locations at which services and data processing can be hosted. This model has been examined extensively in healthcare, where a systematic review in [153] identifies fog computing as a means of addressing challenges associated with large data volumes, response time, latency, and distributed healthcare services. Related work has similarly considered combined fog and cloud environments for context-aware health monitoring across interconnected smart environments [154].

The distinction between cloud, edge, and fog computing has increasingly shifted towards their integration within a common distributed environment. The integration of edge resources, IoT devices, and cloud infrastructure is considered in [155], where these components are framed as complementary parts of a distributed computing architecture rather than isolated alternatives. Distributed computing allows computational functions to be performed across multiple resources [156], creating an infrastructure in which the location of processing can vary according to the resources and services involved.

More recent literature formalises this progression as an edge-fog-cloud or edge-cloud computing continuum. A recent review in [157] traces the evolution from centralised IoT architectures through fog, edge, and local-edge approaches towards a multi-tier computing continuum in which resources closer to users operate alongside centralised data-centre infrastructure. Similarly, a heterogeneous three-tier edge–fog–cloud architecture evaluated on a physical testbed demonstrates the potential benefits of collaboration between resources at different levels [158]. Importantly, the same study finds that the benefit of relocating computation is dependent on the configuration and relative capabilities of the available devices rather than being universally advantageous.

This dependence is consistent with earlier research on workload allocation across IoT, edge, and cloud systems, where processing across multiple levels has been considered to balance delay and resource-related objectives [159]. The literature therefore does not support a simple division in which one computing paradigm is universally preferable to the others. Instead, edge, fog, and cloud resources provide complementary capabilities whose suitability varies according to the service, available infrastructure, and communication environment. The computing environment is therefore better represented as a continuum of complementary resources rather than as a choice between isolated

architectural models.

The characteristics of these computing paradigms are summarised in Table 3-2. Rather than representing mutually exclusive architectural choices, the comparison highlights how cloud, fog, and edge resources occupy different positions within a distributed infrastructure and offer complementary capabilities and constraints. Their relative suitability consequently depends on the computational and communication requirements of the service being supported.

Table 3-2: Characteristics of computing paradigms within the IoT computing continuum.

Paradigm	**Typical Position**	**Key Characteristics**	**Relevant Considerations**
Edge computing [151; 150]	Close to end devices and data sources	Localised computation and storage with reduced dependence on remote infrastructure	Resource capability is typically more constrained than remote infrastructure and varies between edge devices
Fog computing [152; 153]	Intermediate between end devices and cloud infrastructure	Distributed processing across resources positioned within the network	Introduces additional processing locations and supports cooperation between local and remote resources
Cloud computing [149]	Remote centralised infrastructure	Large-scale computing and storage resources accessible to distributed devices	Service delivery depends on communication with remote infrastructure
Edge-fog-cloud continuum [157; 158]	Resources distributed across multiple infrastructure levels	Combines resources at different locations and capability levels within a common architecture	The suitability of each processing location depends on the service, resource configuration, and communication environment

The computing continuum provides an important architectural capability for the IoM vision established in Chapter 2. The interconnected ecosystem does not require every service to be restricted to the processing resources of an individual IoM node, nor does it require all information to be transferred to a central cloud. Existing distributed computing research instead demonstrates that computational resources can be made available at several points throughout a connected environment [155]. At the same time, the literature shows that these resources differ in their location and capability, while access to them is mediated by communication networks whose characteristics can also vary.

The existence of multiple possible computing resources consequently introduces a further system-level question concerning how computational tasks should be distributed between them. This is distinct from establishing the available connectivity or computing architecture itself. Once several execution locations are available, mechanisms are required to determine how workloads are assigned across distributed and potentially

heterogeneous resources.

3.4 Task Distribution and Adaptive Orchestration in Distributed IoT Systems

The distributed computing landscape examined in Section 3.3.2 provides several possible locations at which computational tasks may be executed. A substantial body of research has therefore investigated how workloads can be distributed across these resources, ranging from conventional task offloading and scheduling mechanisms to context-aware and learning-based approaches. Table 3-3 provides an overview of the principal approaches considered in this section and the types of information incorporated into their decision processes. The following subsections examine this progression in detail, beginning with task offloading and context-aware scheduling before considering RL-based approaches for adaptive operation.

Table 3-3: Progression of task distribution and adaptive scheduling approaches.

Approach	**Year**	**Decision Factors**	**Adaptive**	**Scope of Context**
Simple allocation [160]	2024	Availability or predefined allocation order	No	Limited task and node context
Resource and load-aware scheduling [161]	2024	Resource utilisation and system load	Limited	Predominantly system state
Multi-criteria scheduling [162; 163]	2023; 2025	Multiple resource, network, cost, and performance criteria	No	Multiple predefined criteria
Context-aware scheduling [164; 165]	2023; 2022	Task, service, and operating context	Limited	Broader task and system context
Hierarchical offloading [166; 167]	2023; 2017	Resource characteristics across multiple infrastructure levels	Varies	Tier and resource context
RL-based scheduling [168; 169]	2022; 2023	Dynamic resource, queue, and network state	Yes	Predominantly system context
Application and multimodal-aware RL [170; 171; 172]	2025; 2025; 2023	Priority, data characteristics, and service requirements	Yes	Extended task and application context

3.4.1 Task Offloading and Context-Aware Scheduling

Computation offloading concerns the transfer of tasks from resource-constrained devices to other computing resources, while scheduling determines how tasks and available resources are coordinated within the distributed environment [173]. Research in this area

encompasses task assignment, resource allocation, and collaborative scheduling across edge, fog, and cloud resources [174]. Existing surveys show that much of this literature is organised around resource allocation and task assignment, with task characteristics receiving comparatively less consideration as part of the wider decision context [175].

Early and conventional approaches commonly distribute tasks according to individual allocation criteria. Round-robin and random assignment provide simple mechanisms for distributing workloads without detailed consideration of task-to-resource suitability [160], while load-aware approaches redistribute tasks according to current resource utilisation [161]. Other methods prioritise factors such as proximity or available computational resources. These approaches provide comparatively simple decision mechanisms but generally optimise a limited view of the conditions affecting task execution.

Other recent research has increasingly incorporated multiple criteria and contextual factors. Multi-criteria approaches have considered combinations of execution time, energy consumption, cost, resource availability, and processing capability when selecting resources across heterogeneous fog–cloud environments [162]. Resource-ranking methods similarly combine several node and network characteristics when scheduling tasks [163]. Beyond resource characteristics, task dependencies have been incorporated into joint scheduling and resource-allocation formulations [176; 177], while security requirements have also been considered within task distribution [178]. Context-aware scheduling extends this further by incorporating information about the surrounding operating conditions when making allocation decisions [164; 165].

Task distribution has also been investigated across hierarchical and multi-tier computing environments. Offloading mechanisms have been developed for systems in which computational resources are distributed across multiple infrastructure levels [166], including hierarchical mobile edge environments [167]. Adaptive resource-management approaches similarly exploit differences between resources located at different levels of an IoT architecture [179]. These studies demonstrate that both heterogeneous resources and hierarchical organisation can be incorporated into task allocation and scheduling decisions.

The state of the art therefore shows a progression from simple allocation strategies towards approaches incorporating multiple task, resource, network, and contextual factors. This broader decision context becomes particularly relevant in user-centred ecosystems such as the IoM, where scheduling must also accommodate variation across users, applications, and stakeholder-oriented tiers. As these factors and the underlying system conditions change during operation, decisions based on fixed criteria or predefined optimisation priorities may not remain appropriate across all situations.

3.4.2 RL for Task Offloading and Service Orchestration

The increasing number of factors considered in task allocation, together with their variation during operation, has motivated the use of RL for adaptive task offloading

and resource management. Deep RL has been applied across communication, computing, caching, and control problems in IoT, with reported improvements in objectives including latency, energy efficiency, and task completion [180; 181]. Value-based methods such as Deep Q-Networks (DQN) and Double DQN are commonly applied to discrete offloading decisions [182] [183], while other RL formulations address continuous resource allocation and cooperative scheduling [184] [185]. These approaches extend conventional scheduling by allowing allocation policies to be updated in response to observations from a changing environment [180].

RL-based offloading has also been investigated across hierarchical computing environments. End-device, edge or fog, and cloud resources are commonly organised as distinct infrastructure levels through which offloading decisions can be made [186] [187]. Three-tier RL formulations similarly distinguish resources according to factors including computational capability and network proximity [188]. Although this structure is comparable to the tiered IoM architecture established in Chapter 2, the IoM hierarchy additionally reflects the stakeholder roles and service contexts associated with its Consumer, Professional, and Hub tiers.

Contextual information has increasingly been incorporated into RL-based scheduling decisions. System-level observations including resource utilisation, queue state, and network conditions are used to represent changing operating conditions [168] [169]. Application-related requirements have also been represented through factors such as task priority and user-defined performance objectives [170] [189]. Across these approaches, urgency is principally expressed through system state, predefined task priorities, deadlines, or optimisation weights. This differs from a user-centred service environment in which the requirements associated with a task may also depend on the application and operational context in which it is being delivered.

Heterogeneous data has also begun to receive greater consideration within offloading research. Multimodal edge-computing formulations account for differences between data streams through objectives such as synchronisation [171], while reliability-aware approaches incorporate data-dependent service requirements into task processing and offloading decisions [172]. In these examples, the characteristics of the data influence the quality or constraints of the allocation decision. They do not, however, explicitly represent modality as a node-eligibility condition in which a task can only be assigned to resources possessing the corresponding processing capability.

The literature therefore demonstrates that RL provides increasingly adaptive mechanisms for task allocation across dynamic and hierarchical IoT environments. However, the reviewed formulations are predominantly developed around the requirements and infrastructure of the individual environment being managed. The IoM introduces a broader decision context through its user-centred stakeholder hierarchy, diverse application requirements, and envisioned interaction with other IoT ecosystems established in Chapter 2.

Overall, the scheduling and orchestration literature in this domain demonstrates a progression towards increasingly context-aware and adaptive task distribution across heterogeneous IoT environments. Existing approaches account for a growing range of task, resource, network, and application characteristics, although the extent to which these formulations capture the combined user, stakeholder, modality, and cross-ecosystem context required by the IoM remains limited.

3.5 Summary and Research Gap

To provide an overarching summary and comparison of the literature reviewed across Sections 3.1-3.4, Table 3-4 presents the state of the art across the principal research areas relevant to the IoM, the gaps identified within each area, and how the IoM is positioned in relation to them.

This chapter reviewed the state of the art relevant to enabling and realising the IoM vision established in Chapter 2. Existing smart mirror research demonstrates increasingly capable sensing, interaction, analysis, and personalisation, while the wider IoT landscape provides the connectivity and distributed computing technologies through which such services can be supported. The review also shows that these capabilities are heterogeneous in their sensing modalities, data characteristics, processing requirements, interaction patterns, and dependence on contextual information. The principal research challenge is therefore not the absence of the individual enabling technologies, but how these differing requirements can be supported, coordinated, and operated when brought together within a heterogeneous, user-centred IoM ecosystem.

First, although edge, fog, and cloud resources provide multiple locations for service execution, their suitability varies with the application, available resources, network conditions, and deployment environment. Existing research therefore establishes the computing continuum as a viable basis for distributed IoT services, but does not determine how the IoM's heterogeneous services perform when realised across its stakeholder-oriented tier hierarchy under practical operating conditions. This motivates the system-level investigation of distributed IoM service realisation in Chapter 4.

Second, existing task offloading and scheduling approaches have progressed from simple allocation strategies towards multi-criteria, context-aware, and hierarchical mechanisms. Nevertheless, these approaches are predominantly formulated around particular task, resource, or system objectives, with limited consideration of the combined diversity of users, applications, task requirements, node capabilities, and stakeholder-oriented tiers that characterise the IoM. This motivates the context-aware task distribution problem addressed in Chapter 5.

Finally, RL provides a basis for adapting task allocation to changing system conditions, including across hierarchical IoT infrastructures. However, existing formulations are primarily designed around the requirements of a single ecosystem and generally

represent context through infrastructure state, predefined task characteristics, and optimisation objectives. This becomes restrictive when the IoM interacts with external IoT ecosystems, where task origin, modality, compatible resources, and service urgency may depend on the context of the originating ecosystem. This motivates the adaptive cross-ecosystem orchestration problem investigated in Chapter 6.

Table 3-4: Summary and comparison of the state of the art, identified research gaps, and positioning of the Internet of Mirrors.

Research Area	**State of the Art**	**Identified Gap**	**Position of the IoM**
Smart mirror platforms	• Digital display and user interaction • Embedded sensing and networking • Increasingly personalised and intelligent services • [56; 59]	• Predominantly standalone systems • Application-specific implementations • Limited coordination between mirrors	• Interconnected smart mirror nodes • Shared data, services, and resources • Operation as part of a wider IoT ecosystem
Multimodal sensing	• Camera-based sensing • Thermal and RF sensing • Wearable and physiological monitoring • [118; 116]	• Different hardware and data formats • Different acquisition conditions • Modality-specific processing requirements	• Supports heterogeneous sensing modalities • Accounts for differing processing and communication demands • Enables multimodal services within a common ecosystem
Computer vision and personalisation	• Image analysis and classification • Visual augmentation • Personalised recommendation • Context-aware services • [89; 133]	• Commonly application-specific • Reliance on different data and context sources • Limited integration across services	• Integrates multiple personalised services • Supports distributed data and processing • Allows information sharing across devices and resources
Wireless connectivity	• Wi-Fi, cellular, and LPWAN • Support for different ranges and data rates • Increasing support for low-latency services • [143; 146]	• No single technology suits all services • Performance varies with traffic and deployment conditions • Different services impose different communication demands	• Operates across heterogeneous networks • Treats connectivity as a variable system resource • Links communication conditions to service delivery
Distributed computing	• Edge, fog, and cloud processing • Distributed storage and computation • Computing continuum across multiple resource levels • [151; 152]	• Primarily organised around infrastructure capability • Limited consideration of stakeholder and service context	• Consumer, Professional, and Hub hierarchy • Links computation to stakeholder context • Supports distributed service execution across tiers
Task distribution and context-aware scheduling	• Resource- and load-aware allocation • Multi-criteria scheduling • Context-aware and hierarchical offloading • [162; 164]	• Context generally limited to selected task or system factors • Limited joint consideration of users, applications, modalities, and stakeholders	• Combines task, application, resource, and stakeholder context • Considers modality-resource compatibility • Supports allocation across the IoM hierarchy
RL-based orchestration	• Adaptive task offloading • Dynamic resource allocation • Hierarchical edge–cloud orchestration • [180; 184; 185]	• Predominantly single-ecosystem formulations • Context mainly based on system state and optimisation objectives • Limited cross-ecosystem consideration	• Context-aware adaptive orchestration • Incorporates service urgency and modality compatibility • Extends coordination across IoM and external IoT ecosystems

Chapter 4

Proof-of-Concept and System-Level Evaluation of the IoM as a Distributed System

4.1 Introduction

The conceptual and architectural foundations of the IoM were established in Chapter 2, including the stakeholder-driven Consumer, Professional and Hub node hierarchy defined in Section 2.2.2. Chapter 3 subsequently examined the connectivity and computing infrastructure through which this architecture could be realised. In particular, Section 3.3 established that connected IoT services may access computational resources at different locations across the computing continuum, while the suitability of these resources depends on their processing capabilities, the communication environment, and the requirements of the service being supported. Building on these foundations, this chapter addresses Q_2 and O_2 by experimentally investigating the performance implications and trade-offs that arise when IoM services are realised across distributed computing infrastructure under progressively more representative operating conditions.

The investigation proceeds through the three stages defined in O_2. First, a representative automated dental smile-analysis application, developed and evaluated through the work reported in [J3] and [C1], is integrated with the smart mirror interface to establish the feasibility of delivering an assistive IoM service, addressing O_2a. The application is then used to investigate the effect of computational placement between local edge and remote cloud resources, published in [C2], addressing O_2b through the decomposition of end-to-end service latency into its computational and communication components. Building on the findings and limitations of this initial comparison, the evaluation is extended in [C3] to address O_2c with a physical implementation of the Consumer, Professional and Hub hierarchy established in Chapter 2. Multiple computational placement strategies are evaluated under Wi-Fi and 5G network conditions and increasing concurrent demand to examine how processing capability, communication

overhead, resource utilisation, and network conditions interact across the distributed IoM architecture.

The purpose of this evaluation is to characterise the system-level consequences of distributing an IoM workload rather than to prescribe a single computational placement for all services or deployment environments. The developed dental smile analysis use case provides a consistent representative workload through which the architectural and performance implications of distribution can be examined, while the progression from a single application to edge–cloud placement and subsequently to the physical multi-tier hierarchy introduces progressively greater system complexity. Accordingly, the numerical latency, resource-utilisation, and scalability results reported in this chapter are specific to the evaluated dental smile-analysis workload, hardware, and network configurations. Their broader contribution lies in demonstrating how distributed IoM performance depends on the interaction between computational demand, resource capability, communication overhead, network conditions, and system load, rather than implying that the measured numerical values generalise directly to all IoM applications.

4.2 Representative IoM Use Case: Automated Dental Smile Analysis

The application scope established in Section 1.2 is bounded to representative use cases that provide concrete workloads through which the operation of the IoM can be investigated. Accordingly, the first stage of the evaluation develops an automated dental smile-analysis application as a proof-of-concept IoM service, published in [J3] and [C1], addressing O_2a. Dental smile analysis was selected as a visually driven assistive use case that can be delivered through the sensing, processing, and display capabilities of a smart mirror, while also providing a computational pipeline comprising image acquisition, preprocessing, machine-learning inference, and user-facing result delivery. The resulting curated dental smile dataset developed for this use case represents an additional research output of this proof-of-concept, constructed specifically to address the absence of a suitable labelled frontal dental smile dataset for the intended classification task.

The purpose of the proof-of-concept within this thesis is to establish the feasibility of implementing an end-to-end assistive service through the smart mirror interface and to provide a representative application workload for the subsequent system-level evaluations. The following subsections, therefore, describe the application requirements, dataset and model development, integration with the smart mirror interface, and application-level evaluation.

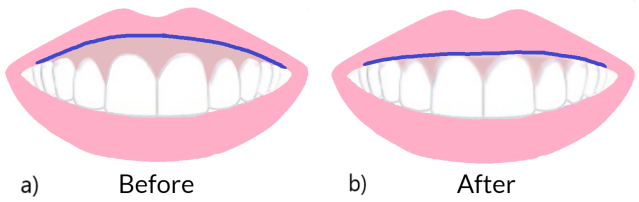


Figure 4-1: Presence of excessive gingival display in (a) before orthodontic treatment and an ideal lip line in (b) following treatment. The blue line indicates the location of the lip line.



Figure 4-2: Examples of the principal smile-arc configurations: (a) ideal, (b) flat, and (c) reversed or inverted. The blue line follows the curvature of the lower border of the maxillary teeth, while the orange line illustrates deviation from this curvature in the flat and reversed cases.

4.2.1 Clinical Motivation and Application Requirements

Aesthetic dental treatment is influenced by the role that smile appearance can play in self-confidence, social acceptance, and psychological well-being [190; 191]. Dissatisfaction with smile appearance may also influence oral self-care behaviours [192], while orthodontic treatment of aesthetic concerns has been associated with improvements across several related factors [193; 194]. Before appropriate treatment can be considered, however, the visual characteristics contributing to a smile must first be assessed. Smile analysis therefore forms an important component of dental and orthodontic assessment and provides a basis for identifying aesthetic characteristics that may require further professional consideration.

Smile analysis considers several visual relationships between the teeth, gingiva, lips, and surrounding facial structures. One such characteristic is the lip line, which describes the extent of the maxillary teeth displayed during smiling [195]. An ideal lip line exposes approximately three-quarters to the full clinical crown of the maxillary teeth together with the interdental gingiva. A low lip line exposes less than three-quarters of the clinical crown height, while a high lip line exposes the full clinical crown together with a continuous band of gingiva [196]. Figure 4-1 illustrates excessive gingival display associated with a high lip line before orthodontic treatment and an ideal lip line following treatment. Another characteristic is the smile arc, which describes the relationship between the curvature of the incisal edges of the maxillary incisors and canines and the curvature of the lower lip during a posed smile [197]. Representative ideal, flat, and reversed smile-arc configurations are shown in Fig. 4-2 [198]. Gingival display and smile arc were selected during development of the proof-of-concept because of their recognised influence on smile aesthetics and their prominence within assessments of smile attractiveness [199; 200; 190].

Conventional assessment of these characteristics relies substantially on visual interpretation by a dental professional. Although digital photography is widely used for clinical documentation and aesthetic assessment [201], the interpretation of these images remains dependent on subjective aesthetic judgement and clinical experience. Different dental specialities may place differing emphasis on individual smile characteristics [202], while the medium through which a smile is presented has also been shown to influence its perceived attractiveness [203]. This introduces variability into an assessment process in which comparatively small differences in the position and relationship of the lips, gingiva, and teeth can alter the resulting aesthetic judgement. Such variability may be particularly relevant during preliminary assessment or where the practitioner has less experience in aesthetic smile analysis. Automated analysis therefore provides an opportunity to introduce an additional and reproducible source of information against which visual assessment can be considered. The intention is not to replace professional judgement, but to provide an assistive service capable of consistently identifying selected smile characteristics and supporting subsequent professional interpretation.

The image-based nature of smile assessment makes computer vision a suitable basis for this form of automation. As reviewed in Section 3.2.2, image classification and segmentation provide mechanisms through which visually identifiable characteristics can be extracted from facial and health-related imagery. Existing dental computer-vision research has similarly demonstrated the use of deep-learning methods for detecting and segmenting dental structures [204; 205]. The present application applies these principles to frontal smile imagery rather than radiographic or intraoral images. A captured facial image is first processed to identify and isolate the mouth region before the resulting image is standardised for model inference. Gingival display can then be formulated as an image-classification problem in which the model distinguishes between normal and excessive display.

The development of an automated smile-analysis capability also aligns with the user-centred healthcare interactions envisioned within the IoM architecture established in Chapter 2. A smart mirror provides a natural interface for acquiring frontal facial imagery while simultaneously presenting analysis and guidance to the user. Within an IoM setting, this creates the possibility for an initial assessment to be performed through a Consumer node and, where appropriate, for the resulting information to support subsequent interaction with a Professional node. The application is therefore intended as an assistive mechanism through which automated analysis can complement rather than replace professional assessment, while demonstrating how sensing, image analysis, personalised feedback, and professional interaction can be combined within an IoM service.

Realising this application required a sufficiently reliable image-analysis model to identify the selected smile characteristics from user imagery. However, unlike more established computer-vision domains, an adequate publicly available dataset of labelled frontal dental smiles was not available for the intended classification task. The rel-

atively limited availability of suitable images creates a particular challenge for deep-learning methods, which are susceptible to overfitting and poor generalisation when trained using small datasets. Addressing this limitation therefore required the development of a dedicated dental smile dataset together with augmentation and preprocessing techniques intended to increase the diversity and consistency of the available training data. The resulting dataset development and image preprocessing methodology are described in the following subsection.

4.2.2 Methodology

In this section, the steps taken to develop this use case are presented, detailing the data collection process, preprocessing steps, and the architecture of the CNN used for smile classification. A CNN was selected because the target task is an image-level binary classification problem in which a pre-localised mouth region is classified as either "gummy" or "normal". The localisation stage is already performed during preprocessing through facial landmark detection and cropping, meaning that an object-detection architecture such as YOLO would introduce an additional detection stage that is not required for the present task. Although YOLO architectures are well suited to real-time object detection, real-time localisation was not required within the developed pipeline. A CNN therefore provides a more direct architecture for the classification objective and the available class-labelled dataset [120]. YOLO-based approaches may instead be appropriate in future work where localisation and classification are performed jointly from uncropped facial imagery.

Generative-AI-Driven Dataset Development: To train CNN models to an acceptable degree, a large dataset is required. There are many resources of fully curated and often readily preprocessed datasets of images for various computer vision uses online. In the context of dental smiles, the absence of a suitable dataset in open-source repositories posed a significant challenge. Consequently, the ethical collection of a diverse array of dental images encompassing various ethnicities, genders, and degrees of gingival display became a persistent endeavour.

In total, two sets of 256 (counting 512 combined) images were collected and evenly distributed into "gummy" and "normal" classes. This distribution into classes was validated by a dentist to ensure clinical accuracy. The first set was manually curated from publicly available pictures scrapped from online frontal face images of people smiling. These included highly positive excessive gingival display samples as well as more discrete ones. The second set was collected from text-to-image AI generated images using Adobe Firefly's diffusion model[1]. The text prompts for the "normal" class varied from "frontal portrait of a person smiling with teeth; this person does not have a gummy smile or braces" to selecting relevant images generated from the broader prompt

[1] `https://www.adobe.com/products/firefly/discover/firefly-vs-stable-diffusion.html`

"frontal portrait of a person smiling". Generating images showing excessive gingival display required greater prompt refinement. During data collection, prompts using the clinical term "excessive gingival display" did not consistently produce images exhibiting the intended characteristic, while more descriptive or commonly used terms such as "gummy smile" and spatial descriptions of the gingival exposure produced more suitable candidates. This should be interpreted as an observed limitation of prompt effectiveness within the selected Firefly configuration rather than as a general limitation of diffusion models in interpreting medical terminology. Generated images were therefore manually screened, and only images exhibiting the required visual characteristics were retained in the dataset.

Both sets contributed 56 randomly chosen images each to the testing folder, setting aside 112 of the original 512 images for testing. The remaining 400 (200 from real, 200 AI-generated by the diffusion model) images were randomly split with 80% assigned to training and 20% to validation. In total, three separate datasets were curated: real images (256 images), AI-generated images (256 images), and combined images (512 images). Compared to other computer vision classification datasets, the model trained on this dataset was more vulnerable to overfitting. Therefore, it was vital that the quality of the dataset was superlative for the model to classify unseen images accurately. This was ensured by balancing a diverse range of individual frontal smile images evenly between the two classes and validating the correctness of the labels with a professional dentist.

Data Preprocessing: Given the relatively limited size of the dataset, proactive measures were taken in the preprocessing step to mitigate the risk of overfitting. Firstly, all images were normalised and converted to grayscale using the OpenCV library [2]. Subsequently, each image underwent a process where a pre-trained facial landmark detector from the Dlib library was utilised to pinpoint the coordinates of the mouth, followed by cropping every image to a 28 by 28 pixel size centred around the mouth. This systematic preprocessing step that remained consistent on training, validation, and testing data contributed to the overall quality and consistency of the dataset. The 28×28 representation was adopted as a fixed, compact input size after isolating the mouth region, providing a consistent representation across the heterogeneous source images and reducing the dimensionality of the proof-of-concept classification problem. However, downsampling to this resolution may remove fine-grained spatial information. The reported classification performance should therefore be interpreted within this input representation.

Before feeding these images into the model, multiple augmentation transformations were applied to artificially expand the training data. Each training image was retained in its original form and supplemented with six augmented variants: one horizontal flip, three random zoom transformations with factors between -0.2 and 0.2, one random

[2] `https://opencv.org/`

contrast transformation with a factor between 0.25 and 0.75, and one random brightness transformation with a factor between −0.5 and 0.5. The resulting sevenfold expansion therefore arose from the six augmentation variants plus the original image. This yielded 3584 images in total.

The pipeline of the entire process is presented in Figure 4-3 as a flowchart from receiving raw images to preprocessing to training with an example Adobe Firefly image. On the right is a visualisation of the process a single image goes through. The image used is a random sample image from the Adobe Firefly AI-generated dataset with some of the included data augmentations.

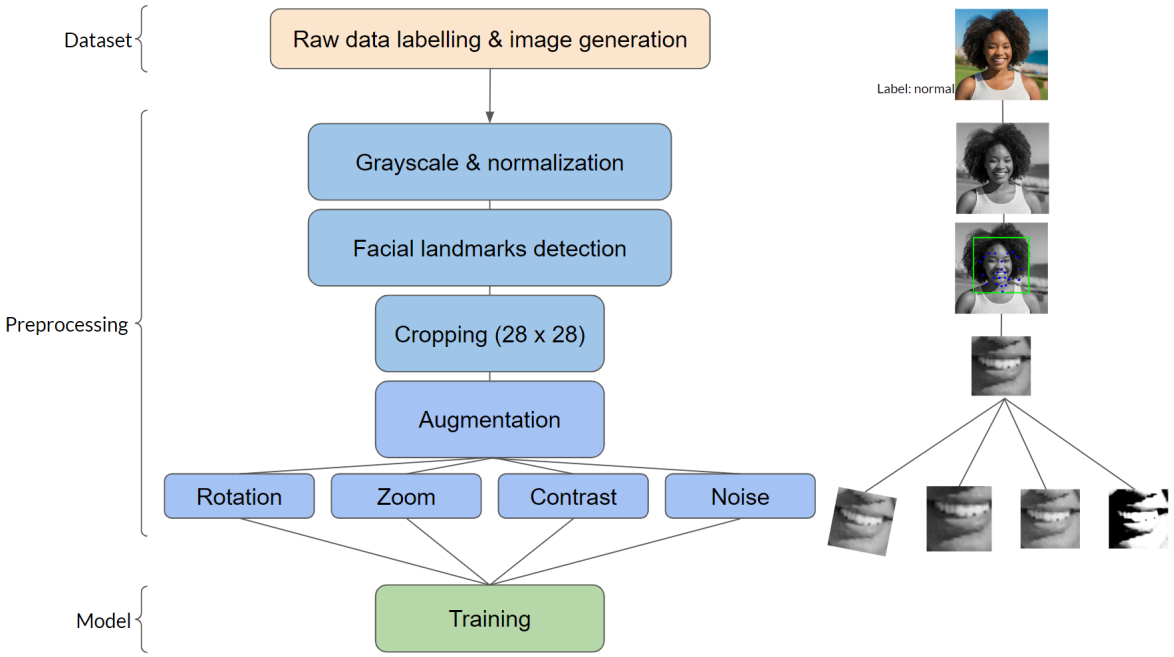


Figure 4-3: Diagram of the dental smile analysis pipeline from data collection to preprocessing to CNN training.

Model Architecture: A CNN classifier model was built using Tensorflow and keras to classify previously unseen images into either "gummy" or "normal" smile classes. A sequential model architecture was selected for its simplicity, offering a linear topology that enables the customisation of layer stacking while maintaining one input tensor and one output tensor. This choice is driven by the primary goal of minimising the risk of overtraining, as small datasets have a tendency to not generalise to unseen testing data. The overall architecture of the model follows a standard layer sequence. The pattern of 2D convolution layer followed by 2D max pooling is used twice. The rectified linear unit (ReLU) function, represented by $R(z) = \max(0, z)$, served as the activation parameter. The ReLU activation function was also used in the first of the three densely-connected NN layers. The second dense layer used the softmax $s(x_i)$ function, which takes a vector x_i and outputs the probability distribution of each class for N classes, denoted by:

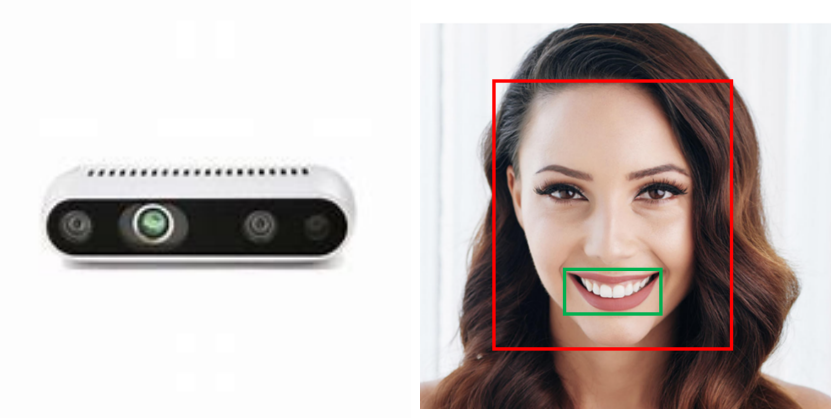
$$s(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}}. \tag{4-1}$$

The final dense layer converted the output into a probability distribution across the classes. Each model, distinguished by real, AI-generated, and combined datasets, was trained using the Adam optimizer [206]. Through hyperparameter tuning, the optimal number of epochs was determined to be at 150 and a batch size 32. All parameters remained constant throughout training sessions.

IoM User Interface: A prototype UI was designed for integration into the envisaged IoM interactive dashboard. A basic layout of applications was created, allowing for flexible addition of functionalities to accommodate the evolving functionalities of the application, as shown in Figure 4-4. For instance, Deep Learning (DL) models trained to detect early stages of oral cancer could be integrated into the dashboard.



Figure 4-4: UI layout for an assistive diagnostic application.



(a) Intel RealSense Depth Camera D435 used for image acquisition.

(b) Illustration of the bounding-box tracking displayed during image capture.

Figure 4-5: Image-acquisition hardware and visual feedback used in the dental smile-analysis interface.

Upon pressing the Smile Analysis button, the system initiates video capture from the Intel RealSense Depth Camera D435, shown in Fig. 4-5 (a), accompanied by the

bounding-box tracking of the user's face and smile on the screen, as illustrated in Fig. 4-5 (b). Simultaneously, pre-trained Haar cascades for face and smile detection operate in the background. The face is indicated by a red bounding box, while the smile bounding box changes from red to green when the user maintains a smile for two seconds, triggering image capture. The captured image then enters the standardised preprocessing pipeline and is analysed by the classification model, with the resulting class displayed on the screen.

4.2.3 Use Case Evaluation and Discussion

Gingival Display Classification: The CNN trained using the combined real and AI-generated dataset achieved a mean test accuracy of 81.607% with a SD of 1.938 across ten independent training trials. Each model was trained for 150 epochs using the configuration described in Section 4.2.2 and subsequently evaluated using the test dataset. The repeated trials were used to account for variation arising from model initialisation and training rather than relying on the outcome of a single training run.

Figure 4-6 shows the training and validation accuracy for a representative run, while Fig. 4-7 shows the corresponding training and validation loss. Both training and validation accuracy increased during the initial training period, although validation performance remained below the training performance. The loss curves showed the corresponding inverse behaviour, with both training and validation loss decreasing during the earlier stages of training. Around 150 epochs, the validation loss began to diverge from the continuing reduction in training loss, indicating that further training was improving the fit to the training data without producing a corresponding improvement in validation performance. Considered together with the divergence in training and validation accuracy, this behaviour indicated increasing overfitting and informed the selection of 150 epochs for the final training configuration.

To examine the contribution of the different image sources, the combined-data model was compared with models trained using only the manually curated real images and only the AI-generated images. Each configuration was trained and evaluated across ten independent trials under the same model settings. Figure 4-8 presents box and whisker plot showing the distribution of test accuracy obtained for the three configurations. The real-only and AI-generated-only models exhibited broadly comparable performance, while the model trained using the combined dataset achieved the highest median test accuracy. This indicates that the combined configuration provided the strongest performance within the evaluated experimental setup.

A representative run of the combined-data model achieved an accuracy of 82.14% and an F1 score of 0.82. The corresponding confusion matrix is shown in Fig. 4-9. Taking the "gummy" class as the positive class, 45 images with excessive gingival display and 47 normal images were classified correctly. Eleven images with excessive gingival display were incorrectly classified as normal, representing false negatives, while nine normal

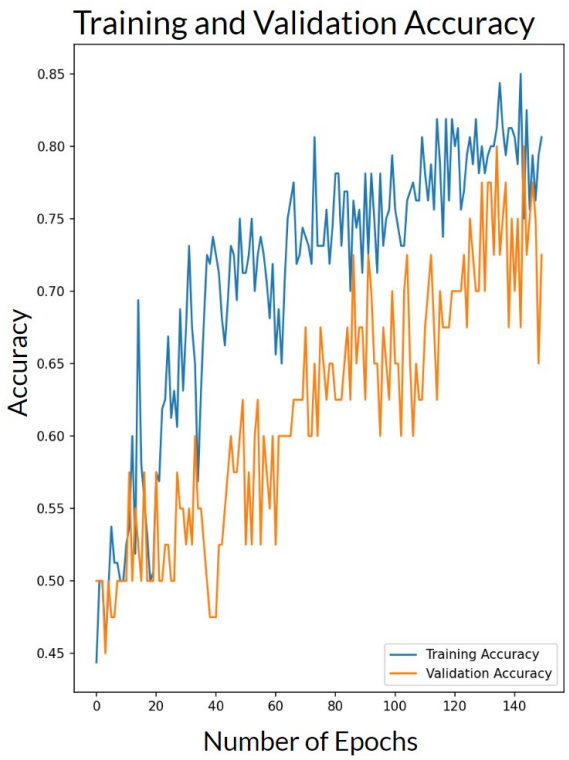


Figure 4-6: Training and validation accuracy of the CNN model.

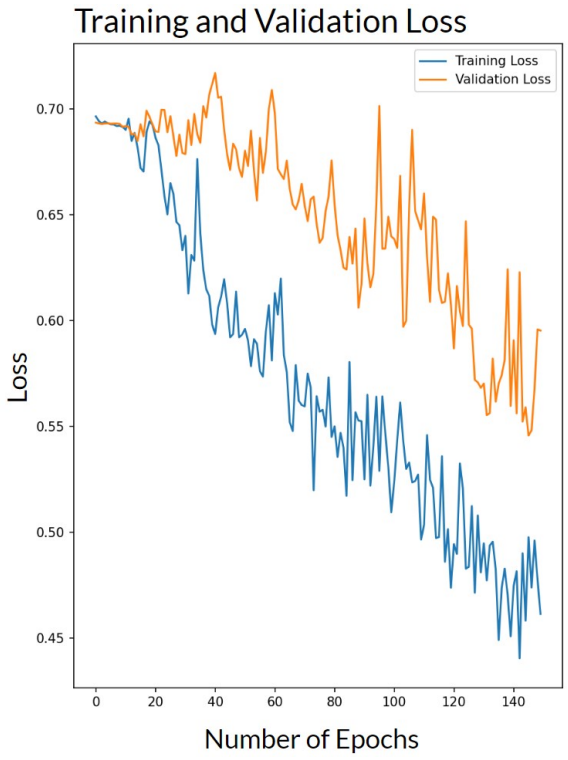


Figure 4-7: Training and validation loss of the CNN model.

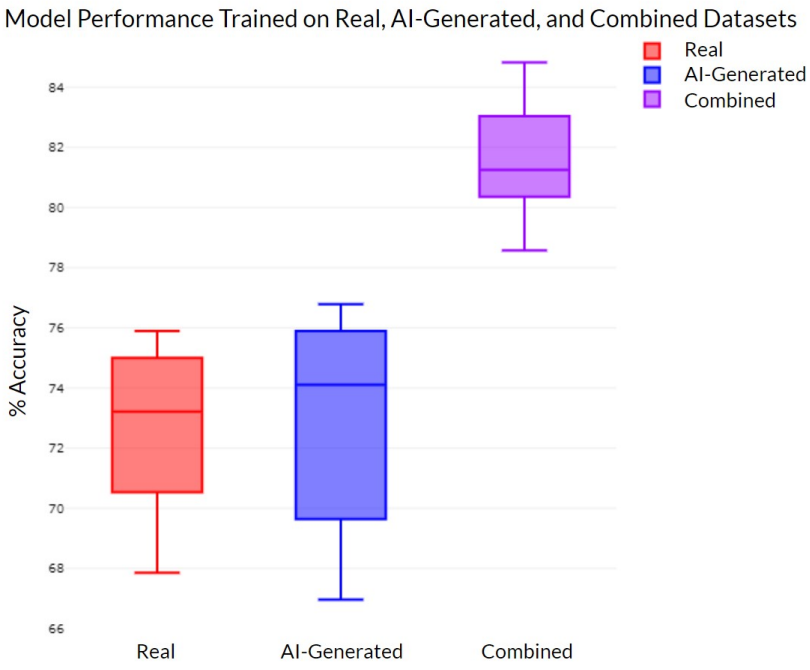


Figure 4-8: Box plots of the test accuracy obtained using models trained on real, AI-generated, and combined datasets.

images were incorrectly classified as exhibiting excessive gingival display, representing false positives. In the context of the proposed assistive application, false negatives are particularly relevant because they represent cases in which excessive gingival display would not be flagged for further consideration.

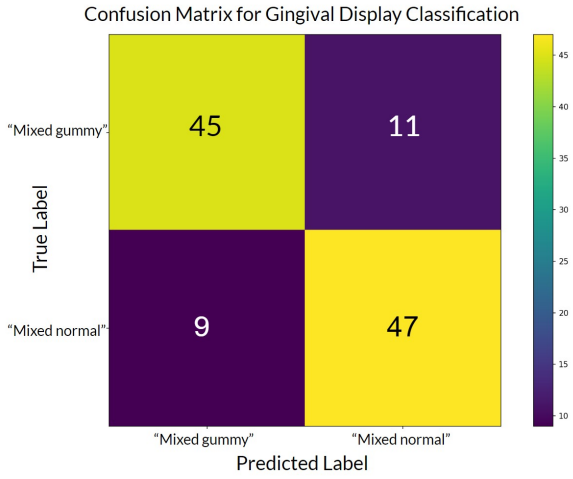


Figure 4-9: Confusion matrix for a representative run of the combined-data CNN model.

Findings and Limitations: The experimental results demonstrate the feasibility of using the developed CNN to distinguish excessive from normal gingival display from frontal smile imagery. The combined real and AI-generated dataset produced the highest mean test accuracy of 81.607% within the evaluated configurations, compared with a 50% baseline for the balanced binary classification problem. This provides evidence that the target visual characteristic can be learned from the available imagery. However, the remaining classification error and comparatively limited dataset mean that the resulting model should be interpreted as a proof-of-concept assistive analysis tool rather than as a clinically validated diagnostic system.

The relatively small dataset also enabled repeated model training to be performed at manageable computational cost, allowing the reported performance to be evaluated across ten independent trials rather than from a single training run. Nevertheless, dataset size remains an important limitation. The manually curated real images vary in image quality, lighting, capture conditions, and demographic representation, while AI-generated images may contain synthetic artefacts or biases introduced by the underlying generative model. Neither image source can therefore be assumed to represent the distribution of images that would be encountered during real-world use. The combined dataset increases the diversity of the imagery available during model development, but evaluation using a substantially larger and independently collected clinical dataset would be required before claims regarding population-level reliability or clinical applicability could be made.

The comparison between the three training configurations should also be interpreted in the context of their differing dataset sizes. The combined configuration contains both the real and AI-generated samples and therefore provides a larger training set than either source-specific model. Although the combined configuration achieved the strongest performance, the present experiment does not isolate whether this improvement results specifically from combining real and generated imagery, from the increased number of training samples, or from a combination of both factors. A controlled eval-

uation using equal-sized training sets with different proportions of real and generated imagery would be required to separate these effects.

The smart mirror implementation introduced an additional practical consideration associated with image acquisition. Reliable analysis depends on capturing a suitably positioned and posed dental smile, and variation in facial orientation, expression, lighting, or distance from the camera can alter the input presented to the classifier. The integration of face and smile detection within the interface provided real-time guidance and enabled automated capture once a suitable smile was detected. This reduced dependence on manual image acquisition, but does not eliminate variation in the resulting input imagery. More robust control and validation of image acquisition would therefore be required for deployment beyond the present proof-of-concept.

During local operation of the application, smile classification was observed to return rapidly once the image had been acquired and pre-processed, while the most noticeable delay occurred during initial camera acquisition. This demonstrated that the developed model could support responsive interaction when executed within the standalone smart mirror implementation. However, this behaviour characterises only local execution. The IoM architecture allows computational resources to be distributed across connected Consumer, Professional, and Hub nodes, meaning that the processing stages associated with an application need not necessarily execute on the same device on which the user interaction occurs. Transferring computation away from the user-facing mirror may provide access to more capable processing resources, but additionally introduces communication latency and dependence on the available network.

The dental smile-analysis application therefore serves two related purposes within this thesis. First, it demonstrates the feasibility of integrating image acquisition, computer-vision-based analysis, and personalised result delivery within a user-facing smart mirror as an assistive IoM proof-of-concept. Second, its operation can be decomposed into distinct image acquisition, preprocessing, classification, and result-delivery stages, providing a consistent application workload through which the system-level consequences of distributed execution can be examined. Section 4.3 therefore formalises this processing pipeline for distributed execution and uses it to investigate the performance trade-off between local edge and remote cloud computation.

4.3 Initial IoM Performance Evaluation: Computational Placement Between Edge and Cloud

The proof-of-concept developed in Section 4.2 demonstrates that automated dental smile analysis can be implemented as an end-to-end smart mirror application when its processing stages are executed within a local system. The wider IoM architecture, however, is inherently distributed and does not require all computation associated with a service to remain at the user-facing node. As established in Section 3.3.2, connected

IoT services can access computational resources positioned at different locations across the computing continuum, introducing a trade-off between the processing capability available at a candidate resource and the communication required to access it. The next stage of the investigation therefore examines how this trade-off affects the realised IoM application when its computation is divided between local edge and remote cloud resources.

Edge and cloud computing represent two contrasting execution locations through which this initial placement question can be examined. Local edge processing retains computation close to the data source and avoids dependence on remote transmission, while cloud execution provides access to remotely hosted computational resources but requires application data and results to traverse the network. These characteristics were reviewed more broadly in Section 3.3, the purpose of the present experiment is therefore not to establish the general advantages of either paradigm, but to quantify how their differing processing and communication paths affect the performance of a concrete IoM workload.

Latency is used as the principal performance measure because the IoM is intended to provide interactive user-facing services for which delays can affect responsiveness and user experience. The significance of latency nevertheless depends on the application being delivered. Time-critical monitoring or emergency healthcare services may impose substantially stricter response requirements than an elective application such as dental smile analysis [23]. Dental smile analysis does not require the immediacy of a safety-critical service, but the interaction remains user-facing and therefore benefits from timely image acquisition, processing, and result delivery. More generally, system latency has been shown to influence user satisfaction and interaction performance, with acceptable delay varying according to the task and user expectations [207]. The experiment therefore focuses on latency not as a universal service requirement, but as a measurable system characteristic through which the consequences of computational placement can be examined.

The dental smile-analysis pipeline developed in Section 4.2 is retained as a consistent workload throughout this comparison. This allows the execution environment to be varied without changing the underlying application functionality. The application workflow is decomposed into image acquisition, preprocessing, and classification as the principal processing stages. Two implementations are evaluated: a local edge configuration in which the full pipeline is executed using an NVIDIA Jetson Xavier device [3], and a cloud configuration in which image acquisition and preprocessing remain local while classification is offloaded to Google Cloud Platform [4]. The comparison is published in [C2] and addresses O_2b by identifying where latency arises under each configuration and establishing an initial empirical basis for understanding computation placement

[3] https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-xavier-series/

[4] https://cloud.google.com/

within the IoM.

4.3.1 System Architecture

To enable a precise comparison between edge and cloud computing paradigms, separate system architectures were developed for each approach. Two execution configurations were implemented for the dental smile-analysis pipeline established in Section 4.2, allowing the same application stages to be evaluated under local edge and remote cloud execution. By focusing on computational latency as the primary performance metric, the modular setup ensures that the testing methodology can be consistently applied across both architectures, allowing for a direct comparison of their performance under identical conditions.

Workload Processing Pipeline: The dental smile-analysis application presented in Section 4.2 was adapted to support distributed execution between local edge and remote cloud resources. The underlying image-analysis functionality remained unchanged, with the preprocessing procedure and gingival-display classifier developed in Section 4.2.2 retained. For the edge–cloud evaluation, the end-to-end workflow comprised the following components:

- **Image Capture:** A dental smile image is acquired through the smart mirror camera using the capture procedure established in Section 4.2.2.
- **Image Preprocessing:** The captured image is normalised and converted to grayscale, after which a pre-trained Dlib facial landmark detector is used to identify the mouth region. The image is then cropped to a 28×28 pixel region centred on the mouth to produce the input required by the classifier. These operations constitute the preprocessing stage measured throughout the edge-cloud evaluation.
- **Image Classification:** The processed image is provided to the trained gingival-display CNN. In the edge configuration this stage is executed locally, while in the cloud configuration it is executed remotely.

To enable remote execution, the trained classification model and its inference dependencies were packaged as a containerised service and deployed using Google Cloud Run [5]. This produced two implementations of the same application workflow: a fully local edge configuration and a cloud-offloaded configuration in which image capture and preprocessing remained local while classification was executed remotely.

[5] `https://cloud.google.com/run`

Edge Computing Setup: In the edge configuration, all stages of the processing pipeline defined in Section 4.3.1 are executed locally on the NVIDIA Jetson Xavier edge device. The system architecture, shown in Fig. 4-10, includes the following key components:

- **Intel RealSense Camera:** Captures high-quality images with depth sensing capabilities, providing raw input for the smile analysis process.
- **Smart Mirror Interface:** Displays real-time feedback to users, showing analysis results directly through the smart mirror's UI.
- **Jetson Nvidia Edge Device:**
  - Manages image capture, preprocessing, and analysis stages locally.
  - Executes the trained gingival-display CNN developed and evaluated in Section 4.2.3.
  - Retains the application data locally, avoiding the external transmission required by the cloud configuration.

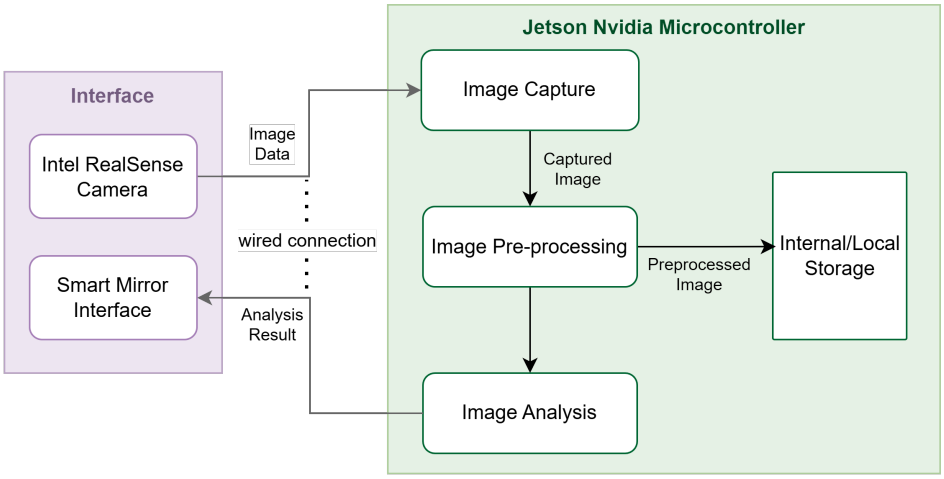


Figure 4-10: Edge computing architecture for dental smile analysis.

In this setup, all operations, from image capture to smile classification are performed on the edge device. The Jetson Nvidia platform was selected for its energy efficiency and robust processing power, making it ideal for edge deployment in resource-constrained environments, like smart mirrors.

Cloud Computing Setup: The cloud configuration distributes the processing pipeline between the local smart mirror client and Google Cloud Platform. Image acquisition and preprocessing remain local, while the containerised classification service is executed remotely through Google Cloud Run before the resulting class is returned to the local interface for result delivery. The system architecture is shown in Fig. 4-11.

- **Local Client:**

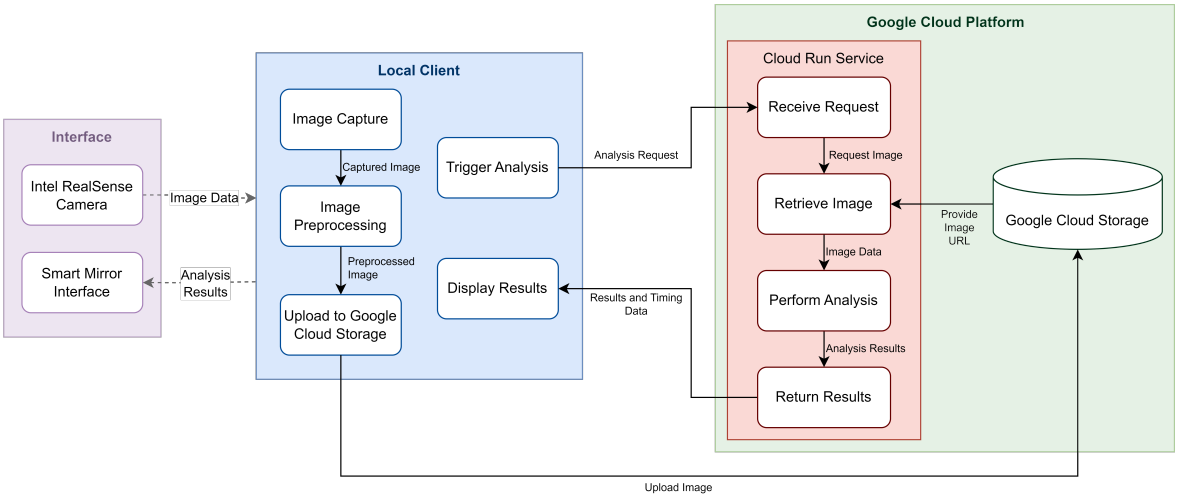


Figure 4-11: Cloud computing architecture for dental smile analysis.

  - Handles initial image capture and basic preprocessing tasks.
  - Displays results to the user through the smart mirror interface after analysis.
  - Uploads pre-processed images to Google Cloud Storage, triggering the cloud-based analysis.

- **Google Cloud Platform:**
  - **Cloud Run Service:** Hosts the containerised CNN model for image classification.
  - **Google Cloud Storage:** Stores the pre-processed images temporarily during analysis.

4.3.2 Methodology

Key components of the methodology for this evaluation and comparison of approaches include:

1. **System Decomposition:** To enable a precise comparison, the principal processing operations were divided into three measured stages: image capture, preprocessing, and image classification. For the cloud configuration, communication associated with remote execution was measured separately through uplink and downlink latency, with the downlink corresponding to the return of the classification result to the local client.

2. **Edge Computing Set-up:** Using the Jetson Nvidia edge device with the intel RealSense camera, the edge architecture was set up as shown in Fig. 4-10.

3. **Cloud-Computing Set-up:** After establishing the edge set-up, the most computationally intensive task, the smile classification, was offloaded onto the cloud. Google Cloud Platform's Cloud Run services hosted the smile analysis model,

while image capture and preprocessing were still performed on the local client. An overview of the network elements for this set-up is as follows:

- Local client: Performing image capture and preprocessing
- Wi-Fi router: Providing internet connection
- ISP network: Routing traffic to Google's infrastructure
- Google Cloud Platform: Hosting the Cloud Run service for smile analysis

4. **Latency Measurement:** For latency analysis, the first three pipeline stages were timed as processing components, while communication associated with remote classification and result return was measured separately through the uplink and downlink components. Precise timing mechanisms were implemented to measure latency at each stage of the process. Python's time.time() function was used for high-precision timing, wrapping each process in the start and end time measurements. The difference between these timestamps provided the latency for each stage.:

   - **Image Capture Latency** was measured from the time of initiating the capture command to the successful acquisition of the image. To focus on system processing performance, the user-dependent time of adjustment in the frame was measured separately and excluded from the image capture latency.
   - **Preprocessing Latency** was measured from the start of image normalisation and grayscale conversion to the completion of facial landmark detection and cropping of the 28×28 mouth region.
   - **Image Classification Latency** was measured from the moment the processed image was input into the CNN model until the classification result was returned.
   - **Network Latency:** In the cloud setup, a standard Wi-Fi connection was used for which network latency was measured, distinguishing between uplink and downlink components:
     - **Uplink Latency:** The time from when the preprocessed image was sent from the local client to the cloud until confirmation of receipt by the cloud server.
     - **Downlink Latency:** The time from when the cloud server began sending the classification result until it was fully received by the local client.

5. **Experimental Validation:** 100 iterations of the complete dental smile analysis process was conducted for both edge and cloud implementations. Each iteration consisted of:

   - Capturing a new dental smile image

- Preprocessing the captured image
- Performing smile classification (either on the edge device or in the cloud)
- Recording the latency for each stage and the total process

All iterations were performed under consistent lighting and environmental conditions to ensure comparability. To validate the statistical significance of the performance differences between edge and cloud computing, independent two-sample t-tests were conducted for each processing stage and the overall latency [208].

4.3.3 Results

This section presents the findings from the comparative analysis of the edge and cloud computing setups. The performance characteristics of both implementations are examined, highlighting key differences in latency and processing time across different stages of the analysis pipeline.

Edge Computing Performance: Fig. 4-12 illustrates the breakdown of latency across different processing stages. The total processing latency for the edge setup averaged 5.140 seconds, distributed as follows:

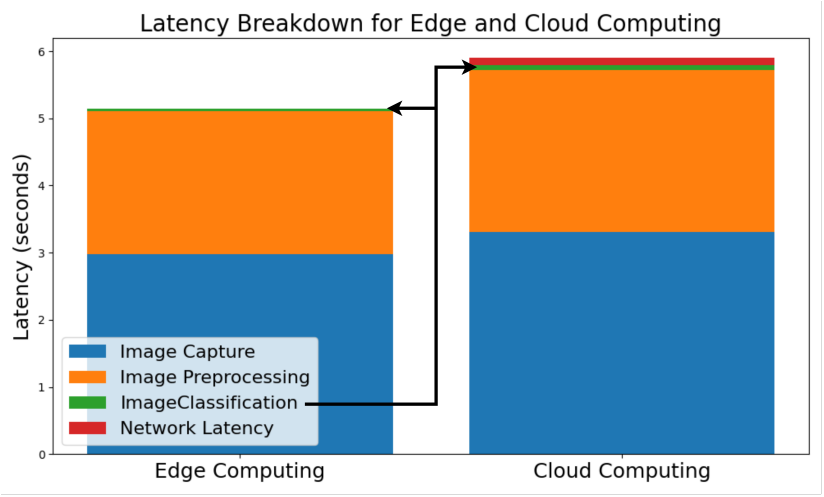


Figure 4-12: Latency component breakdown for edge and cloud computing systems.

- Image Capture: 2.972 seconds (57.8%)
- Image Preprocessing: 2.137 seconds (41.6%)
- Image Classification: 0.028 seconds (0.6%)

The cloud setup exhibited slightly higher latencies in image capture and preprocessing compared with the edge setup, despite the same local hardware and preprocessing procedure being used in both configurations. A plausible explanation is the additional

implementation overhead introduced by adapting the application for cloud execution, including Docker containerisation, Google Cloud Run integration, and the associated preparation of data for remote transmission. Although this overhead was not isolated experimentally, the observed difference indicates that processing latency can be influenced not only by the computational task itself, but also by the wider implementation workflow through which that task is executed. This reinforces the importance of considering the complete processing and communication pipeline when evaluating computational placement within the IoM. The analysis and classification stage exhibited a relatively lower latency (0.028 seconds), showing the efficiency of the implemented CNN model.

Cloud Computing Performance: The cloud-based implementation of the dental smile analysis system yielded different results when compared to the edge setup. Fig. 4-13 provides a side-by-side comparison of the latency components for both setups. The total processing latency for the cloud setup averaged 5.897 seconds, distributed as follows:

- Image Capture: 3.306 seconds (56.1%)
- Image Preprocessing: 2.414 seconds (40.9%)
- Image Classification: 0.177 seconds (3.0%)

In addition to the processing latency, the cloud configuration incurred an average uplink network latency of 0.166 seconds. Network latency is reported separately because it is introduced only by the remote cloud configuration and is therefore excluded from the direct comparison of processing latency between the two implementations. The network latency primarily occurred during the uplink process when images were uploaded to the cloud for processing, with the downlink (returning the result) being negligible in comparison.

The cloud setup exhibited higher latencies in image capture and preprocessing compared to the edge setup. This may be attributed to the additional overhead of preparing images for network transmission in the cloud scenario.

Comparative Analysis: Fig. 4-14 provides a focused comparison of the image-classification latency for the edge and cloud setups together with the additional network latency introduced by cloud execution. Independent two-sample t-tests were used to compare the mean processing latencies of the edge and cloud configurations for each measured stage and for the overall processing latency. Statistical significance was assessed using a threshold of $\alpha = 0.05$. A p-value smaller than 0.05 ($p < 0.05$) was therefore interpreted as evidence of a statistically significant difference between the mean latencies, whereas a p-value greater than or equal to 0.05 ($p \geq 0.05$) would not provide sufficient evidence to reject the null hypothesis of no difference. The p-values

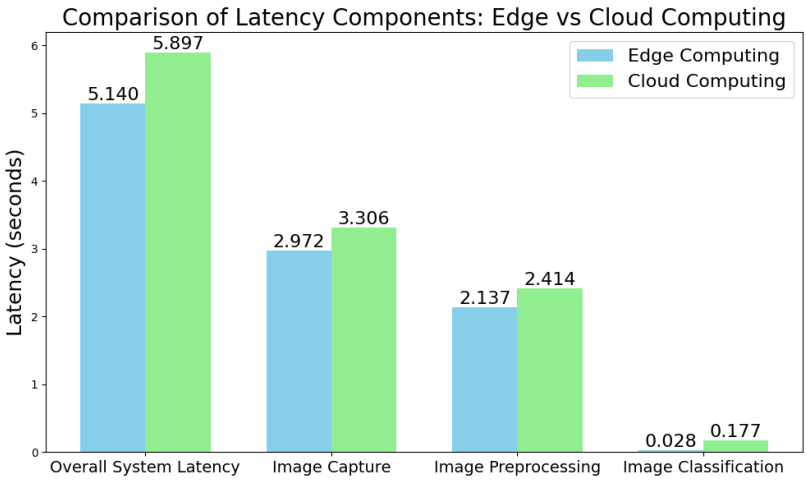


Figure 4-13: Latency component comparison for edge vs cloud.

are interpreted alongside the observed latency differences, since statistical significance alone does not indicate the magnitude or practical importance of an effect. Key observations from the comparative analysis include:

1. **Overall Processing Latency:** The edge setup outperformed the cloud setup by approximately 0.757 seconds (($p < 0.001$)), primarily due to reduced latency in the image capture and preprocessing stages.

2. **Image Capture:** Edge computing showed significantly lower latency ($p = 0.007$) compared to cloud computing for image capture.

3. **Image Preprocessing:** The edge setup demonstrated significantly faster preprocessing times (($p < 0.001$)) compared to the cloud setup.

4. **Image Classification:** The cloud classification latency was 0.177 seconds compared with 0.028 seconds for the edge configuration. This represents processing latency only and excludes the additional network communication required for cloud execution. The difference in classification times between the two setups was statistically significant ($p < 0.001$).

5. **Network Latency:** The cloud configuration incurred an additional uplink network latency of 0.166 seconds as the preprocessed image was transmitted for cloud processing. The downlink latency associated with returning the classification result was negligible in comparison due to the small size of the result.

6. **Scalability:** While not directly measured, the cloud setup offers superior scalability for handling multiple simultaneous requests, a crucial factor for large-scale IoM deployments.

7. **Flexibility:** The cloud setup enables easier updates and improvements to the analysis model without requiring changes to the edge devices, providing significant advantages for evolving IoM applications.

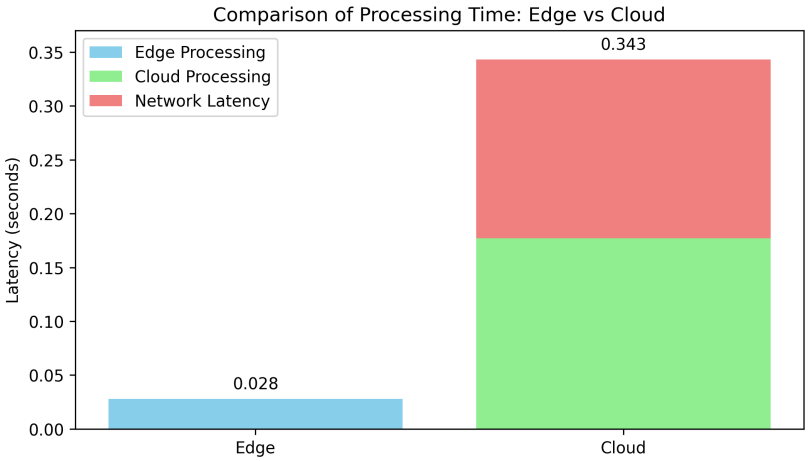


Figure 4-14: Comparison of image-classification latency and additional network latency for the edge and cloud configurations.

Performance Variability: The 100 iterations conducted for each setup also provide an indication of the variability in observed latency. The edge setup exhibited an SD of 0.152 seconds for overall latency, compared with 0.283 seconds for the cloud setup. The larger SD therefore indicates greater trial-to-trial dispersion in the cloud latency measurements under the evaluated conditions. This is consistent with the additional network-dependent component introduced by remote cloud execution, although the contribution of network conditions to the observed variability was not isolated experimentally. As the statistical tests conducted in this study compare mean latencies rather than variances, no statistical significance is attributed to the difference in SD itself.

4.3.4 Discussion and Implications

The edge-cloud comparison provides an initial empirical characterisation of how computational placement affects the performance of the developed IoM application. The results show that the two configurations exhibit different processing and communication characteristics, with the overall service latency determined by the complete application pipeline rather than by model inference alone. The implications of these findings are considered below in terms of the observed latency bottlenecks, the trade-offs between local and remote execution, and their relevance to distributed operation within the wider IoM architecture.

Latency and Processing Bottlenecks: The overall processing latencies of the edge and cloud setups differed by 0.757 s, with the edge configuration exhibiting the lower mean latency. This difference was statistically significant ($p < 0.001$), with the p-value falling below the predefined significance threshold of 0.05. This could impact user experience in interactive IoM applications such as real-time makeup try-ons or remote dermatological treatment, where immediate feedback is critical. Bottlenecks primarily

lie in image capture and preprocessing, suggesting that advancements in sensors and image processing could enhance IoM performance.

Edge vs. Cloud Computational Trade-offs:

- **Edge Computing Advantages:** Edge computing demonstrates consistently lower latency, making it suitable for time-sensitive IoM applications such as home security systems in smart homes, where instant alerts are needed for security breaches. Its ability to function independently of network connectivity enhances system reliability, especially in environments where stable internet access is not guaranteed. Additionally, local processing preserves privacy by minimising data transmission to external servers.
- **Cloud Computing Advantages:** Despite higher latency, cloud computing provides superior scalability and the capability to handle more complex analyses. Cloud-based setups can support large numbers of simultaneous users and facilitate complex processing tasks that exceed the capabilities of edge devices. This makes it ideal for IoM applications where scalability outweighs low latency, such as eco-friendly inventory management, where analysing customer interactions and predicting product demand would need cloud computing's computational power.

These trade-offs suggest that the optimal architecture for IoM applications may often involve a hybrid approach, combining the low-latency benefits of edge computing with the scalability and processing power of cloud offloading.

Implications for Distributed IoM Operation: The results provide several implications for the wider operation of distributed IoM services:

1. **Distributed Execution:** The application pipeline does not need to be treated as an indivisible workload. Image acquisition, preprocessing, classification, and result delivery can potentially be assigned to different computational resources according to their individual requirements.
2. **Processing Bottlenecks:** The most computationally intensive model stage is not necessarily the dominant contributor to end-to-end latency. Placement decisions should therefore consider the complete processing pipeline rather than assuming that offloading inference alone will provide the greatest performance improvement.
3. **Network Considerations:** Remote execution introduces communication as an additional component of service performance. The uplink required to transfer the processed image accounted for the majority of the measured network overhead in the cloud configuration, demonstrating that the amount and direction of transferred data can influence the benefit obtained through offloading.

4. **Resource Capability:** Moving computation away from a user-facing device can provide access to different levels of processing capability, but the benefit obtained must be considered against the communication required to reach that resource. The suitability of a placement therefore depends on both the computational demand of the task and the characteristics of the candidate execution node.

5. **Data Locality:** Local execution minimises the transmission of user imagery beyond the smart mirror, whereas remote execution requires information to be transferred between devices or infrastructure. Although privacy and security were not evaluated experimentally in this study, data locality therefore represents an additional consideration when selecting an execution location for applications involving personal health or facial information.

6. **Adaptive Placement:** As computational requirements and operating conditions may vary between applications and deployments, a fixed placement decision may not remain appropriate in all circumstances. The observed computation-communication trade-off therefore provides an initial motivation for mechanisms capable of selecting between available resources according to the conditions under which the service is operating.

The edge-cloud comparison establishes an initial empirical understanding of computational placement for the developed IoM application by contrasting local and remote execution under controlled conditions. The experiment demonstrates that placement changes both the magnitude and composition of end-to-end latency and that the resulting performance depends on the interaction between application processing and communication. It therefore provides a first practical step beyond the standalone proof-of-concept towards understanding the IoM as a distributed computing system.

Building on this foundation, the next stage broadens the investigation to reflect the full stakeholder-oriented Consumer, Professional, and Hub hierarchy established in Chapter 2. This enables computational placement to be examined across multiple candidate resources and under a wider range of operating conditions, including different wireless technologies, resource utilisation, and concurrent user demand. The following Section 4.4 therefore extends the evaluation to a physical multi-tier IoM deployment in which several computational placement strategies are assessed under Wi-Fi and 5G connectivity and increasing concurrent load.

4.4 Physical Testbed Evaluation: Computational Placement Across the IoM Tier Hierarchy

The preceding edge-cloud evaluation established that computational placement changes both the magnitude and composition of end-to-end latency for the developed IoM

application. However, the IoM architecture provides a broader distributed environment than the local–remote comparison considered in that initial evaluation. The three-tier hierarchy established in Section 2.2.2 provides Consumer, Professional, and Hub nodes with differing computational capabilities and stakeholder roles, creating several locations across which the processing stages of an IoM service may be distributed. The next stage of the investigation therefore extends the evaluation from two execution environments to a physical implementation of this multi-tier architecture, addressing O_2c and published in [C3].

This extension also introduces operating conditions that become relevant once multiple IoM nodes interact. As established in Chapter 3, communication performance varies with the available network technology and the characteristics of the data being transferred, while distributed execution places computational demand on different resources throughout the system. The interaction between these factors cannot be characterised through processing latency alone. The physical evaluation therefore considers end-to-end and per-stage latency together with inter-tier network transfer, resource utilisation, and performance under increasing concurrent demand. Wi-Fi and 5G are evaluated as alternative communication environments to examine how the network interacts with computational placement under the same application workload.

The dental smile-analysis use case developed in Section 4.2 is retained in the physical testbed evaluation, with the same processing pipeline and containerised workload used in the preceding edge-cloud study. The principal implementation difference is the image-acquisition stage, which uses a different capture path within the physical testbed. The evaluation additionally instantiates the workload across different local computing hardware to represent the Consumer, Professional, and Hub tiers of the IoM architecture. The study therefore varies where the processing stages are executed across the IoM hierarchy rather than the application functionality being delivered.

The absolute latency values reported in this section are consequently not directly comparable with those reported in Section 4.3. The two studies address different questions rather than successive measurements of the same implementation. The previous evaluation established how computational placement affects a service divided between local and remote execution across two endpoints, whereas the present evaluation establishes how placement behaves across the multi-node hierarchy under differing network technologies and concurrent demand. Comparisons throughout this section are therefore drawn between placement strategies evaluated within the same testbed.

4.4.1 System Design

The system design adopts the three-tier IoM node hierarchy established in Section 2.2.2, comprising Consumer, Professional, and Hub nodes. Rather than redefining these tiers, the present evaluation instantiates their differing stakeholder roles and computational contexts within a physical distributed system and examines how the varied steps of the

dental smile-analysis workload can be partitioned between them.

Distributed Application Pipeline: Building on the edge-cloud workflow established in Section 4.3.1, the application is represented as four sequential stages for the physical multi-tier evaluation:

1. **Image Capture:** A 600×600 pixel JPEG image is acquired at the Consumer node using the connected camera.
2. **Image Preprocessing:** The captured image undergoes the preprocessing required to produce the input representation used for inference.
3. **Image Classification:** The processed input is classified using the CNN implementation employed within the physical testbed.
4. **Result Delivery:** The resulting classification is returned to the Consumer node for presentation through the user-facing interface.

Unlike the preceding edge-cloud comparison, these stages are not restricted to a local-client and remote-cloud execution model. Their execution locations can instead be varied across the Consumer, Professional, and Hub tiers, allowing different forms of distributed processing to be evaluated while preserving the same overall application workflow.

Computational Placement Strategies: To represent the spectrum of possible distribution across the IoM, four computational placement strategies are defined by assigning the four stages established in Section 4.4.1 to different tiers within the IoM hierarchy. The strategies therefore preserve the same application workflow while varying the location of processing and the resulting inter-tier communication, as illustrated in Fig. 4-15 and detailed below.

- **Approach 1: Consumer-only** executes all four pipeline stages locally on the consumer node. The hub receives only a lightweight summary log. Network overhead is minimal, but the full classification burden falls on the resource-constrained consumer node. This approach can be envisioned as a remote home-user scenario, in which the patient interacts with a personal mirror without proximity to a professional or hub node, and results are available for asynchronous review by a clinician.
- **Approach 2: Professional-offload** performs image capture and preprocessing on the consumer node, then transfers the pre-processed image to the professional node for CNN classification. The result is returned to the consumer for display, and a log is forwarded to the hub. The network payload at the consumer-to-professional hop is a pre-processed feature image. In a real IoM scenario this can

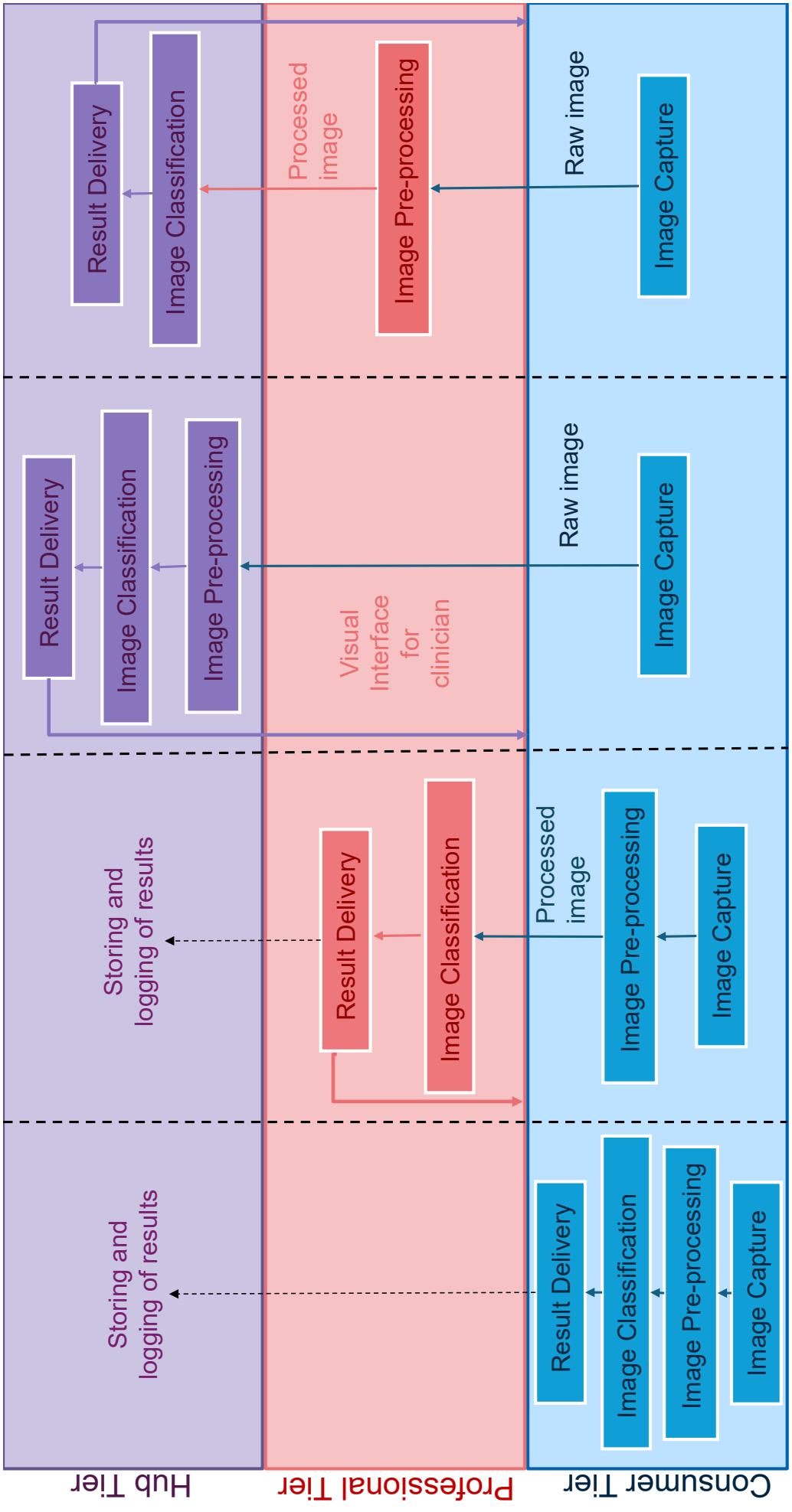


Figure 4-15: Computational placement strategies across the IoM tier hierarchy.

map to a clinic or pharmacy waiting-area scenario, where a walk-up consumer mirror offloads inference to a nearby professional workstation for near-edge, chair-side assessment.

- **Approach 3: Hub-offload** transfers the raw captured image directly from the consumer to the hub, where preprocessing and classification are both performed. The result is returned to the consumer, and a log is retained at the hub. The professional node serves as a visual interface for the clinician rather than a processing node. The raw image payload transferred to the hub is approximately 2–3 MB raw captured image. This strategy is suited to a large multi-room clinic with a reliable network and a centralised server handling simultaneous requests from several mirrors.

- **Approach 4: Tiered-distributed** distributes the processing pipeline across all three IoM tiers. Image capture is performed at the consumer node, after which the raw captured image is transferred to the Professional node for preprocessing. The resulting processed image is then forwarded to the Hub node for classification and result delivery, with the classification result returned to the Consumer node for display. This strategy therefore introduces three inter-tier transfers and exercises computational resources across the complete hierarchy. It represents a deployment in which initial user interaction occurs at the Consumer tier, intermediate image processing is supported by a Professional resource, and the analytical stage is executed at the higher-capability Hub tier.

These four strategies span the range from fully local to fully distributed execution, and together they characterise the computation-communication trade-off space of the IoM tier hierarchy. A summary of the stage-to-tier mapping, network hops, and transmitted payloads for each strategy is provided in Table 4-1.

4.4.2 Experimental Setup

Testbed Hardware: The physical testbed implements the three-tier IoM hierarchy using representative embedded and edge computing hardware comprised of five active computing nodes.

The consumer tier comprises three Raspberry Pi 4 Model B devices (4 GB RAM, quad-core Cortex-A72 CPU), each connected to a USB webcam, representing resource-constrained consumer nodes.

The professional and hub tiers are each implemented on an NVIDIA Jetson Xavier NX, reflecting the higher processing capacity available in clinical or centralised deployment settings. CNN inference at the professional and hub tiers executes on the Volta GPU, while all consumer-tier processing runs on general-purpose CPU cores.

Table 4-1: Summary of computational placement strategies.

Strategy	**Stage-to-Tier Mapping**	**Payload Transferred**	**Hops**	**Deployment Scenario**
Consumer-only	Stages 1–4: Consumer Log upload: Consumer → Hub	JSON log (∼few KB)	1	Patient uses the mirror at home, with inference performed locally. Only logs or summary results are uploaded for the dentist to review later.
Professional-offload	Stages 1–2: Consumer Stage 3: Professional Stage 4: Professional → Consumer Log: Consumer → Hub	Pre-processed image (∼200–500 KB)	3	Clinic mirror: a walk-up mirror in a pharmacy or clinic waiting area sends pre-processed images to a nearby dental workstation for quicker chair-side assessment.
Hub-offload	Stage 1: Consumer → Hub Stages 2–3: Hub Stage 4: Hub → Consumer	Raw image (∼2–3 MB)	2	Large clinic setup: several mirrors send images to a central practice server, suitable when multiple patients are assessed across different rooms or nearby clinics.
Tiered-distributed	Stage 1: Consumer Stage 2: Professional Stage 3: Hub Stage 4: Hub → Consumer	Raw image (∼2–3 MB), followed by processed image (∼500 KB), and classification result	3	Chair-side distributed scenario: the Consumer mirror captures the image, the Professional node performs image preprocessing, and the Hub performs classification before returning the result to the Consumer. The Hub could additionally support centralised storage or longitudinal record management where required by the deployment.

Network connectivity is provided via a local Wi-Fi router and a public 5G CPE (Customer Premises Equipment). The physical testbed setup is shown in Figure 4-16.

Experimental Workload: The four-stage application pipeline defined in Section 4.4.1 is used as the workload throughout the physical testbed evaluation. A fixed set of captured input images is reused across trials to control for image variability, allowing computational placement, network condition, and concurrent Consumer demand to be treated as the experimental variables.

Experimental Design: Experiments are conducted across a full factorial design covering all combinations of the following variables:

- **Placement strategy:** four approaches (Consumer-only, Professional-offload, Hub-offload, Tiered-distributed)

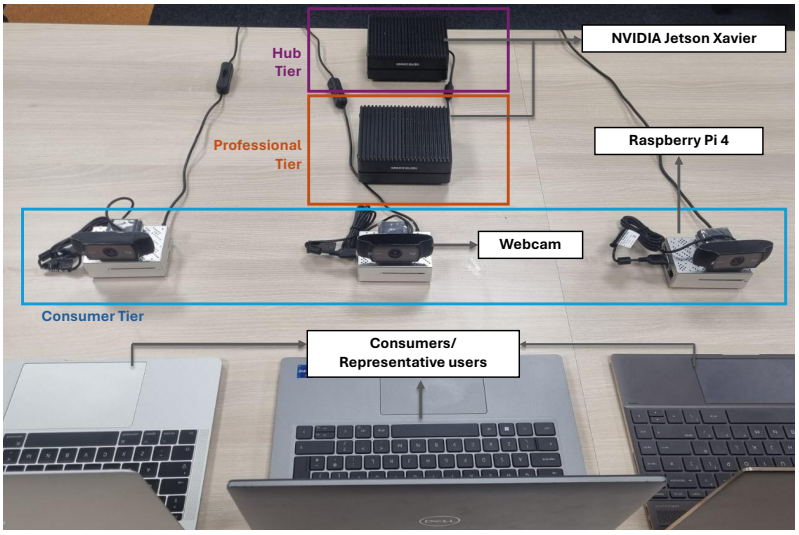


Figure 4-16: Physical IoM testbed configuration showing consumer, professional, and hub nodes

- **Network condition:** Wi-Fi and 5G
- **Concurrent consumers:** 1, 2, and 3 simultaneous consumer nodes.

Each combination is repeated for 25 independent trials, yielding a total of 600 trials (4 strategies × 2 networks × 3 concurrency levels × 25 repetitions).

This is presented in Table 4-2

Table 4-2: Experimental design summary for the physical IoM testbed evaluation.

Experimental Factor	**Evaluated Conditions**
Computational placement	Consumer-only, Professional-offload, Hub-offload, Tiered-distributed
Network technology	Wi-Fi, 5G
Concurrent Consumers	1, 2, 3
Repetitions per configuration	25
Total configurations	24

Metrics: Prior to each experiment, device clocks are synchronised using `timedatectl` with Network Time Protocol (NTP) enabled on all nodes to ensure consistent cross-device timestamping for accurate metric collection. Performance is characterised through five key metrics:

- **End-to-end (E2E) latency** - total elapsed time from image capture to result display at the consumer node, decomposed by pipeline stage.
- **Per-stage latency** - individual processing time for each pipeline stage.
- **Network transfer latency** - per-hop communication time between tiers, capturing the cost of each inter-tier data transfer.

- **Resource utilisation** - combined CPU and GPU utilisation at each active tier node, collected using the `psutil` library; GPU utilisation is recorded on Jetson devices where applicable.
- **Scalability** - degradation in E2E and network latency as concurrent consumer count increases from 1 to 3, including variance metrics (coefficient of variation, percentile analysis).

4.4.3 Results and Discussion

The results presented in this section are structured to progressively characterise the computation-communication trade-off across the four placement strategies, beginning with overall end-to-end latency, decomposing network contributions by hop, examining resource distribution across tiers, and assessing scalability under concurrent load. Together, these perspectives provide the empirical grounding to evaluate where IoM applications sit on the computing continuum and what this means for deployment.

End-to-End Latency Breakdown: Figure 4-17 presents E2E latency decomposed by pipeline stage and network technology. Consumer-only exhibits the highest latency on both Wi-Fi (1,481 ms) and 5G (1,515 ms), with Classification alone accounting for 68% of total E2E time (1,007 ms). This reflects the fundamental constraint of executing CNN inference sequentially on general-purpose CPU cores at the Consumer tier meaning that the processing burden is not a function of network conditions but of local hardware capacity, which is why Wi-Fi and 5G results are nearly identical for this approach.

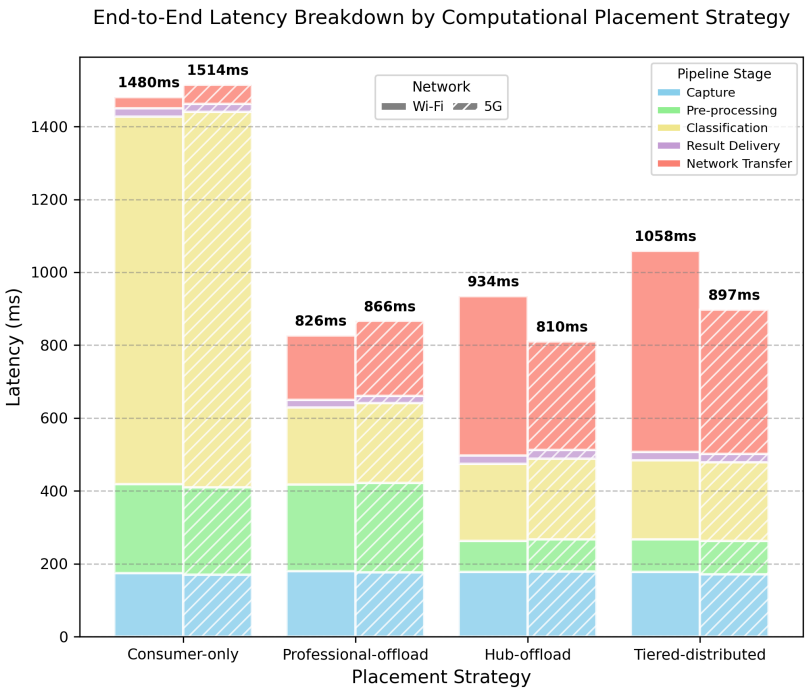


Figure 4-17: End-to-end latency decomposed by pipeline stage and network technology across the four computational placement strategies.

Offloading classification to higher tiers substantially reduces this bottleneck. Professional -offload achieves the lowest Wi-Fi latency (826 ms), reducing Classification time to 214 ms , around a quarter of the end-to-end latency by executing inference on a higher tier. This comes at the cost of 176 ms of network transfer overhead (21% of E2E). On 5G, Hub-offload becomes the fastest approach (810 ms), benefiting from reduced raw image transfer latency compared to Wi-Fi (298 ms vs 438 ms). Tiered-distributed incurs the highest network overhead of all offloading approaches (52% of E2E on Wi-Fi, 44% on 5G), reflecting its three-hop routing structure.

A notable observation is that the relative performance ranking of strategies shifts between network conditions. On Wi-Fi, Professional-offload is the fastest approach, as its smaller pre-processed image payload incurs limited transfer overhead. Hub-offload is penalised on Wi-Fi by the cost of transferring a larger raw image to the hub, making it slower despite its higher processing capacity. On 5G, this penalty is substantially reduced, bringing Hub-offload to the fastest position overall. This ranking inversion has a direct implication for IoM deployment, the optimal placement strategy is not fixed, but depends on the network available. In this context, where only Wi-Fi is accessible, near-edge offloading to a professional node offers the best latency outcome and where 5G is available, centralised hub offloading suits larger, multi-mirror clinic settings where a shared server handles multiple simultaneous users.

These results establish that offloading relieves the classification bottleneck but substitutes it with a network cost that grows with payload size and hop count. The degree to which this substitution is beneficial depends on both the network technology available and the tier to which processing is offloaded. Understanding exactly where this network overhead originates requires examining each transfer hop individually.

Network Transfer Analysis: Figure 4-18 decomposes total network transfer latency by hop direction, revealing the mechanisms behind the Wi-Fi versus 5G patterns observed in Figure 4-17. Consumer-only incurs minimal transfer cost (30 ms Wi-Fi, 52 ms 5G), uploading only a small log file. The slight 5G penalty here is notable: for a payload of just a few KB, the cellular access overhead exceeds any bandwidth benefit, making 5G marginally worse for this approach.

Professional-offload shows moderate total overhead (176 ms Wi-Fi, 205 ms 5G), with the Consumer→Professional hop carrying a pre-processed image of 200-500 KB (117 ms Wi-Fi, 97 ms 5G). The relatively small 5G benefit at this hop reflects the moderate payload size, 5G's bandwidth advantage is present but limited. Hub-offload and Tiered-distributed incur substantially higher costs due to raw JPEG payloads of 2-3 MB at the first hop. The Consumer→Hub hop in Hub-offload costs 406 ms on Wi-Fi but only 242 ms on 5G, a 40% reduction that contributes substantially to Hub-offload's 124 ms reduction in overall E2E latency on 5G. Tiered-distributed's Consumer→Professional hop carries the same raw image (407 ms Wi-Fi, 242 ms 5G), with a further Professional→Hub hop adding 116 ms (Wi-Fi) or 97 ms (5G), yielding

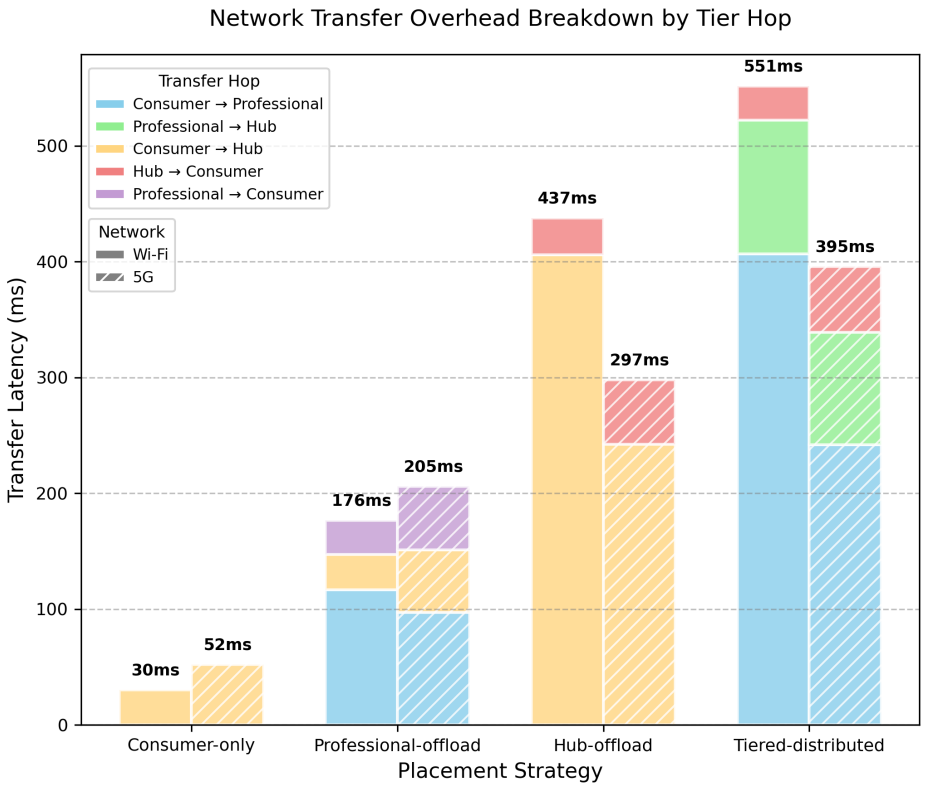


Figure 4-18: Network transfer overhead decomposed by inter-tier hop direction for each placement strategy.

a total network latency of 551 ms on Wi-Fi versus 395 ms on 5G.

The Wi-Fi-to-5G network latency difference therefore scales with payload size and hop count. Consumer-only sees a small penalty (+22 ms), Professional-offload a marginal increase (+29 ms), while Hub-offload and Tiered-distributed realise reductions of −140 ms and −156 ms respectively. For IoM deployments, this finding has a direct implication, 5G's bandwidth advantage only becomes meaningful when there is sufficient data volume to exploit it. For small payloads, the cellular access overhead dominates, whereas for large raw image transfers across multiple hops, 5G delivers a compounding benefit at each hop. For the IoM, this establishes that the value of 5G connectivity is not uniform across deployment scenarios but is specifically significant for hub-centric or fully distributed strategies, and largely irrelevant for consumer-local or near-edge approaches. This should directly inform infrastructure decisions, utilising 5G makes sense where the intended placement strategy involves large inter-tier transfers, but offers little return where processing remains close to the consumer.

Resource Utilisation: Fig. 4-19 presents resource utilisation across IoM tiers as a heatmap, showing how each placement strategy distributes the computational burden of the use case pipeline. Consumer-only concentrates 68.8% utilisation at the Consumer tier, with the Professional tier unused and the Hub at 7.2% for log storage only. Professional-offload reduces Consumer utilisation to 39.0% while shifting the principal processing demand to the Professional tier, which reaches 61.6%. Hub-offload simi-

larly concentrates the principal processing demand at the Hub, which reaches 68.3%, while Consumer utilisation falls to 22.1%. Tiered-distributed engages all three tiers, producing utilisation levels of 24.3% at the Consumer, 37.3% at the Professional, and 60.1% at the Hub. These percentages represent the utilisation measured independently at each tier rather than proportions of a fixed total workload, and therefore are not expected to sum to 100% or to be equal across tiers.

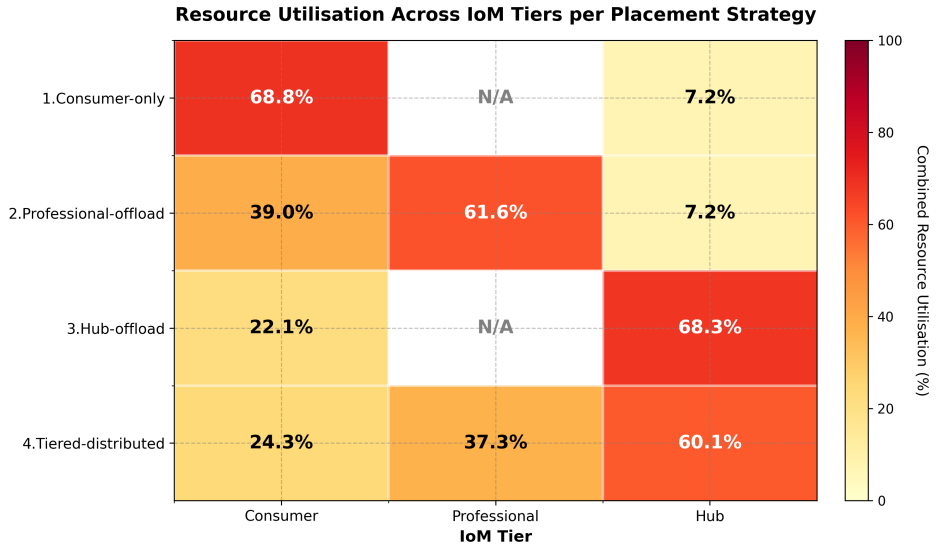


Figure 4-19: Combined Resource utilisation (%) across IoM tiers per placement strategy.

The observed pattern is directly explained by where the principal processing stages are executed in each strategy. In Consumer-only, the Raspberry Pi CPU carries the full inference load, producing the highest Consumer utilisation and leaving the higher tiers largely idle. In Professional-offload, classification is handled by the GPU-capable Jetson Xavier at the Professional tier, while in Hub-offload both preprocessing and classification are transferred to the Hub, accounting for the concentration of utilisation at the corresponding higher-capability node. Tiered-distributed is the only strategy in which application processing is explicitly distributed across all three tiers, with image capture at the Consumer, preprocessing at the Professional, and classification and result delivery at the Hub. This produces computational activity across the complete hierarchy, although not equal utilisation across the tiers. The higher Hub utilisation of 60.1% reflects the computational demand of the stages assigned to that resource, while the lower Consumer and Professional values correspond to their respective assigned tasks and hardware capabilities.

These results show that Tiered-distributed should be interpreted as a distributed execution strategy rather than an equal-load-balancing strategy. The utilisation values are measured independently at each tier and do not represent proportions of a fixed total workload. Achieving a more even utilisation profile would require load-aware reassignment of application stages according to current resource conditions rather than the fixed stage-to-tier mappings evaluated here. This motivates the move towards context-aware task distribution considered in Chapter 5. More generally, computa-

tional placement affects not only end-to-end latency but also where resource demand, queuing, and contention arise within the IoM system.

Scalability Under Concurrent Load: Having established how load distributes across the hierarchy, these strategies are examined to see how they react as concurrent demand increases. Figure 4-20 presents a four-panel analysis separating compute and network latency across concurrency levels, with shaded bands indicating mean ± 1 SD. The compute panels show nearly identical slopes and band widths between Wi-Fi and 5G, confirming that compute behaviour is network-independent. This validates that the E2E differences between network types observed in Section 4.4.3 are attributable to network transfer alone, and provides a clean baseline for interpreting the network panels below.

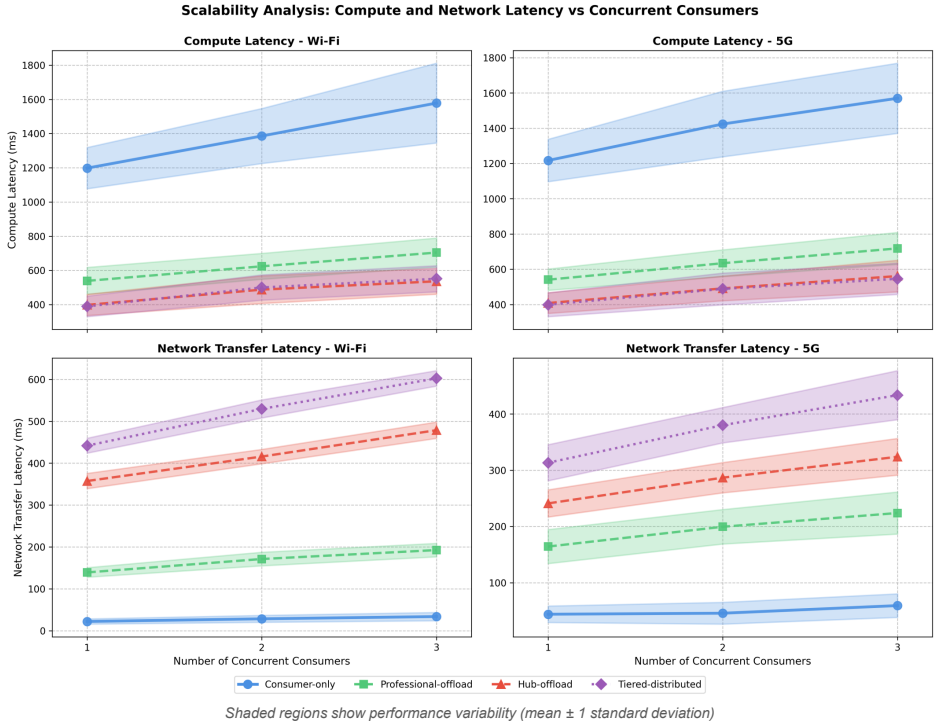


Figure 4-20: Scalability analysis under concurrent consumer load.

Within the compute panels, Consumer-only exhibits the steepest degradation, rising from 1,198 ms at one Consumer to 1,578 ms at three, an average increase of 190 ms per additional Consumer. Notably, its shaded variance band widens substantially, with SD growing from 121 ms to 233 ms (93%). This reflects increasing queuing and resource contention at the Consumer tier as concurrent inference demand increases, producing not only higher mean latency but greater performance variability. The three offloading strategies, by contrast, remain close together in the compute panels, with flatter slopes and narrower bands. Professional-offload degrades at only 82 ms per additional Consumer, with the coefficient of variation decreasing as the higher-capability tiers accommodate concurrent classification requests with less contention.

The network panels (bottom row) show a clear separation between strategies. Consumer-only remains flat and low across all concurrency levels, consistent with its minimal transfer activity. Hub-offload and Tiered-distributed exhibit the steepest gradients on Wi-Fi, as increasing concurrent requests amplify inter-tier transfer congestion. On 5G, both strategies show shallower gradients, demonstrating that 5G limits how quickly congestion accumulates under concurrent load. However, this comes with a trade-off, the shaded variance bands are visibly wider on 5G than Wi-Fi for these strategies, indicating that while 5G reduces mean latency growth, it introduces greater trial-to-trial variability. This likely reflects the less deterministic nature of cellular network conditions compared to a controlled local Wi-Fi environment.

Deployment Implications: Across the four subsections, a consistent picture emerges of how the computation-communication trade-off manifests across different IoM deployment contexts. Table 4-1 maps each strategy to a scenario, and the results validate and refine those mappings empirically.

Consumer-only keeps all processing local, minimising network dependency and making it suitable for private, localised deployments such as a home smart mirror. However, its high consumer tier utilisation and steep latency degradation under concurrent load make it unsuitable for any multi-user scenario. Its growing variance under load is particularly problematic for the IoM, where user experience consistency is as important as mean latency.

Professional-offload emerges as a balanced strategy. It achieves the lowest Wi-Fi latency, a stable variance under load, and relieves the consumer node sufficiently for responsive concurrent operation. It is the recommended strategy for clinic or pharmacy settings where a professional node is co-located and Wi-Fi is the available network.

Hub-offload produced the lowest latency under the evaluated 5G condition, where its raw image transfer cost is substantially reduced. It is well-suited to large multi-room clinic deployments with reliable high-bandwidth connectivity, where a centralised server can handle simultaneous requests from multiple mirrors without requiring a professional node at every point of care.

Tiered-distributed exercises the complete IoM hierarchy by assigning image capture to the Consumer, preprocessing to the Professional, and classification and result delivery to the Hub. This distributes the application workflow across all three tiers rather than concentrating processing at a single location. However, it also incurs the highest cumulative network overhead because both the raw captured image and subsequently processed image must traverse tier boundaries before the result is returned to the Consumer. The strategy therefore represents a deployment in which the capabilities of all three tiers are available and application stages can be distributed between them, although its suitability depends strongly on the communication environment and the additional transfer overhead introduced by this distribution.

Together, these findings demonstrate that no single placement strategy is universally optimal for the IoM. Each strategy represents a different position on the computation-communication trade-off: Consumer-only minimises network dependency but is constrained by Consumer hardware capacity and experiences increasing queuing and resource contention under load; Professional-offload and Hub-offload reduce the computational burden at the Consumer tier but introduce network overhead that grows with payload size and concurrency; and Tiered-distributed partitions application processing across the complete Consumer-Professional-Hub hierarchy, but incurs the highest cumulative transfer overhead. The optimal choice shifts depending on the network available, the proximity and availability of higher-tier nodes, the computational intensity of the application, and the number of concurrent users, all of which vary across real IoM deployments. This context-dependence is itself a key finding of this work, indicating that a fixed placement policy is unlikely to accommodate the range of operating conditions represented within the IoM and motivating intelligent, dynamic task placement responsive to application requirements and live ecosystem conditions.

4.5 Summary

This chapter addressed Q_2 and O_2 by evaluating the IoM from an application-level proof-of-concept to a physical multi-tier distributed system. The first stage developed an automated dental smile-analysis application as a representative assistive IoM use case. A dedicated dataset combining real and AI-generated smile imagery was constructed to support model development, and the resulting gingival-display CNN achieved a mean test accuracy of 81.607% across ten independent training trials. The model was subsequently integrated with a smart mirror interface to demonstrate an end-to-end user-facing application. This established both the feasibility of the application and a representative computational workload for the subsequent system-level evaluations.

The second stage investigated how the performance of this workload changes when computation is placed between local edge and remote cloud resources. The application was adapted for distributed execution, with the classification model containerised to support remote inference through Google Cloud Run. The edge configuration achieved a lower mean end-to-end latency than the cloud configuration under the evaluated conditions, although decomposition of the processing pipeline showed that image acquisition and preprocessing contributed substantially more latency than model inference itself. The cloud implementation additionally introduced communication delay and greater performance variability. These findings demonstrated that computational placement affects the complete processing and communication path of an IoM service.

Building on this initial comparison, the final stage extended the evaluation to a physical implementation of the Consumer, Professional, and Hub hierarchy. Four com-

putational placement strategies were evaluated under Wi-Fi and 5G connectivity and increasing concurrent demand, considering end-to-end latency, inter-tier network transfer, resource utilisation, and scalability. The results showed that offloading computationally demanding stages can reduce the processing burden at the Consumer tier, but introduces communication overhead that varies with payload size, network technology, and the number of transfers between tiers. The relative performance of the evaluated strategies also changed with the operating conditions, with Professional-offload achieving the lowest latency under Wi-Fi and Hub-offload under 5G. Increasing concurrent demand further demonstrated that placement affects both processing and communication behaviour across the hierarchy.

Collectively, these evaluations establish that computational placement within the IoM represents a coupled computation-communication problem rather than a fixed architectural decision. The suitability of an execution location depends on the computational requirements of the application, the capabilities and availability of candidate resources, the communication environment, and the current system demand. These factors may vary across users, applications, devices, and deployment environments, making context an important consideration within the IoM, particularly where services must accommodate diverse users and changing operating conditions. The chapter therefore provides the empirical foundation for moving from static placement strategies towards context-aware task distribution, in which execution decisions can respond to both application requirements and the conditions of the surrounding IoM ecosystem. This is investigated in the following Chapter 5.

Chapter 5

Context-Aware Task Distribution (CATD) for IoM Systems

5.1 Introduction

Chapter 4 established that computational placement within the IoM cannot be treated as a fixed architectural decision. The physical testbed evaluation demonstrated that the relative performance of different placement strategies changes with the available network, processing capability, communication path, and concurrent demand. These findings showed that distributing an IoM service involves a coupled computation-communication problem in which the suitability of an execution resource depends on the conditions under which the service is operating. As the IoM expands beyond a single representative workload, this decision becomes increasingly complex because different applications and users introduce heterogeneous computational, communication, latency, and data-handling requirements.

This challenge reflects a broader characteristic of distributed IoT environments, where increasing device heterogeneity and interconnectivity create systems containing resources with substantially different processing capabilities, network conditions, and availability [1]. Similar distribution problems arise in smart healthcare, industrial IoT, and smart-city systems, where tasks must be coordinated across combinations of resource-constrained devices, edge resources, and higher-capability infrastructure while satisfying application-specific performance requirements [209; 210]. In these environments, assigning a task according to computational capability alone may overlook communication cost, while prioritising network proximity alone may select a resource that cannot efficiently satisfy the computational requirements of the workload.

Task distribution and offloading mechanisms have consequently evolved from simple allocation strategies towards approaches that consider multiple dimensions of system state. Traditional methods such as Round Robin, proximity-based selection, and greedy resource allocation provide computationally efficient placement decisions but generally optimise a limited aspect of the allocation problem [160; 161; 211]. More recent ap-

proaches incorporate multiple criteria, including resource availability, network conditions, task dependencies, load state, and security requirements [162; 212; 213; 176]. As reviewed in Chapter 3, however, the relevance and combination of these factors depend on the characteristics of the ecosystem in which task distribution is performed. Existing work therefore provides the methodological basis for context-aware allocation, but does not directly address the stakeholder-oriented hierarchy and range of task and user requirements introduced by the IoM. Table 5-1 positions CATD against the principal task-distribution and dynamic scheduling approaches reviewed in Chapter 3. The comparison is methodological rather than performance-based, since these approaches are evaluated under different architectures, workloads, and optimisation objectives.

Table 5-1: Methodological comparison of CATD with representative task-distribution and scheduling approaches.

Approach	**Primary Decision Context**	**Adaptive**	**Relationship to CATD**
Simple and load-aware allocation [160; 161]	Availability, allocation order, resource utilisation, and system load	No / Limited	Provides simple and interpretable allocation, but considers a narrower range of task and system context than CATD.
Multi-criteria and context-aware scheduling [162; 163; 164; 165]	Multiple resource, network, cost, performance, task, and operating-context factors	Limited	Closest methodological basis to CATD; CATD structures the selected factors around IoM task requirements and its stakeholder-oriented tier hierarchy.
Hierarchical offloading [166; 167]	Resource characteristics and allocation across multiple infrastructure tiers	Varies	Shares hierarchical resource organisation, while CATD additionally evaluates residual capability, network quality, proximity, and data sensitivity for each task.
Learning-based dynamic scheduling [168; 169; 170]	Dynamic resource, network, queue, priority, and workload state	Yes	Learns allocation behaviour through interaction with the environment, whereas CATD uses an explicit deterministic filtering and scoring process without a training phase.
Application- and data-aware optimisation [171; 172]	Application characteristics, data requirements, reliability, and multiple optimisation objectives	Varies	Extends scheduling beyond infrastructure state; CATD similarly incorporates task-specific requirements but tailors them to IoM locality, sensitivity, and tier feasibility.

Within the IoM, this problem is amplified by the combination of heterogeneous Consumer, Professional, and Hub resources with applications spanning lightweight monitoring, interactive user-facing services, and computationally demanding analytics. The context relevant to an allocation decision may include the computational and memory requirements of the task, the residual capability of available nodes, current network

quality, geographic proximity, and the sensitivity of the data being processed. These factors may vary between applications and users and may change as devices become loaded or unavailable, network conditions fluctuate, or the composition of active workloads changes. A task-distribution mechanism for the IoM must therefore move beyond selecting a generally preferred execution tier and instead determine which available node provides an appropriate match for the current task under the prevailing ecosystem conditions.

This chapter addresses Q_3 and O_3 through the development of the CATD framework reported in J4, which coordinates tasks across the Consumer, Professional, and Hub hierarchy established in Chapter 2. CATD is designed as a deterministic and interpretable per-task decision mechanism in which the factors contributing to each allocation decision correspond directly to requirements of the IoM system. Tier-based filtering first constrains candidate resources according to task feasibility, after which residual computational capability, network quality, geographic proximity, and data sensitivity are considered to determine an appropriate execution resource under the current system conditions.

The novelty of CATD therefore does not lie in the individual concepts of resource filtering, load-aware allocation, network-aware scheduling, or proximity-based selection, but in combining and prioritising them within a unified framework tailored to the heterogeneous tasks, users, and stakeholder-oriented tier hierarchy of the IoM. This explicit formulation was selected to retain transparency in how individual task-placement decisions are reached while allowing the relevant system factors to be evaluated jointly rather than in isolation. Although developed for the IoM, the underlying design principles may also be transferable to other user-centred distributed systems in which heterogeneous tasks require different combinations of computational, communication, locality, and data-handling requirements.

The framework is evaluated through a discrete-event simulation of a representative multi-tier IoM deployment containing heterogeneous computational resources, network links, and application workloads. Five representative IoM task categories are used to exercise different computational, memory, bandwidth, latency, and data-sensitivity requirements. CATD is compared with Round Robin, Random, Greedy Resource, and Nearest Node allocation across eight tests covering allocation behaviour, workload scaling, changing workload composition, node and network degradation, dynamic operating conditions, and data locality. The chapter first defines the computational and task model used to instantiate the IoM hierarchy, before presenting the CATD scoring and distribution mechanisms, simulation methodology, and resulting evaluation.

5.2 System Architecture and Model

The CATD system model instantiates the Consumer, Professional, and Hub hierarchy established in Section 2.2.2 as a distributed computational environment for task allocation. The purpose of this section is to specify the node, network, and task characteristics required by the CATD formulation and subsequent simulation evaluation. The network topology and tier capabilities are first defined before introducing the representative task profiles and contextual requirements used by the framework.

5.2.1 IoM Network Architecture

The considered IoM network architecture represents a distributed computing ecosystem designed to serve health and beauty applications across diverse deployment contexts ranging from high-performance medical facilities to residential consumer environments. The architecture models a representative urban deployment spanning a 5×5 km area, a scale consistent with metropolitan IoT ecosystem studies [214].

The designed network implements a three-tier hierarchical mesh topology that reflects real-world deployment constraints while enabling efficient task processing. The tiers comprise of Hub, Professional, and Consumer nodes (detailed specifications in Section 5.2.2). Hub nodes represent high-capability computing infrastructure for complex analytics. Professional nodes provide medium-capability processing for interactive applications. Consumer nodes offer basic computational resources for routine tasks and user interaction. This hierarchical design serves multiple purposes: (1) it mirrors the natural clustering of computational resources in urban environments, (2) it enables node to node efficient data routing, and (3) it supports data locality requirements critical for privacy sensitive applications.

Fig. 5-1 illustrates the relationship between the IoM's conceptual layers and the physical node hierarchy. The User/Perception layer interfaces through user interactions; the Network layer encompasses the three-tier topology enabling data routing and communication; and the Application layer, where the CATD algorithm operates, makes allocation decisions based on the holistic view of user requirements, network conditions, and available computational resources across all tiers.

Although nodes naturally cluster in geographic proximity and may share network infrastructure and security protocols, the architecture was designed to enable any node to allocate tasks to any other node in the network. This design reflects the interconnected nature of the IoM ecosystem while supporting both local processing preferences and global resource optimisation when necessary. The resulting spatial deployment and cross-tier interactions between Hub, Professional, and Consumer nodes in an envisioned urban area are illustrated in Fig. 5-2.

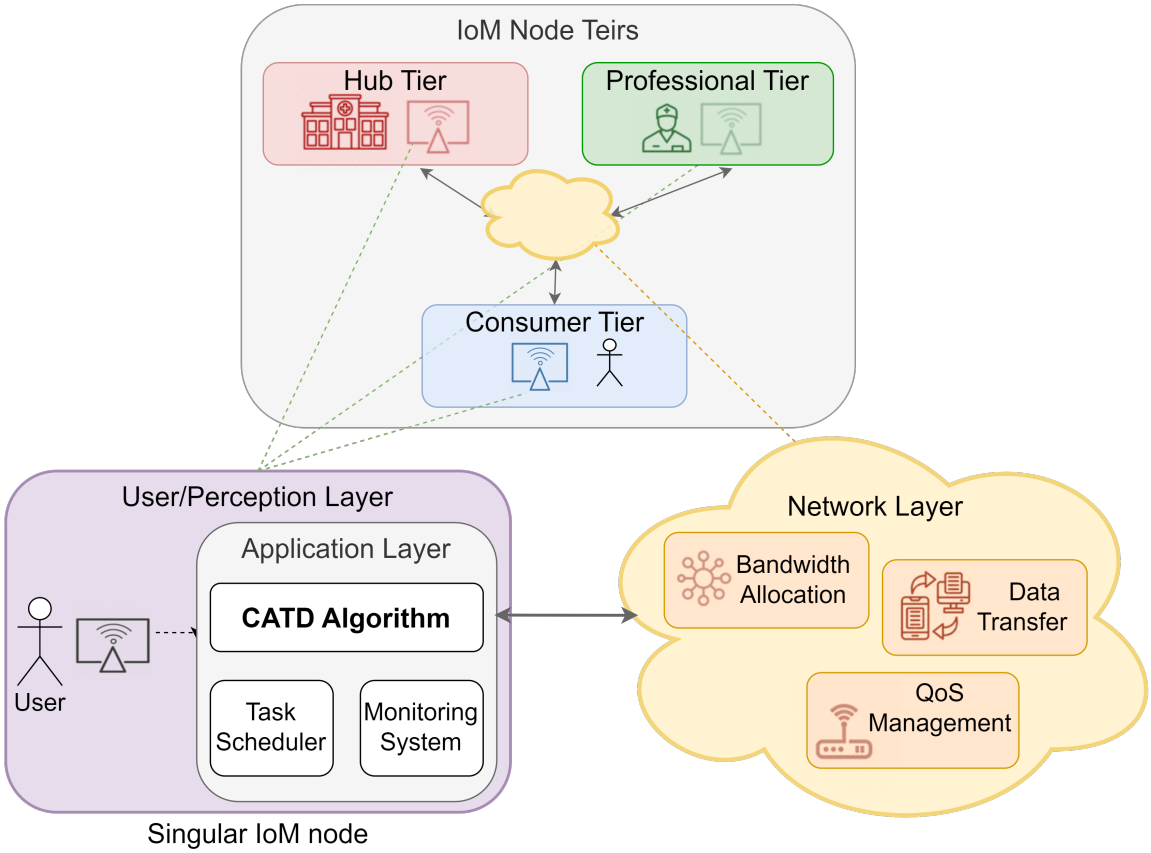


Figure 5-1: IoM system architecture showing the integration of its conceptual layers with the three-tier physical node hierarchy.

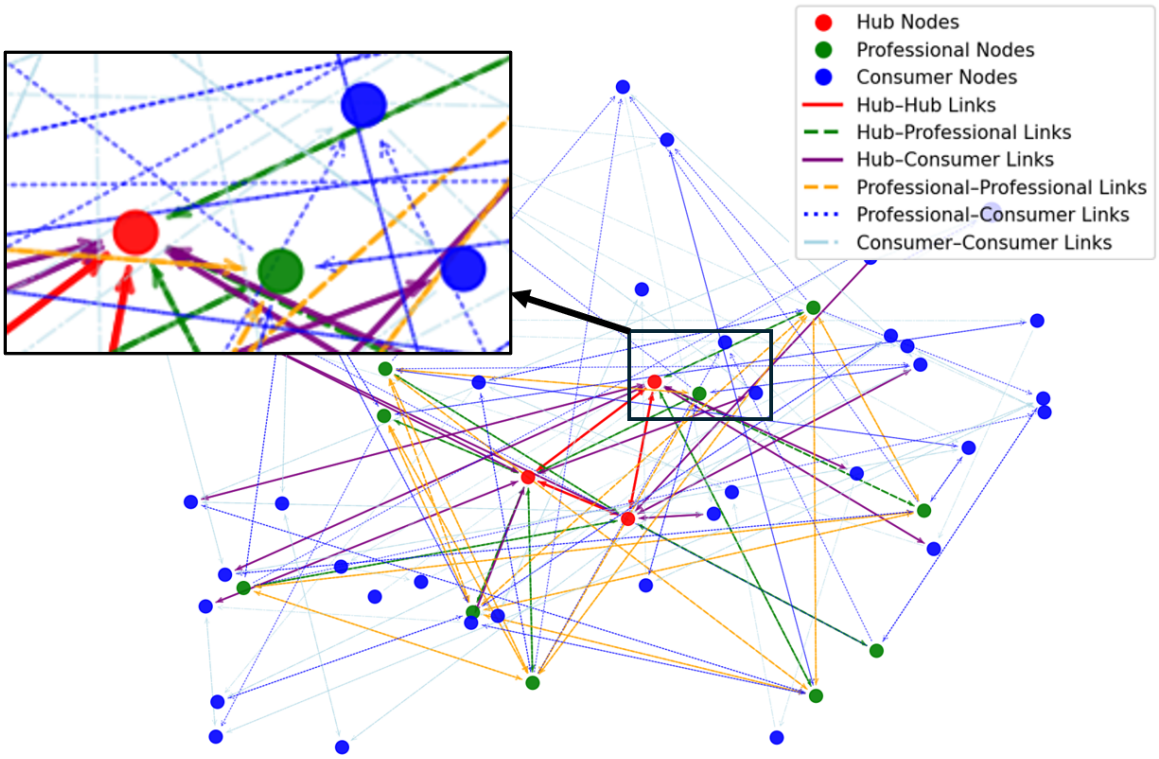


Figure 5-2: IoM topology showing the spatial distribution of Hub, Professional, and Consumer nodes and the possible cross-tier interactions.

5.2.2 Node-Tier Capability Model

The CATD framework adopts the Consumer, Professional, and Hub hierarchy established in Section 2.2.2 and translates the stakeholder-oriented architecture into computational resource classes for task distribution. The purpose of the present model is therefore not to redefine the roles of the three tiers, but to specify the relative capabilities and resource characteristics used by CATD when evaluating candidate execution nodes. The tier specifications are informed by representative enterprise computing infrastructure [215; 216], professional edge-computing platforms [151], and consumer embedded systems [217; 218].

The three tiers are modelled as follows:

- **Hub Nodes (H):** Hub nodes represent the highest-capability computational resources within the IoM hierarchy, comparable to enterprise cloud infrastructure and data-centre computing platforms [215; 216; 219]. They are intended to support computationally intensive tasks such as complex health analytics, large-scale data processing, and resource-intensive AI inference. Within the simulation, Hub nodes provide computational capacities in the range of 800-1200 Gigafloating-point operations per second (GFLOPS). In an IoM deployment, these resources could correspond to infrastructure operated by hospitals, large clinics, retailers, or service providers supporting centralised analytical services.

- **Professional Nodes (P):** Professional nodes provide intermediate computational capability comparable to professional workstations and edge-computing platforms [151; 150]. They support tasks whose requirements exceed typical Consumer resources but do not necessarily require Hub-level processing, including interactive skin analysis, augmented-reality beauty services, and other moderately demanding applications. Professional nodes are modelled with computational capacities of 150-350 GFLOPS and may represent resources located within professional environments such as dermatology clinics, beauty salons, pharmacies, or retail settings.

- **Consumer Nodes (C):** Consumer nodes represent the most resource-constrained tier and reflect the capabilities of consumer embedded systems and smart mirror devices [57; 217; 218]. These nodes primarily support user interaction, lightweight image processing, local data handling, and communication with higher tiers when additional computational capability is required. Consumer nodes are modelled with computational capacities of 40-100 GFLOPS and represent personal or small-scale IoM deployments such as residential smart mirrors and consumer-facing kiosks.

The computational capacities assigned to the three tiers are representative simulation ranges rather than categorical task thresholds. Ranges are used instead of fixed values

to introduce intra-tier heterogeneity while preserving the intended ordering of computational capability from Consumer to Professional to Hub resources. They are informed by the relative capabilities of consumer embedded systems, professional edge-computing platforms, and higher-capability enterprise infrastructure and are intentionally separated to represent the heterogeneity expected across the IoM hierarchy. GFLOPS is used as the common measure of computational capability.

The use of capability ranges rather than fixed values introduces heterogeneity within each tier, allowing individual nodes of the same type to differ in their available computational capacity. This is important for CATD because tier membership provides only an initial indication of task suitability. The final allocation decision additionally depends on the current residual capacity of the node, network conditions, geographic proximity, and task-specific requirements as defined in Section 5.3.1. The hierarchy therefore provides the structural basis for the tier-based filtering performed during the resource-assessment phase of CATD while retaining flexibility for context-aware selection between individual nodes.

Table 5-2: Task profile categories and characteristics.

Task Category	**Compute Demand**	**Memory Demand**	**Bandwidth Demand**	**Data Sensitivity**	**Latency Requirement**
Health Monitoring	Low	Low	Medium	High	Stringent
Health Analytics	Med-High	High	High	Medium	Tolerant
AR Beauty	Medium	Medium	High	Low	Stringent
Product Recommendations	Low	Medium	Medium	Low	Moderate
System Management	Low	Low	Low	Medium	Variable

5.2.3 Task Characterisation and Profiles

Task characterisation within the CATD framework captures the heterogeneous requirements of IoM applications across multiple dimensions. Five representative task categories are defined, which are informed by application domains in health monitoring [220], medical image analysis [221], augmented reality systems [222; 124], interactive applications [223], and advanced skin analytics [22]. These categories span the IoM application space from lightweight edge processing to resource-intensive analytical workloads.

Each task profile is characterised along five dimensions, with terminology aligned to Table 5-2:

- **Compute Demand** specifies the processing capacity required for task execution, ranging from lightweight sensor data processing to computationally intensive medical image analysis and deep learning inference [221; 224].

- **Memory Demand** captures the working memory needed during task execution, varying with algorithm complexity and data structure requirements.
- **Bandwidth Demand** reflects data transfer requirements based on payload size and streaming needs. Interactive applications like AR beauty consultations demand high bandwidth for video streaming [222], while batch analytics operate with lower bandwidth requirements.
- **Data Sensitivity** classifies the privacy implications of processed data. Health monitoring and diagnostic tasks involve highly sensitive medical information that is prioritised for processing within secure local domains, in this context this would be node clusters sharing common network boundaries, unified security policies, and trusted communication paths [220].
- **Latency Requirement** captures timing constraints, from stringent real-time demands for interactive applications to tolerant thresholds for background analytics.

To enable systematic comparison across task types and node tiers, a qualitative scale is adopted where descriptors are defined relative to the node tier capabilities established in Section 5.2.2:

- **Low**: Requirements satisfiable by Consumer-tier nodes (e.g., compute <100 GFLOPS, memory <16 GB)
- **Medium**: Requirements satisfiable by Professional-tier nodes (e.g., compute 100–350 GFLOPS, memory 16–64 GB)
- **High**: Requirements necessitating Hub-tier nodes (e.g., compute >350 GFLOPS, memory >64 GB)

The qualitative task-demand categories used below are defined relative to the computational capabilities of the node tiers established in Section 5.2.2. They describe the minimum level of computational capability generally required by a task rather than reproducing the complete capacity range of each tier. Accordingly, tasks below approximately 100 GFLOPS can generally be accommodated by Consumer resources, tasks between approximately 100 and 350 GFLOPS require Professional-level capability or higher, and tasks exceeding approximately 350 GFLOPS exceed the modelled Professional-tier maximum and therefore require Hub-level capability. The larger Hub capacity range of 800-1200 GFLOPS provides additional computational headroom for demanding workloads rather than defining the lower boundary of the "High" task category.

Compound descriptors (e.g., "Low-Med", "Med-High") indicate requirements spanning adjacent tiers, enabling flexible allocation based on node availability. Table 5-2 sum-

marises the characteristics of each task type using this scale, enabling systematic evaluation of the CATD framework's allocation decisions across diverse application requirements.

5.3 CATD Framework Design

This section presents the CATD framework, comprising two components: the scoring system that quantifies task-node compatibility (Section 5.3.1), and the distribution algorithm that applies these scores within a hierarchical allocation workflow (Section 5.3.2). The framework addresses task distribution across heterogeneous nodes by systematically evaluating resource availability, network conditions, data sensitivity, and physical proximity through a three-phase workflow: resource assessment for tier-based filtering, node selection and scoring, and decision handoff for final allocation and failure handling.

5.3.1 Scoring System Design

The CATD framework employs a multi-factor scoring system that evaluates task-node compatibility through three primary scoring mechanisms. Each mechanism addresses specific aspects of the allocation decision while maintaining computational efficiency suitable for real-time operation.

In what follows, scoring equations are derived that integrate established approaches from resource assessment, network quality evaluation, and path-based proximity optimisation into a unified framework tailored to the hierarchical IoM architecture. These equations determine task placement by evaluating node capabilities, network conditions, and physical constraints:

1. **Resource Capability Score:** This score evaluates a node's ability to handle a task based on available capacity and current load:

$$R(n,t) = \min_{r \in \{c,m\}} \frac{Cap_r(n) \cdot \big(1 - Load(n)\big)}{RC_r(t)}, \tag{5-1}$$

where r indexes the resource dimensions compute (c) and memory (m), the numerator is the node's residual capacity in dimension r, and $RC_r(t)$ is the task's requirement in that dimension. The minimum is taken over the binding dimension, so $R(n,t) \geq 1$ holds only when residual capacity meets the task requirement in both compute and memory.

2. **Network Quality Score:** This score evaluates network suitability based on bandwidth, latency, and reliability. For a candidate node n, the relevant network path $w(t,n)$ is the minimum-latency path from the task source $Src(t)$ to n. The path-level metrics are aggregated from its constituent links, end-to-end latency is the sum of link latencies, available bandwidth is set by the bottleneck (minimum-bandwidth) link, and

reliability is the product of link reliabilities:

$$NQ(w,t) = Rel(w) \times \min\left(\frac{B(w)}{PC(t).b}, \frac{PC(t).l}{L(w)}\right), \tag{5-2}$$

where $NQ(w,t)$ is the network quality score for task t over network path w; $Rel(w)$ is the end-to-end reliability of path w; $B(w)$ is the available bandwidth of the path, determined by its bottleneck link; $PC(t).b$ is the bandwidth requirement of task t; $PC(t).l$ is the maximum latency requirement of task t; and $L(w)$ is the end-to-end latency of path w. The ratio $\frac{B(w)}{PC(t).b}$ therefore represents bandwidth suitability, while $\frac{PC(t).l}{L(w)}$ represents latency suitability.

The network is represented as a weighted graph in which each communication link is assigned its current simulated latency as the edge weight. For a task originating at $Src(t)$ and a candidate node n, the selected path is therefore

$$w(t,n) = \arg\min_{w \in \mathcal{P}(Src(t),n)} \sum_{e \in w} L_e, \tag{5-3}$$

where $\mathcal{P}(Src(t),n)$ is the set of available paths between the task source and candidate node, and L_e is the latency associated with link e. The end-to-end path latency is consequently $L(w) = \sum_{e \in w} L_e$. Once the path is selected, its available bandwidth is determined by the minimum-bandwidth link along the path, while path reliability is obtained from the product of the constituent link reliabilities.

3. **Node Assignment Score:** This score combines resource and network capabilities with distance considerations,

$$Score(t,n,w) = \min\big(R(n,t), NQ(w,t)\big) \times \frac{1}{1 + \dfrac{d(Src(t),n)}{local}}, \tag{5-4}$$

The minimum operator in $\min\big(R(n,t), NQ(w,t)\big)$ treats resource and network suitability as jointly necessary rather than mutually compensatory. A candidate node with strong computational capability but poor network quality, or strong network conditions but insufficient residual resources, is therefore limited by its weaker dimension. This reflects the coupled computation-communication dependency established experimentally in Chapter 4. Therefore, the formulation is intentionally conservative, since a single weak dimension can dominate the combined suitability score rather than being compensated for by a high value in the other dimension.

The term $d(Src(t),n)$ is the distance from the task source to node n. The proximity term is largest, equal to 1, when the task runs on a co-located node and decreases gradually with distance, so that nearby nodes are favoured without allowing distance to override the resource and network suitability of the candidate node.

5.3.2 CATD Algorithm

The CATD algorithm optimises task placement across the IoM hierarchy while considering computational resources, network conditions, and data sensitivity requirements. The algorithm operates through the three phases listed below, detailed in Algorithm 1. The complete three-phase CATD workflow is illustrated in Fig. 5-3. The flowchart shows the progression from initial tier selection through task–node compatibility assessment and node scoring to final allocation, together with the tier-widening and retry procedure used when no suitable node is initially available.

- In **Phase 1 (Resource Assessment)**, the algorithm evaluates the task's compute requirement $RC_c(t)$ against the tier thresholds (τ_H, τ_P) to determine the initial candidate node set, directing demanding tasks to higher tiers while allowing lightweight tasks to utilise any available tier.

- In **Phase 2 (Node Selection & Scoring)**, candidate nodes satisfying the residual feasibility condition ($R(n,t) \geq 1$, Eq. 5-1) are evaluated using the assignment score (Eq. 5-4). Here, $SC(t)$ denotes the data-sensitivity classification of task t, encoded as 1 for Low, 2 for Medium, and 3 for High sensitivity. For highly sensitive tasks ($SC(t) = 3$), nodes within the local proximity threshold receive the sensitivity multiplier. Because node load enters both the feasibility condition and the score, allocation responds dynamically to runtime conditions: as nodes become loaded or fail, subsequent allocations shift towards nodes with available residual capacity.

- In **Phase 3 (Decision Handoff)**, the highest-scoring node is selected and the task registry updated. If no feasible node is found, the tier constraint is relaxed: each retry widens the candidate set to the next tier ($H \rightarrow H \cup P \rightarrow N$), up to $\kappa_{\max}$ attempts, so demanding tasks can be served by lower-tier nodes with sufficient capacity when their preferred tier is saturated.

Algorithm 1 Context-Aware Task Distribution Algorithm

Require: t: Task, H, P, C: Node sets, W: Network links, $\kappa_{\max}$: Maximum retries
Ensure: Assignment result and task status update

Initialize: $BestNode \leftarrow$ null, $BestScore \leftarrow 0$, $\kappa \leftarrow 0$ ▷ **Phase 1: Task Resource Assessment**
if $RC_c(t) \geq \tau_H$ **then**
    $NodeSet \leftarrow H$ ▷ High-resource tasks target H nodes
else if $RC_c(t) \geq \tau_P$ **then**
    $NodeSet \leftarrow H \cup P$ ▷ Medium tasks consider H and P nodes
else
    $NodeSet \leftarrow N$ ▷ Light tasks can use any tier
end if
▷ **Phase 2: Node Selection & Scoring**
while $\kappa < \kappa_{\max}$ **do**
    for each $n \in NodeSet$ **do**
        if $R(n,t) \geq 1$ **then** ▷ Residual feasibility, Eq. 5-1
            $w \leftarrow$ minimum-latency path from $Src(t)$ to n
            if $SC(t) = 3$ **and** $d(Src(t), n) \leq local$ **then**
                $score \leftarrow Score(t, n, w) \times sensitive$ ▷ High-sensitivity task within local threshold
            else
                $score \leftarrow Score(t, n, w)$
            end if
            if $score > BestScore$ **then**
                $BestScore \leftarrow score$
                $BestNode \leftarrow n$
            end if
        end if
    end for
    ▷ **Phase 3: Decision & Handoff**
    if $BestNode \neq$ null **then**
        Assign task t to node $BestNode$
        Update IoM task registry with assignment status
        return $BestNode$ ▷ Successful allocation
    end if
    ▷ Relax tier constraint, widen candidate set
    if $NodeSet = H$ **then**
        $NodeSet \leftarrow H \cup P$
    else if $NodeSet = H \cup P$ **then**
        $NodeSet \leftarrow N$
    end if
    $\kappa \leftarrow \kappa + 1$ ▷ Retry over widened set
end while
▷ **Allocation Failure Handling**
if $\kappa \geq \kappa_{\max}$ **then**
    Add task t to delayed retry queue with exponential backoff
    Update IoM task registry with failure status
    return null ▷ No suitable node found
end if

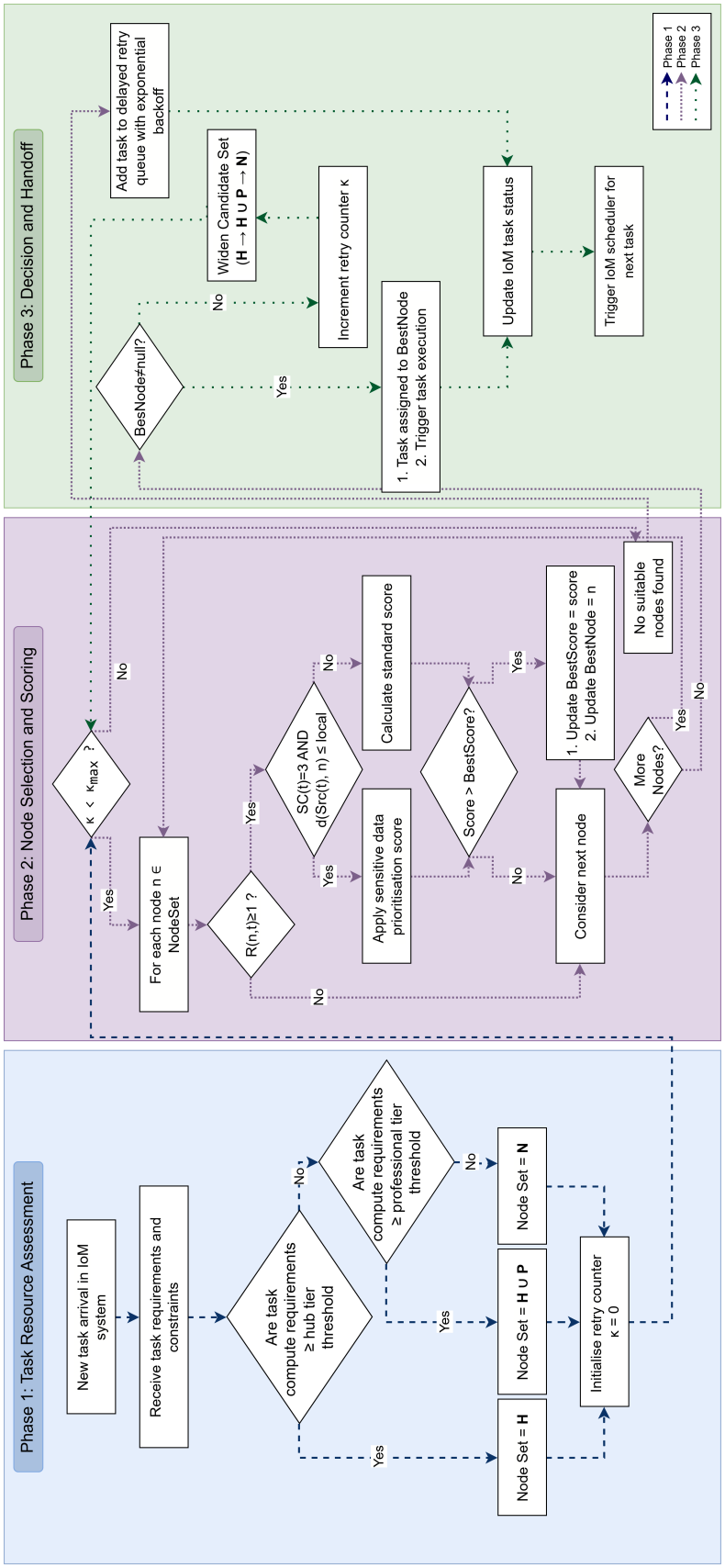


Figure 5-3: CATD Algorithm Flowchart.

5.4 Methodology and Simulation Design

This section presents the methodology employed to evaluate the CATD framework's performance across diverse IoM deployment scenarios. The section describes the simulation environment, system parameter configuration, performance metrics, baseline algorithms, and testing framework that enables systematic evaluation of CATD's effectiveness under varying operational conditions.

5.4.1 Simulation Environment and Implementation

The evaluation utilises a discrete-event simulation framework implemented in Python, specifically designed to model IoM ecosystem characteristics and task distribution dynamics. The discrete-event simulation approach models the system as a sequence of instantaneous events (task arrivals, task assignments, task completions, node state changes) occurring at discrete points in time. This methodology enables precise modelling of asynchronous operations, concurrent task processing across nodes, and dynamic system state evolution while maintaining computational efficiency for large-scale scenario analysis.

The simulation framework comprises five interconnected modules coordinated by the discrete-event simulation engine, as illustrated in Fig. 5-4. The modular design separates infrastructure generation, network modelling, workload generation, task distribution, and performance measurement while allowing their states to evolve concurrently as simulation events occur.

Simulation Framework Modules: Detailed specifications for each module within the simulation framework are presented below:

- **Core Infrastructure Module**:The Core Infrastructure Module instantiates the hierarchical node architecture used throughout the evaluation. Node capabilities are drawn from uniform distributions within tier-specific ranges, with Hub nodes providing 800–1200 GFLOPS, Professional nodes 150–350 GFLOPS, and Consumer nodes 40–100 GFLOPS. These ranges represent the differing computational capacities associated with enterprise infrastructure, professional edge-computing platforms, and consumer embedded systems [215; 217; 218].

- **Network Topology Module**: The Network Topology Module constructs the three-tier hierarchical mesh using a graph-based representation. Nodes are geographically positioned within the 5×5 km deployment area, with Euclidean distance used to represent their spatial relationships. Communication paths are identified using latency-weighted shortest-path computation, while heterogeneous 5G, Wi-Fi, and LPWAN connectivity is modelled to represent the different communication environments available across the simulated ecosystem.

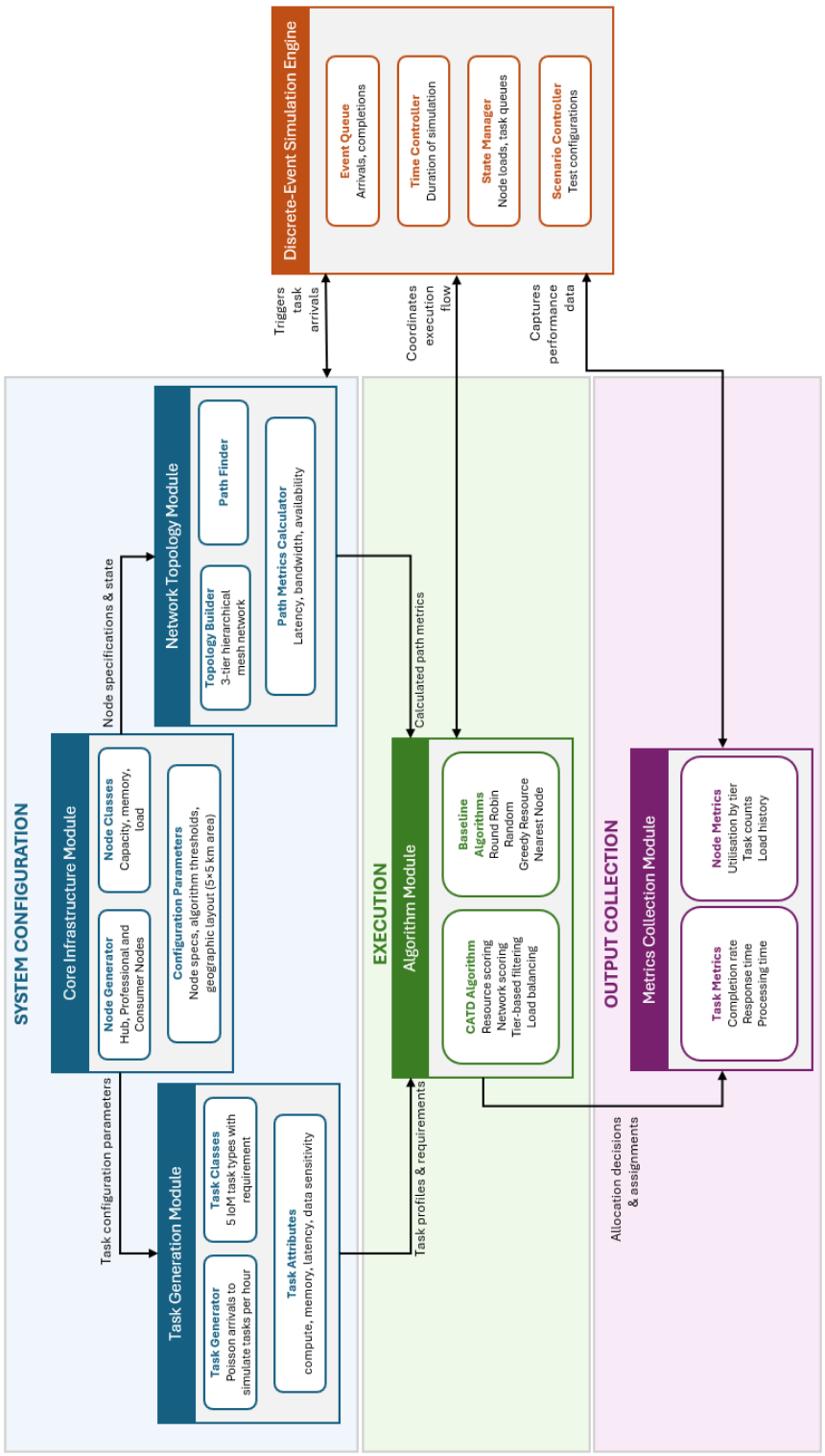


Figure 5-4: Simulation Framework Architecture and Module Interactions

- **Task Generation Module:** The Task Generation Module creates the heterogeneous IoM workloads used throughout the evaluation. Task arrivals follow Poisson processes [225] with configurable arrival rates between 15 and 180 tasks per hour. Task types are selected according to the distribution specified for each experimental scenario, and each generated task is instantiated with computational, memory, bandwidth, latency, sensitivity, and location characteristics derived from the task profiles established in Section 5.2.3.

- **Algorithm Implementation Module:** The Algorithm Implementation Module provides CATD and the four baseline allocation algorithms as interchangeable components operating within the same simulated environment. The module maintains current information describing node utilisation, task queues, and network conditions and updates this state as simulation events occur. CATD additionally applies the tier filtering and multi-factor scoring mechanisms defined in Section 5.3.1 and Algorithm 1.

- **Metrics Collection Module:** The Metrics Collection Module records task-, node-, and network-level behaviour throughout each simulation. Task-level measurements include completion and response times, allocation decisions, and success or failure status. Node-level measurements capture utilisation, task counts, load evolution, and queue state, while network-level measurements record path characteristics, bandwidth usage, latency, and link congestion. These measurements provide the data used to calculate the performance indicators and experimental outcomes presented later in the chapter.

Deployment Scenario Configuration: Geographic node placement follows hierarchical clustering patterns with the following tier composition:

- **Hub nodes** (20% of total node count) are positioned at urban centers using predefined coordinates representing major medical facilities or data centers
- **Professional nodes** (30% of total node count) are distributed in medium-density areas within 800m radius of hub locations, representing professional service establishments
- **Consumer nodes** (50% of total node count) are distributed throughout the deployment area using a combination of clustered placement near hubs (30% probability) and uniform random distribution (70% probability), reflecting residential and small retail patterns

This hierarchical composition intentionally creates realistic resource scarcity that challenges allocation algorithms, particularly limiting Hub nodes to approximately 20% of infrastructure to simulate the practical constraint of high-capability resources expected in real deployments.

Task Workload Modelling: The task generation system creates realistic application workloads through five representative categories detailed in Section 5.2.3. Task arrival follows Poisson processes with configurable base rates between 15-180 tasks per hour, enabling evaluation across light, moderate, and heavy workload conditions. A light workload reflects scenarios dominated by low-demand tasks such as Health Monitoring and System Management, a moderate workload introduces a balanced mix of all categories and a heavy workload increases the proportion of computationally intensive tasks such as Health Analytics and AR Beauty.

Task type distribution can be configured for different test scenarios:

- **Balanced distribution**: Equal 20% allocation across all five task types for algorithmic validation.
- **Workload-specific distributions**: Varied percentages emphasizing particular task categories for load scaling tests.
- **Sensitivity-skewed distributions**: Higher proportions of data-sensitive tasks for privacy-aware allocation evaluation.

Each generated task includes:

- **Resource requirements**: compute (GFLOPS) and memory (GB) drawn from task-type-specific ranges. Compute requirements span Health Monitoring (20-40 GFLOPS), System Management (15-35 GFLOPS), Product Recommendations (30-70 GFLOPS), AR Beauty (100-200 GFLOPS), and Health Analytics (300-500 GFLOPS), with memory scaled accordingly.
- **Network constraints**: Bandwidth requirement is calculated from the task data size and its latency tolerance, representing the minimum transmission rate required to transfer the task payload within the permitted time. The latency deadline is specified separately in milliseconds and represents the maximum communication delay tolerated by the task, these requirements are subsequently compared against the available path bandwidth and latency during node evaluation.
- **Data Sensitivity classification**: Sensitivity level (1-3) determining locality requirements.
- **Temporal characteristics**: Creation timestamp, processing duration estimated from compute requirements.
- **Contextual metadata**: Source location coordinates, task type identifier, unique task ID.

5.4.2 Performance Metrics

The evaluation framework capture several metrics related to system performance, resource efficiency, and task completion rate. These metrics are selected to enable direct assessment of the results presented in Section 5.5.

Primary Performance Indicators:

- **Task Completion Rate (η)**: The percentage of successfully allocated and completed tasks, calculated as:

$$\eta = \frac{\text{Completed Tasks}}{\text{Total Generated Tasks}} \times 100\% \tag{5-5}$$

  This metric provides the fundamental measure of system effectiveness under varying conditions.

- **Average System Latency ($\overline{L}_{sys}$)**: System latency is estimated within the discrete-event simulation from the elapsed time between task generation and completion rather than through a separate learned prediction model. For an allocated task, the latency comprises the task-allocation decision time, communication delay along the selected network path, processing duration at the allocated node, and the corresponding response transmission. Processing duration is determined from the computational requirements assigned to the generated task and the computational capability of the selected node, while the communication component is determined from the simulated characteristics of the selected network path. The reported average system latency is the mean of these end-to-end completion times across the tasks evaluated within each experimental condition.

- **Resource Utilisation by Tier (U_{tier})**: Time-weighted average computational resource usage for each node type (Hub, Professional, Consumer), calculated as:

$$U_{\text{tier}} = \frac{1}{T} \int_0^T \frac{1}{|N_{\text{tier}}|} \sum_{n \in N_{\text{tier}}} \text{Load}(n, t')\, dt' \tag{5-6}$$

  where T is the simulation duration, N_{tier} is the set of nodes in the tier, and $\text{Load}(n, t')$ is node n's load at simulation time t'.

5.4.3 Baseline Algorithms

To evaluate CATD's integrated multi-factor approach, four baseline algorithms are implemented within the same simulation framework. Each represents a different allocation principle against which CATD's combined consideration of task, resource, network, and locality context can be assessed:

- **Round Robin (RR):** Tasks are allocated sequentially across available nodes in a fixed order. The implementation cycles through nodes satisfying the basic capability requirements, providing allocation fairness without considering current resource load, network state, or detailed task–node compatibility.

- **Random:** Tasks are assigned through uniform random selection from nodes capable of servicing the task. This represents context-unaware allocation and provides a baseline against which the effect of informed placement decisions can be evaluated.

- **Greedy Resource:** Each task is assigned to the node with the greatest available computational capacity at allocation time. This represents resource-centric optimisation without explicit consideration of network conditions, geographic proximity, or data sensitivity.

- **Nearest Node:** Each task is allocated according to geographic proximity, selecting the nearest available node capable of servicing it. This represents proximity-oriented allocation without explicit consideration of current computational load or wider task requirements.

These are established single-factor allocation strategies used here to provide complementary reference behaviours against which CATD's integrated consideration of task, resource, network, and locality context can be assessed. Collectively, they represent fairness-oriented, context-unaware, resource-centric, and proximity-oriented allocation, allowing the contribution of CATD's combined contextual assessment to be examined against approaches that optimise or prioritise individual allocation dimensions.

The evaluation does not include stronger multi-criteria optimisation or learning-based task-offloading baselines. Consequently, the reported comparisons establish CATD's performance relative to the selected representative single-factor strategies, but should not be interpreted as demonstrating superiority over the broader state of the art in adaptive, optimisation-based, or learning-based offloading. A direct comparison with such methods would require evaluation under matched objectives, constraints, training assumptions, and computational budgets. Task scheduling in edge-cloud environments more broadly remains an active research area [174], within which CATD contributes an integrated context-aware formulation tailored to the IoM domain.

5.4.4 Experimental Testing Framework

The experimental evaluation follows a systematic four-category testing framework designed to assess CATD performance across operational dimensions as well as in comparison to the baseline algorithms detailed in Section 5.4.3. Table 5-3 summarises the complete testing matrix.

Table 5-3: Testing Setup and Configuration

Test ID	Test Name	Objective	Variable Parameter	Parameter Range/Method	Key Metrics
Category 1: Algorithm Validation					
1.1	Appropriate Allocation Validation	Validate task-to-node matching logic and allocation patterns	Task type distribution	Balanced 20% per type, multiple random seeds	Task allocation accuracy, resource matching efficiency, allocation pattern consistency
1.2	Randomisation Robustness Testing	Verify algorithmic consistency across random seed variations	Random seed variation	Five different seeds with identical scenarios	Allocation consistency (±2%), performance variance, logic preservation
Category 2: Performance Scaling and Capacity Analysis					
2.1	Workload Scalability	Identify system breakpoints and capacity limits	Task arrival rate	Incremental scaling	Breakpoint identification, system latency trends, throughput correlation
2.2	Resource-Intensity Robustness	Assess performance under varying computational complexity	Resource-intensive task percentage	0-100% increments focusing on high-compute tasks	Hub node utilisation, load distribution efficiency, resource-throughput ratio
Category 3: System Resilience and Fault Tolerance					
3.1	Resilience to Node Failure	Evaluate adaptability under infrastructure failures	Node failure rate	Progressive failure rates by node tier until system degradation	Performance degradation rates, load redistribution effectiveness, failure impact by tier
3.2	Resilience to Bandwidth Degradation	Test performance under communication constraints	Network bandwidth reduction	Incremental bandwidth reduction on subset of connections	Network adaptation effectiveness, routing optimisation, latency impact
Category 4: Context-Awareness and Adaptability					
4.1	Dynamic Multi-Phase Adaptability	Assess adaptation to changing operational conditions	System conditions (multi-phase)	Sequential phases: Baseline → Network stress → Topology disruption → High compute demand → Stabilisation	Adaptation latency, transition stability, cross-phase performance
4.2	Data Sensitivity and Locality Optimisation	Evaluate privacy-aware allocation and geographic optimisation	Task sensitivity level distribution	Varied sensitivity distributions with geographic clustering	Privacy satisfaction rate, task-to-node distance by sensitivity, locality compliance

Category 1: Algorithmic Validation and Stability: These tests validate fundamental algorithmic correctness and consistency:

Test 1.1 - Appropriate Allocation Validation examines whether CATD's tier-based filtering logic correctly directs tasks to node tiers matching their computational requirements. Using a balanced workload (20% per task type), five independent simulation runs analyse the distribution of task-to-tier assignments. Validation criteria include: (i) resource-intensive tasks (Health Analytics) predominantly allocated to Hub nodes, (ii) medium-complexity tasks (AR Beauty, Product Recommendations) distributed appropriately between Professional and Hub tiers, and (iii) lightweight tasks (Health Monitoring, System Management) efficiently utilising Consumer and Professional nodes.

Test 1.2 - Randomisation Robustness: This test verifies that CATD's allocation logic produces consistent tier-level decisions across stochastic variations. Five simulation runs with identical scenario parameters but different random seeds assess whether core allocation behaviour remains stable. The key validation criterion is that tier-boundary enforcement remains deterministic: resource-intensive tasks should never allocate to Consumer nodes, and lightweight tasks should show minimal Hub allocation across all runs. Variation within appropriate tiers (e.g., distribution between Hub and Professional for medium tasks) is expected and reflects load-aware balancing rather than algorithmic inconsistency.

Category 2: Performance Scaling and Capacity Analysis: These tests identify system capacity limits and performance degradation patterns:

Test 2.1 - Workload Scalability: This test incrementally increases task arrival rates (i.e. the workload) from baseline until the system completion rate drops below 50%. This is defined as the threshold because it represents the crossover point where task failures exceed successes. Load is expressed as the dimensionless ratio $\rho = \lambda/\mu_{\text{sys}}$, where λ is the task arrival rate and μ_{sys} is the aggregate service rate of the system, i.e. the number of tasks the deployment can complete per unit time at full utilisation. This metric not only enables generalisation across different deployment scales but also allows systematic comparison of algorithm resilience as load increases beyond sustainable levels.

Test 2.2 - Resource-Intensity Robustness: This test varies the percentage of high-resource tasks (Health Analytics: 300-500 GFLOPS, AR Beauty: 100-200 GFLOPS) from 0% to 100% while maintaining a constant arrival rate. This test evaluates algorithmic resilience in handling heterogeneous workload composition, particularly assessing Hub node utilisation patterns and load distribution effectiveness as computational demands increase.

Category 3: System Resilience and Fault Tolerance: These tests assess system adaptability under adverse conditions:

Test 3.1 - Resilience to Node Failure: This test progressively disables nodes at increasing failure rates, ranging from 0% to 100% of total infrastructure across all tiers, limiting task allocation options. The test measures completion rate at each failure level to construct degradation curves, enabling comparison of algorithm resilience. Additionally, tier-specific failures are analysed in isolation to assess the relative impact of Hub, Professional, and Consumer node losses on system performance.

Test 3.2 - Resilience to Bandwidth Degradation: This test reduces available bandwidth on a subset of network links (30-50% of connections) by 10-90%, simulating congestion or infrastructure constraints. Links are selected to affect different parts of the topology. The key metric being tested is the average task latency as network links become progressively congested.

Category 4: Context-Awareness and Adaptability: These tests evaluate CATD's capabilities in adapting to dynamic conditions and data sensitivity requirements:

Test 4.1 - Multi-Phase Dynamic Adaptability simulates evolving operational conditions through sequential phases, one phase followed by the other, to mimic evolving ecosystem conditions. These phases are: (i) baseline operation, (ii) bandwidth reduction (-30%), (iii) node failures (20%), (iv) load surge ($+40\%$ arrival rate), (v) recovery to baseline. Each phase runs for sufficient duration to achieve steady-state (minimum 300 task completions). This test assesses: transition stability between phases and adaptation effectiveness to new conditions.

Test 4.2 - Data-Sensitivity and Locality Optimisation varies task sensitivity distributions from uniform (equal percentages across sensitivity levels 1-3) to level 2 and then level 3 dominant. The test evaluates: (i) compliance rate—percentage of Level 3 tasks allocated within local proximity threshold, (ii) task-to-node distance metrics by sensitivity level.

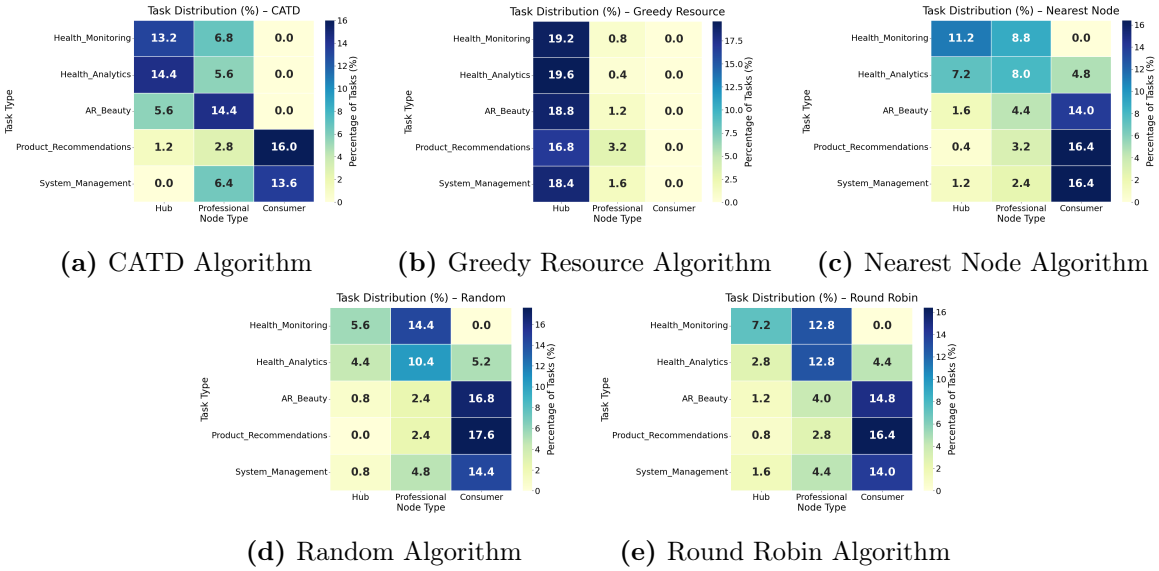
5.5 Results and Discussion

This section presents the evaluation of the CATD framework with the approach detailed in Section 5.4.4. Results demonstrate CATD's superior performance compared to baseline algorithms.

5.5.1 Category 1: Algorithm Validation

Test 1.1: Appropriate Allocation Validation: Fig. 5-5 reveals fundamental differences in the allocation effectiveness through heatmap visualisations. CATD exhibits hierarchical intelligence: Health Analytics tasks route predominantly to Hub nodes,

reflecting their high computational demands. AR Beauty tasks, with medium-high computational requirements, show balanced Professional-Hub distribution. Lower-demand tasks are efficiently directed to appropriate tiers: Product Recommendations and System Management allocate primarily to Consumer nodes, while Health Monitoring utilises both Consumer and Professional tiers.



(a) CATD Algorithm **(b)** Greedy Resource Algorithm **(c)** Nearest Node Algorithm

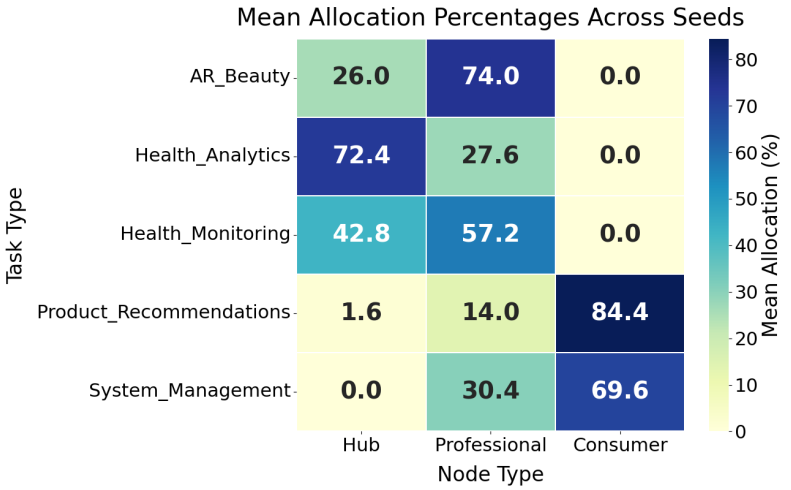
(d) Random Algorithm **(e)** Round Robin Algorithm

Figure 5-5: Distribution of task allocation percentages across different node types for CATD and baseline algorithms.

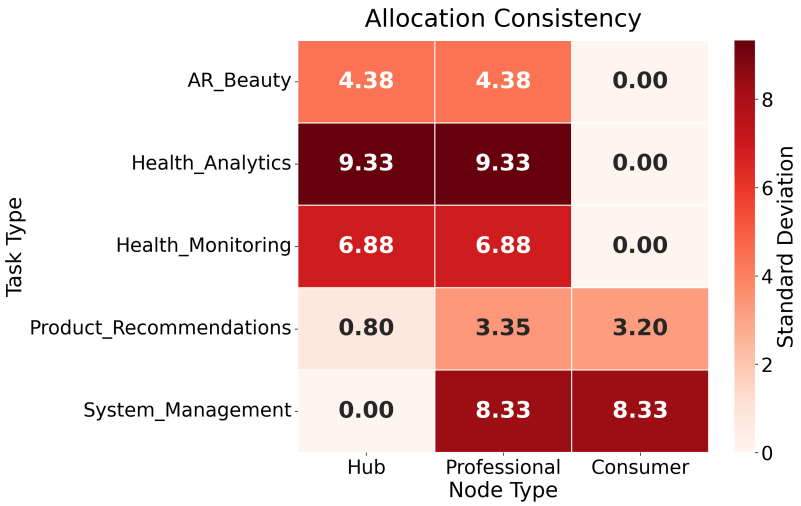
CATD's tier-appropriate matching optimises the use of scarce high-capability resources, a critical consideration for IoM deployments. In the simulated infrastructure, Hub nodes represent only 20% of deployed devices but provide 60-70% of total computational capacity. By routing lightweight tasks (System Management, Health Monitoring) to the abundant Consumer and Professional tiers (80% of infrastructure), CATD preserves Hub capacity for computationally intensive analytics that cannot execute elsewhere. This prevents the resource misallocation seen in baseline algorithms. For example, Greedy Resource over-allocates Hub nodes even for lightweight tasks, directing 45.8% of Product Recommendation tasks to Hub nodes, unnecessarily consuming scarce capacity and creating artificial bottlenecks that degrade performance under load (see Section 5.5.2). The Random baseline algorithm distributes uniformly without capability awareness (5.2% of Health Analytics to inadequate Consumer nodes), while Nearest Node and Round Robin ignore task-node computational fit entirely. The CATD framework demonstrates substantially higher tier-appropriate allocation compared to baselines, as evidenced by the concentration of resource-intensive tasks at Hub nodes and lightweight tasks at Consumer nodes. This intelligent matching prevents the resource misallocation seen in baseline algorithms.

Test 1.2: Randomisation Robustness Testing: Fig. 5-6 demonstrates CATD's algorithmic robustness across five independent runs with different random seeds. Fig. 5-6 (a) heatmap shows mean allocation percentages, while the Fig. 5-6 (b) displays SDs

as a measure of consistency across runs.



(a) Allocation consistency across different seeds.



(b) Performance variance across different seeds.

Figure 5-6: Allocation consistency and performance variance across different seeds.

The results demonstrate consistent enforcement of tier-appropriate boundaries. High-resource tasks (Health Analytics) never allocate to Consumer nodes across any run (SD = 0.00), and lower-demand tasks (Product Recommendations, System Management) show minimal Hub allocation (SD $\leq$ 0.80). The observed variation occurs within acceptable tier combinations. For example, Health Analytics, a resource-intensive task primarily targeting Hub nodes, distributes between Hub (72.4%) and Professional (27.6%) tiers. This distribution reflects CATD's adaptive behaviour under resource constraints: when Hub nodes approach capacity limits, the algorithm allocates demanding tasks to Professional nodes rather than risking task failure due to Hub unavailability. This trade-off prioritises successful completion over outright rejection, consistent with CATD's design objective of maximising task completion under dynamic conditions. The SD of 9.33 across runs reflects varying Hub availability rather than algorithmic inconsistency. Similarly, AR Beauty tasks maintain balanced distribution between Professional (74.0%) and Hub (26.0%) tiers, with low SDs (4.38) confirming

consistent medium-tier allocation. This pattern confirms that CATD's core logic of tier-boundary enforcement for capability matching remains deterministic, while node-level selection adapts to runtime conditions, a desirable property for load balancing in heterogeneous systems.

5.5.2 Category 2: Performance Scaling and Capacity Analysis

Test 2.1: Workload Scalability: Fig. 5-7 examines system breakdown behaviour under progressive load stress testing. Load is expressed as ρ, the ratio of task arrival rate to baseline system capacity (defined in Section 5.4.4). Values of $\rho > 1$ represent intentional overload conditions designed to reveal performance degradation patterns and comparative breakdown thresholds.

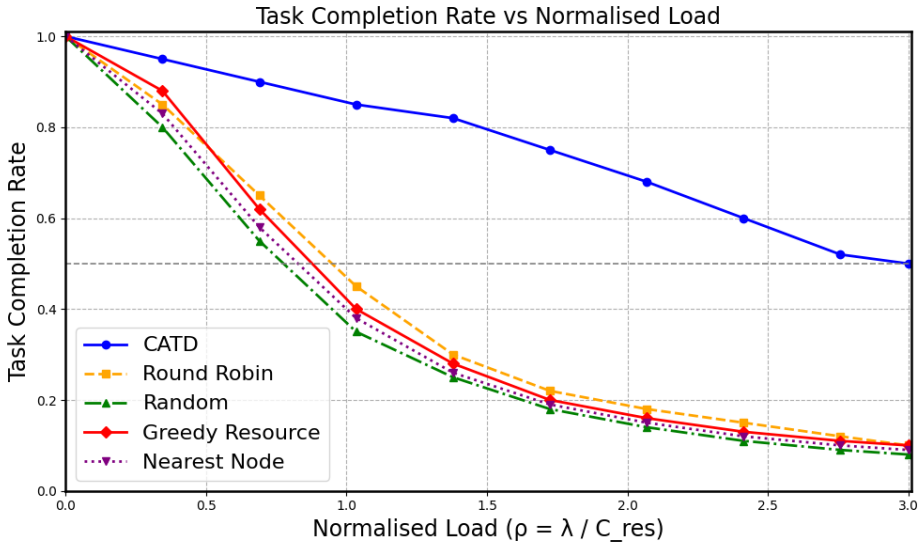


Figure 5-7: Task completion rate against normalised system load.

Baseline algorithms exhibit rapid degradation as system load approaches baseline capacity. Greedy Resource, Random, and Nearest Node remain above 50% completion at $\rho \approx 0.7$ but fall below 50% by $\rho \approx 1.05$. Round Robin remains marginally higher than the other baselines at this point, but also falls below 50% completion. CATD demonstrates substantially more gradual degradation, maintaining $> 50\%$ completion until $\rho \approx 3$. At $\rho = 2$, CATD achieves approximately 65% completion while all baselines fall below 20%.

CATD's comparatively better performance stems from its load-aware scoring mechanism (Equation 5-1) combined with tier-based escalation. As system load increases, the scoring function dynamically penalises heavily-loaded nodes, redistributing incoming tasks to nodes with available capacity rather than allowing bottlenecks to form. When preferred tiers approach saturation, the CATD algorithm escalates tasks to alternative tiers, delaying system saturation through adaptive redistribution. In contrast, baseline algorithms lack mechanisms to respond to increasing load: Greedy Resource continues targeting high-capacity nodes regardless of their current state, Nearest Node creates ge-

ographic hotspots, and Round Robin cycles indiscriminately through saturated nodes. This resilience is particularly valuable for IoM deployments, which may experience unpredictable demand surges during peak usage periods—such as retail environments during promotional events or clinical settings during high patient intake. CATD's ability to maintain approximately 65% task completion at 2× baseline capacity, while all baseline algorithms fall below 20%, demonstrates substantially greater resilience under overload conditions. This can provide substantially longer windows for implementing protective responses (traffic shaping, auto-scaling).

Test 2.2: Resource-Intensity Robustness: Fig. 5-8 examines performance under varying workload composition while maintaining constant arrival rate, progressively increasing the resource-intensive task percentage. CATD maintains approximately 75% or higher completion through a 60% resource-intensive workload. By 40%, all baseline algorithms have declined to approximately 62–66% completion, compared with approximately 82% for CATD. At 60%, CATD achieves approximately 75% completion compared with approximately 47–52% for the baselines. At 80%, CATD maintains approximately 63% completion, while the baseline algorithms fall to approximately 27–34%.

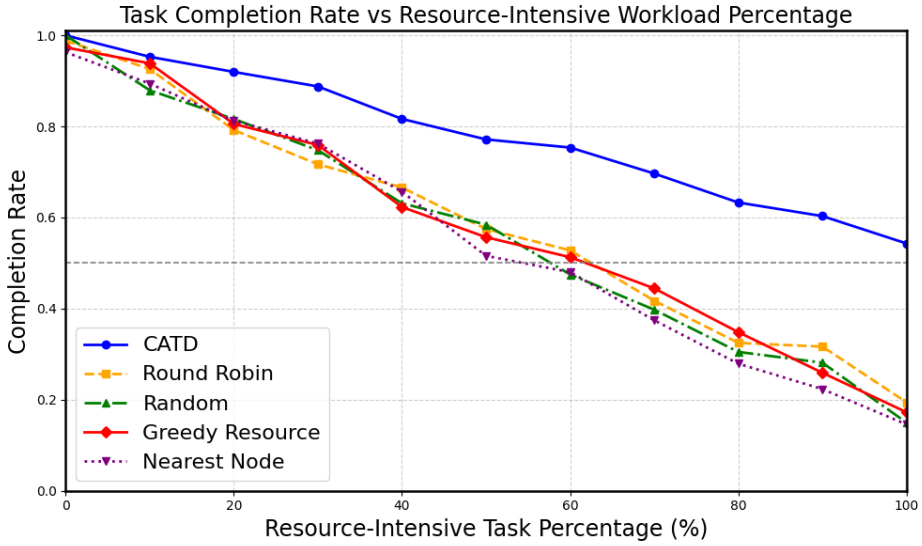
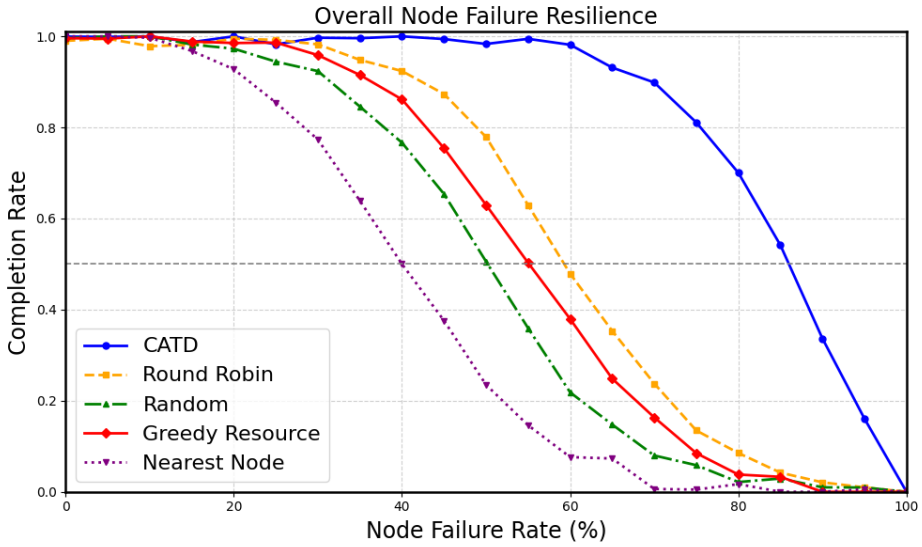


Figure 5-8: Task completion rate versus resource-intensive workload percentage.

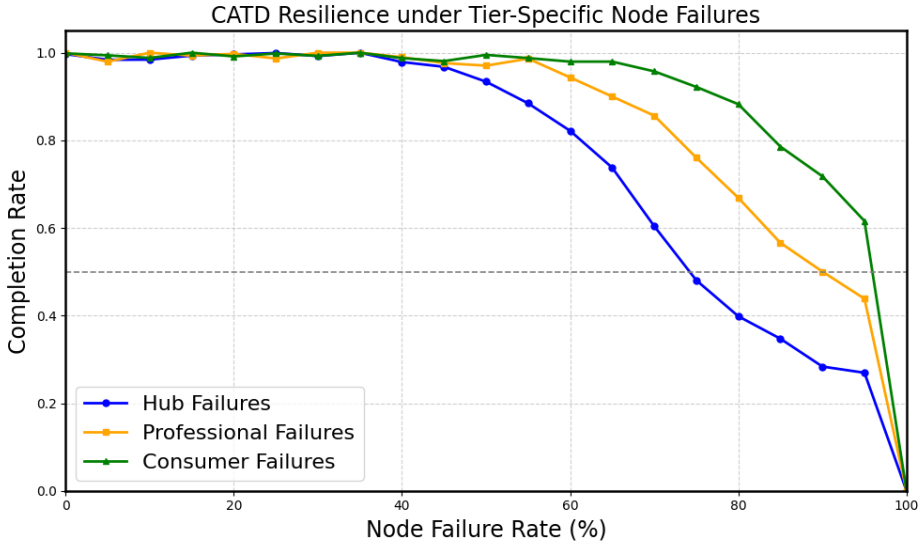
CATD's compositional adaptability stems from dynamic tier reallocation, as resource-intensive tasks increase, the framework efficiently routes demanding workloads to Hub nodes while redistributing lightweight tasks downward, preserving Hub capacity for tasks requiring it. Baseline algorithms exhibit rigid allocation patterns where Greedy Resource over-commits Hub nodes regardless of workload composition, while Random and Round Robin distribute uniformly without recognising computational tier requirements. This demonstrates that static allocation strategies optimised for average workloads fail when composition shifts substantially from expected distributions.

5.5.3 Category 3: System Resilience and Fault Tolerance

Test 3.1: Resilience to Node Failure: Fig. 5-9a and Fig. 5-9b examine system resilience under progressive infrastructure failures, with nodes randomly disabled across all tiers to simulate hardware failures, network partitions, or maintenance events. CATD maintains above 50% task completion through approximately 85% node failure, falling below this level between 85% and 90% failure. Among the baselines, Nearest Node falls below 50% completion after approximately 40% node failure, followed by Random after approximately 50%, while Greedy Resource and Round Robin fall below this level at approximately 55-60% failure.



(a) Task completion rate as a function of node failure percentage.



(b) CATD performance under tier-specific node failures.

Figure 5-9: Resilience analysis under varying node failure conditions.

The tier-specific analysis in Fig. 5-9b reveals CATD's resilience mechanism. Consumer tier failures (50% of infrastructure) cause minimal degradation, maintaining >95% completion until 60% Consumer failures due to high redundancy. Professional failures produce moderate impact, with performance declining linearly. Hub failures, despite

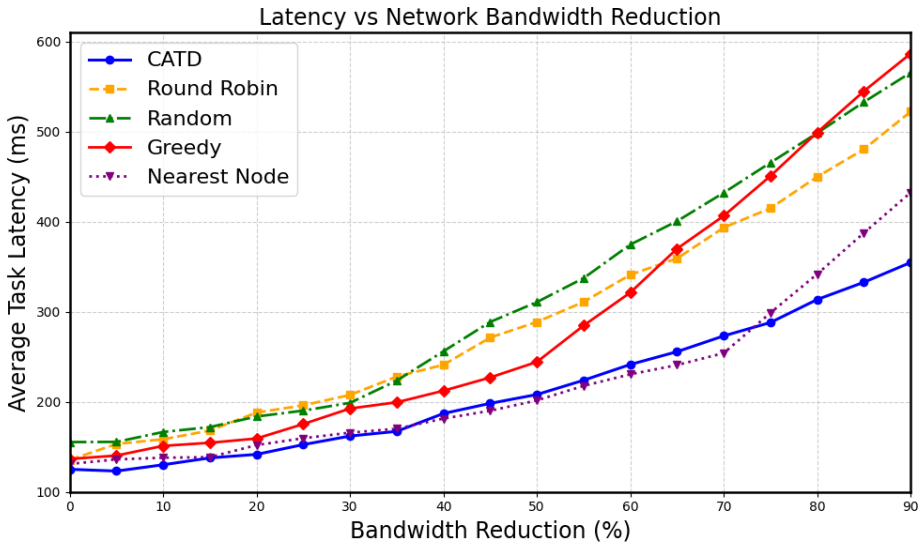
representing only 20% of infrastructure, create disproportionate degradation due to computational criticality, yet CATD maintains >50% completion until 70% Hub failures through aggressive task redistribution. This demonstrates that hierarchical systems require failure handling mechanisms recognising both computational criticality and redundancy patterns.

This fault tolerance is critical for IoM deployments, where node availability is inherently variable. Consumer-tier devices in residential settings may disconnect due to network issues or user behaviour, while Professional nodes in clinics may be taken offline for maintenance. IoM architectures must anticipate infrastructure unavailability as a normal operating condition. CATD's ability to maintain >50% completion through approximately 85% node failure demonstrates greater resilience under progressive infrastructure loss, indicating improved service delivery in this scenario.

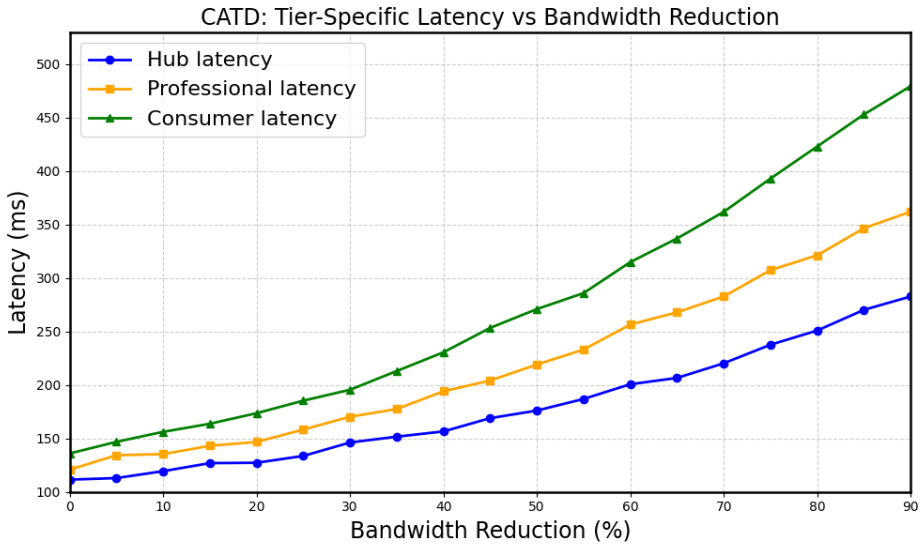
test 3.2: Resilience to Bandwidth Degradation: Fig. 5-10 demonstrates CATD's network-aware adaptation under bandwidth degradation. CATD maintains lower latency than all baselines, with apparent advantages as constraints intensify. The Random algorithm suffers immediate spikes due to link-unaware selection; Greedy experiences mid-range collapse as preferred high-capacity links saturate; Nearest Node performs initially well but fails sharply when local links congest.

The tier-specific latency analysis reveals how CATD's adaptation mechanism leads to Hub backbone latency being preserved the longest, Professional-level paths show moderate degradation, and only under severe constraints the algorithm shifts toward Consumer-level paths, where latency grows steeply.

This prioritisation reflects the scoring system's network quality evaluation combined with tier preference. Network-aware task distribution should consider both link characteristics and computational topology. CATD's integrated scoring mechanism enables dynamic path selection that preserves backbone performance while shifting to alternative paths only under constraint, thereby helping to maintain service quality under variable network conditions. This compositional adaptability is particularly relevant for IoM deployments, where workload profiles may vary significantly across contexts and time. A hospital dermatology department may generate predominantly resource-intensive Health Analytics tasks, while a retail beauty kiosk primarily invokes lightweight Product Recommendations and individual deployments may shift between profiles throughout the day as user behaviour changes. CATD's ability to maintain performance as resource-intensive task proportions increase would ensure consistent service quality across these varying operational conditions.



(a) The effect of bandwidth reduction on task latency.



(b) CATD tier specific latency response under bandwidth reduction.

Figure 5-10: Performance analysis in response to bandwidth degradation.

5.5.4 Category 4: Context-Awareness and Intelligent Adaptation

Test 4.1: Dynamic Multi-Phase Adaptability: Table 5-4 demonstrates CATD's performance across sequential operational phases simulating realistic condition evolution. CATD maintains 12-30 percentage point advantages over baselines across all stress conditions, with particularly pronounced gaps during node failures (Phase III: 28-30 points) and load surges (Phase IV: 25-27 points).

This emerges from cross-phase comparison, CATD's completion rate degrades by smaller margins than the baselines at each stress phase ($100 \rightarrow 92 \rightarrow 88 \rightarrow 90 \rightarrow 98$), whereas the baselines exhibit larger drops, particularly under topology disruption and load surge. During recovery (Phase V), CATD returns to 98% completion, a higher level than all baselines (80–92%), indicating effective re-stabilisation once conditions normalise.

Table 5-4: Task completion rates (%) across dynamic operational phases.

Phase	Condition	CATD	Nearest	Greedy	RR	Random
I. Baseline	Normal operation	**100**	96	94	92	90
II. Bandwidth −30%	Network stress	**92**	83	78	70	65
III. Node failure 20%	Topology disruption	**88**	80	73	60	58
IV. Load surge +40%	High compute demand	**90**	82	74	65	63
V. Recovery	Stabilisation	**98**	92	88	85	80

CATD exhibits not only better steady-state performance but also superior transient behaviour during condition changes. The ability to adapt allocation strategies dynamically to environmental factors such as network quality, resource availability and load intensity, proves essential for maintaining service continuity in real-world IoT and specifically IoM deployments where conditions may fluctuate continuously.

Test 4.2: Data Sensitivity and Locality Optimisation: Fig. 5-11 reports the normalised average locality distance, i.e. the mean distance from task source to allocated node, across the three data-sensitivity levels. CATD exhibits a clear sensitivity-aware trend: as data sensitivity increases from Low to High, its average allocation distance falls from approximately 0.60 to 0.26, indicating that sensitive tasks are progressively placed on nearer nodes. In contrast, Round Robin, Random, and Greedy Resource remain largely flat across all sensitivity levels, as none of these baselines incorporate data sensitivity into their allocation decisions.

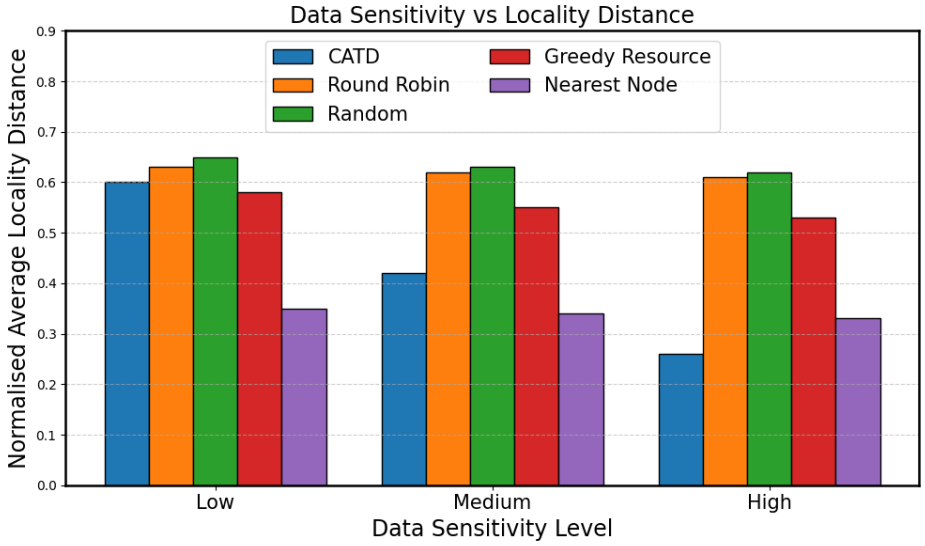


Figure 5-11: Data Sensitivity and Locality Analysis

Nearest Node maintains a consistently low allocation distance across the sensitivity levels because geographic proximity is its sole selection criterion and is applied iden-

tically to every task. CATD instead varies the influence of proximity according to task sensitivity while also considering resource and network conditions. Consequently, lower-sensitivity tasks may be allocated to more distant suitable resources, while highly sensitive tasks receive a stronger preference for nodes within the local proximity threshold.

The lower average distance achieved by CATD for High-sensitivity tasks should therefore be interpreted as a system-level effect of sequential allocation rather than CATD selecting a nearer node than Nearest Node from an identical candidate set at a single decision point. Because Nearest Node consumes nearby available resources indiscriminately across all sensitivity levels, those resources may already be occupied when highly sensitive tasks arrive. CATD's sensitivity-aware allocation preserves greater opportunity to use suitable local resources where locality is most important, producing the lower average distance observed for High-sensitivity tasks in Fig. 5-11.

5.5.5 Broader Implications

The test evaluation reveals a consistent pattern CATD's advantages stem from integrated multi-factor evaluation that adapts to system context. Under normal conditions (Test 1.1), this presents as intelligent tier matching. Under overload (Tests 2.1-2.2), it enables efficient hierarchy utilisation. Under failures (Tests 3.1-3.2), it provides adaptive resilience. Under dynamic conditions (Test 4.1), it maintains consistent performance through environmental changes. The performance gaps between CATD and baselines, represent qualitative differences in system behaviour. These advantages emerge directly from context-awareness, understanding not just what resources exist, but how they relate to tasks, network conditions, and operational context.

This integrated approach proves particularly valuable for IoT environments characterised by heterogeneous infrastructure (multiple capability tiers), diverse applications and dynamic conditions. While developed for IoM applications, CATD's principles of hierarchical awareness, multi-factor scoring and dynamic adaptation could be applied broadly to similar tiered computing environments including smart healthcare systems, industrial IoT, and smart cities.

5.6 Summary

This chapter addressed Q_3 and O_3 by developing and evaluating the CATD framework for coordinating heterogeneous application demands across the Consumer, Professional, and Hub hierarchy of the IoM. Building on the context dependence identified experimentally in Chapter 4, CATD formulates task distribution as a multi-factor compatibility problem rather than a fixed placement decision. Candidate resources are first constrained according to the computational requirements of the task before individual

nodes are evaluated according to their residual capacity, network quality, geographic proximity, and data-sensitivity requirements.

The framework was evaluated through discrete-event simulation of a representative multi-tier IoM deployment containing heterogeneous computational resources, network conditions, and five application task profiles. CATD was compared with Round Robin, Random, Greedy Resource, and Nearest Node allocation across eight experimental tests covering allocation behaviour, increasing system load, changing workload composition, node failures, bandwidth degradation, multi-phase operating conditions, and data locality. The results demonstrate that considering these contextual factors jointly produces behaviour that cannot be achieved consistently by allocation strategies optimising a single dimension in isolation.

The evaluation showed that CATD preserves high-capability Hub resources for tasks that require them while allowing lighter workloads to utilise Consumer and Professional resources. This behaviour contributes to greater resilience as demand increases, with CATD maintaining task completion beyond the load levels at which the baseline approaches degrade substantially. The framework similarly redistributes tasks as nodes become unavailable, responds to changing network conditions, and alters placement according to task sensitivity rather than applying the same locality preference to all workloads. Across the dynamic multi-phase evaluation, CATD maintained higher task completion than the baseline approaches as network, topology, and workload conditions changed, demonstrating the value of incorporating current ecosystem context into task placement decisions.

These findings extend the system-level observations of Chapter 4, which established that the preferred execution location changes with computational capability, communication conditions, and system demand. This chapter demonstrates a mechanism through which these differences can be incorporated into individual allocation decisions as the IoM supports heterogeneous applications and users. Context-aware distribution therefore provides a means of coordinating the IoM's internal computational resources while retaining the stakeholder-oriented tier hierarchy established in Chapter 2.

The present evaluation focuses on allocation effectiveness and system-level performance under changing operating conditions; the computational cost and runtime overhead introduced by the CATD decision process itself were not separately instrumented and remain an important consideration for future physical deployment. Nevertheless, CATD operates through an explicitly defined scoring and tier-selection mechanism within the boundaries of the IoM ecosystem. The next stage of the investigation broadens this decision context beyond the IoM's own infrastructure, where task requirements may additionally depend on ecosystem origin, data modality, compatible hardware, and application urgency. Chapter 6 therefore extends task coordination towards adaptive, learning-based orchestration between the IoM and external IoT ecosystems.

Chapter 6

Bridging the IoM and the Internet of Sounds (IoS): An RL-Driven Multimodal Task Orchestration Approach

6.1 Introduction

The preceding chapters progressively established the IoM as a distributed computational ecosystem. Chapter 4 demonstrated that the suitability of different execution locations depends on the interaction between computational capability, communication conditions, and system demand. Chapter 5 subsequently addressed this context dependence through the CATD framework, which coordinates heterogeneous application workloads across the Consumer, Professional, and Hub hierarchy according to task requirements and current ecosystem conditions. These evaluations establish mechanisms for distributing computation within the IoM itself. However, the wider vision established in Chapter 2 extends beyond an isolated mirror ecosystem and considers the IoM as one component within a broader landscape of interconnected Internet of Things (IoT) ecosystems.

The continued development of specialised IoT ecosystems around particular sensing modalities and application domains creates opportunities for services that draw upon capabilities distributed across more than one ecosystem. Examples include the IoV, IoMT and IoS, each extending conventional IoT connectivity towards domain-specific devices, interactions, and services [6; 9; 8]. The IoS in particular encompasses interconnected Sound Things capable of sensing, processing, actuating, and exchanging sound-related information across distributed environments [226; 8]. The IoM provides a complementary visually oriented ecosystem centred on sensing, personalised interaction, and assistive health and beauty services. Their co-deployment therefore creates the possibility for visual and auditory information to contribute to the same user-

facing service without requiring the underlying ecosystems to be merged into a single platform.

Coordinating such services introduces requirements that are not fully represented by computational heterogeneity alone. Distributed task-offloading research has already shown that effective placement can depend on interacting factors including resource availability, latency, reliability, network conditions, and application requirements [227; 172; 184]. Multimodal workloads introduce an additional source of heterogeneity because the processing requirements of a task may depend on its data modality and the capabilities available at candidate execution resources [171]. In a cross-ecosystem setting, an externally originated task may therefore carry a modality that only a subset of available nodes can process, enter the IoM through a different source from a native task, and have an urgency determined by the application context in which it was generated. These characteristics interact with the dynamic system conditions considered in Chapter 5, including node utilisation, queue state, network bandwidth, and reliability.

The resulting orchestration problem requires decisions to adapt as both task characteristics and system conditions change. RL has increasingly been applied to task offloading, resource allocation, and scheduling in IoT and edge-computing environments because it enables policies to be learned through interaction with dynamic system states rather than requiring a fixed allocation rule for every operating condition [181; 180]. Deep RL approaches have subsequently been applied to dynamic edge–cloud task allocation [184], hierarchical offloading [183], and multi-objective resource scheduling [189]. As reviewed in Chapter 3, however, the present problem additionally requires feasibility to be considered explicitly because a computationally available node cannot process a task for which it lacks the necessary modality-specific capability. The orchestration mechanism developed in this chapter therefore combines learning-based allocation with hard action masking and context-conditioned objectives, allowing the policy to adapt between feasible resources while preventing assignments that violate modality or stakeholder-access constraints. Double DQN learning is used to realise this adaptive policy [182].

This chapter addresses Q_4 and O_4 through the cross-ecosystem orchestration approach reported in [C4], extending task coordination beyond the IoM's internal hierarchy to an ecosystem-to-ecosystem interaction between the IoM and IoS. An Intelligent Orchestration Layer is introduced as a control plane through which native visual IoM tasks and externally originated audio IoS tasks can be evaluated within a common decision process. The formulation incorporates ecosystem origin, task modality, hardware compatibility, system state, and application context, with task urgency represented through Live-Interactive, Ambient-Monitoring, and Background-Batch modes. Hard action masking restricts node selection to feasible execution locations, while a mode-conditioned reward allows the relative importance of latency, deadline satisfaction, resource efficiency, and packet loss to vary with the context in which a task is generated.

A home-based vocal rehabilitation session is used as a representative multimodal scenario in which an IoM smart mirror and a co-deployed IoS Sound Thing contribute complementary visual and audio observations to the same user interaction. Voice-assisted technologies provide an appropriate setting for examining this interaction because home-based speech and voice therapy is an active area of assistive technology research [228]. Audio-processing workloads additionally provide a meaningful external modality because IoS research encompasses distributed sound sensing, processing, and communication across connected Sound Things [8], while latency and reliability have been identified as important characteristics of interactive networked audio systems [145]. The scenario therefore provides a concrete workload through which cross-ecosystem orchestration can be evaluated without restricting the proposed formulation to a single application.

The resulting Double DQN policy is evaluated through three stages. First, its learned behaviour is validated to establish whether task placement reflects the modality constraints and context-dependent priorities encoded by the formulation. Second, orchestration of IoS workloads through the IoM hierarchy is compared with remote cloud offloading under increasing system load, considering task completion, end-to-end latency, deadline satisfaction, and packet-loss behaviour. Finally, the availability of audio-capable processing resources is varied to examine how the orchestration mechanism responds to cross-ecosystem modality heterogeneity. Together, these evaluations investigate whether the IoM can extend its distributed computational capability beyond its native applications and operate as an adaptive resource within a wider interconnected IoT environment.

6.2 System Architecture and Model

Fig. 6-1 shows the overall architecture, spanning the IoM's own structure, the orchestration layer introduced in this chapter, and its connection to the co-deployed IoS.

6.2.1 IoM Architecture

The IoM is deployed as a three-tier hierarchical network of nodes, reflecting increasing compute capability rather than a restriction on which nodes can communicate with one another:

- **Consumer tier** - home mirrors, retail kiosks, and personal devices
- **Professional tier** - clinics, salons, and pharmacies
- **Hub tier** - hospitals, enterprise servers, and service providers

Independently of this tier hierarchy, the IoM architecture is organised into three functional layers:

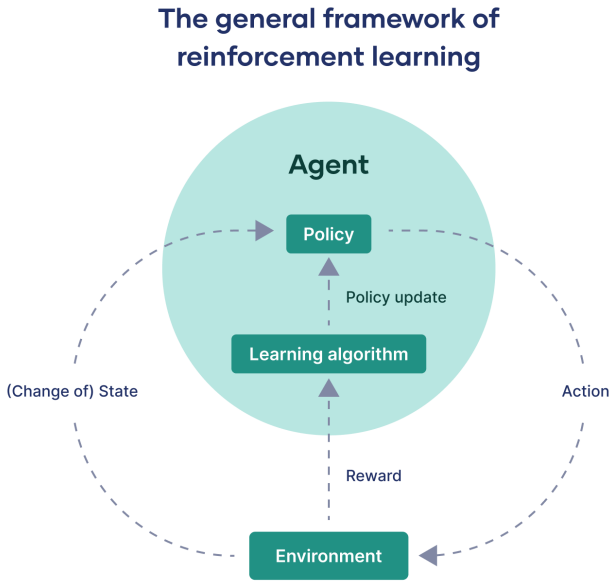


Figure 6-1: IoM system architecture, showing the tier hierarchy and functional layers, the Intelligent Orchestration Layer, and its connection to the co-deployed IoS and the broader IoX vision.

- **Perception layer** - sensing and user interaction through the smart-mirror interface
- **Network layer** - cross-tier connectivity, bandwidth allocation, and quality of service
- **Application layer** - service execution and delivery at the assigned tier node, spanning health, beauty, and assisted-living services

The tier hierarchy describes *where* computation happens; the functional layers describe *what kind* of work is done at each stage. This tiered structure and the latency-privacy trade-offs it introduces have been characterised in Chapter 4 through an edge-cloud latency analysis and a physical testbed evaluation of computational placement strategies across the tier hierarchy.

6.2.2 Orchestration Layer

The Intelligent Orchestration Layer is a control plane that coordinates task assignment across the tier hierarchy and functional layers shown in Fig. 6-1: it reads context signals and metrics from the Perception layer and network state from the Network layer, and writes its routing decision to the Application layer, where the task is then executed at the assigned tier node.

Operationally, the Intelligent Orchestration Layer acts between task generation and service execution. When a task is generated by either the IoM or the connected IoS, the layer first receives its task profile, including its ecosystem of origin, modality,

trigger, and service requirements. The Context Inference Module uses the task trigger and session state to determine its operating context, while the Constraint and Masking Layer identifies the processing nodes that are feasible according to modality-hardware compatibility and stakeholder-tier access. The resulting feasible action set is combined with the current node and network state and passed to the Double DQN agent, which selects either a processing node or the defer action. The task is then executed at the selected node and the resulting latency, packet loss, deadline satisfaction, and resource use are evaluated through the mode-conditioned reward. During training, this outcome forms the learning signal used to update the routing policy for subsequent decisions.

The orchestration layer therefore combines predefined system constraints with learned task-routing behaviour. Modality compatibility, tier-access rules, context-mode determination, and the structure of the reward are specified explicitly by the system design. The Double DQN does not learn these constraints; rather, it learns which feasible routing action is preferable under the observed task and system state. This separation ensures that learning operates within the architectural and hardware constraints of the IoM while allowing routing behaviour to adapt according to the outcomes experienced during training.

Chapter 5 demonstrated that, for single-modality tasks generated within the IoM's own tier hierarchy, task distribution can be performed using a deterministic, context-aware allocation mechanism. CATD is dynamic at the point of allocation: each incoming task is evaluated against the current resource availability, network conditions, geographic proximity, and data-sensitivity requirements, meaning that the selected execution resource can change as system conditions change. However, the decision logic itself remains predefined and deterministic; CATD does not learn from previous allocation outcomes or update its allocation policy during operation. This is appropriate for the bounded internal IoM setting considered in Chapter 5, where the task categories, relevant contextual factors, and tier structure are defined in advance.

The orchestration problem considered in this chapter extends this setting beyond the IoM's own hierarchy to tasks originating across the IoM and an externally deployed ecosystem. In addition to changing resource and network conditions, the allocation decision must account for ecosystem origin, multiple data modalities, modality-dependent hardware compatibility, and differing service urgency. These additional factors alter both which resources are feasible for a task and which routing behaviour is appropriate under the current context. In this formulation three representative contexts are defined, Live-Interactive (LI), Ambient Monitoring (AM) and Background-Batch (BB). The Intelligent Orchestration Layer shown in Fig. 6-1 therefore acts as an adaptive control plane, using reinforcement learning to develop a routing policy from repeated interaction with the environment and the outcomes of previous decisions rather than relying solely on a predefined allocation rule. This is the role the orchestration layer plays in the system, and its formal basis is defined in Section 6.3.

6.2.3 IoS as an External Ecosystem

The IoS device in Fig. 6-1, a co-deployed smart speaker referred to here as a Sound Thing, is not a node in the IoM tier hierarchy and does not participate directly in the Perception/Network/Application structure. It connects to the orchestration layer through a single task-profile-and-request link, and any task it originates is, from that point on, handled like a native IoM task, assigned to a tier node, subject to the same constraints, scored by the same reward, with the result returned to the IoS device over the same link. In this context and initial setup, the IoS device performs no processing itself; it is a sensing and delivery endpoint, not a compute participant in the IoM hierarchy.

To visualise how the orchestration layer could interact with other ecosystems and Things under the wider IoT umbrella, as previously established in Fig. 2-1, the IoX is shown alongside the IoS in Fig. 6-1 as a general representation of this integration point. The IoM-IoS is one instance of this broader IoX vision, and the orchestration layer's connection to it is structured accordingly, as an external link to a co-deployed ecosystem.

6.3 Problem Formulation

This section formalises the orchestration problem introduced in Section 6.2. Each subsection defines one part of the formulation realised by the orchestration layer in Fig. 6-1. The formulation separates the orchestration process into the following components:

- **Observed state:** the current task characteristics together with dynamic node and network conditions available to the Double DQN at each routing decision.
- **Context mode:** a deterministic representation of the task's operational urgency, used to determine the corresponding deadline and reward priorities rather than the destination node itself.
- **Feasibility constraints:** hard modality-hardware and stakeholder-access rules that determine which processing nodes are permissible before learning-based action selection takes place.
- **Learned routing decision:** the Double DQN selects between the feasible processing nodes, or the defer action, according to the current task and system state.
- **Outcome evaluation:** the latency, deadline satisfaction, resource efficiency, and packet-loss consequences of the selected action are combined through the mode-conditioned reward to provide the learning signal used during training.

The following subsections define each of these components and their relationship within the orchestration formulation.

6.3.1 Task Profiles

A task profile is the load that the orchestration layer actually routes, it fixes a task's ecosystem origin, modality, operation type, and resource demands. Six representative task profiles are used in this work. The purpose of these profiles is to show the diversity of modality, ecosystem origin, and urgency that the orchestration layer must jointly handle within a single representative scenario, rather than testing the formulation against a narrow or homogeneous task set. They are defined along the following dimensions:

- **Ecosystem** - whether the task is native to the IoM or externally originated from the IoS.
- **Modality** - visual or audio, determining which nodes are hardware-eligible via ϕ.
- **Operation** - the type of processing performed (detection, tracking, scoring, classification, or reporting).
- **Trigger(s)** - which context mode(s) the profile can be invoked under.
- **Compute and latency demand** - the resource and timing requirements the profile places on its assigned node.

Table 6-1 specifies the six profiles spanning both ecosystems along these dimensions, three per ecosystem, together covering all three context modes.

Table 6-1: Task Profile Categories and Characteristics

Task	Ecosystem	Modality	Operation	Trigger(s)	Compute	Latency
Expression Detection	IoM	Visual	Detection	Live + Ambient	Medium	Stringent
Lip Movement Tracking	IoM	Visual	Tracking	Live	Medium–High	Stringent
Session Summary Report	IoM	Visual	Reporting	Background	High	Tolerant
Pitch Scoring	IoS	Audio	Scoring	Live + Ambient	Medium	Stringent
Speech Clarity Classification	IoS	Audio	Classification	Live	Medium–High	Stringent
Vocal Trend Report	IoS	Audio	Reporting	Background	High	Tolerant

Stringent and Tolerant latency map onto the deadlines defined in Section 6.3.6: Stringent tasks carry the 200 ms (LI) or 1000 ms (AM) budget, Tolerant tasks the 30 s (BB) budget. Compute and bandwidth (Low/Medium/High) are qualitative descriptors of relative demand across the six profiles rather than a hard resource constraint modelled in this work, the only hard feasibility constraint enforced here is modality-hardware compatibility, compute and bandwidth inform the relative characterisation of each profile.

The metrics used to define these profiles are based on existing related application. Pitch Scoring and Speech Clarity Classification are grounded in prior work on classical pitch-tracking methods and measured 5G-IoMusT link figures for the former [145],

and compact audio-classification CNN architectures such as DS-CNN for the latter [229; 230]. The remaining profiles' compute and bandwidth levels are scaled relative to these two, reflecting the added cost of continuous tracking over single-frame detection, and of full-session aggregation over per-instance inference.

6.3.2 Representative Use Case

The task profiles above are intended to generalise across the diverse accessible use cases envisioned by the IoM and IoS, rather than being specific to any one application. To ground their use concretely, this chapter instantiates and evaluates the formulation through one representative scenario: a home-based vocal rehabilitation session, a setting in which voice-assisted technology for home speech and voice therapy is an active and clinically motivated research direction [228]. The scenario is deliberately chosen to exercise both ecosystems together rather than in isolation since visual and vocal cues are complementary evidence of the same recovery process, and the session's structure naturally produces the mix of live, low-urgency, and post-session processing the orchestration layer must handle. Concretely, one session proceeds through three stages:

- **Active exercise repetitions**, during which an IoM smart mirror monitors the user's facial expression and lip movement while a co-deployed IoS smart speaker delivers a structured vocal exercise and captures the user's spoken response, both requiring immediate feedback to sustain the exercise loop.
- **Inter-repetition gaps**, during which the same sensing pipelines continue at a reduced rate, confirming the user is still present and audible rather than actively scoring an exercise.
- **Session close**, at which the accumulated visual and vocal data are compiled into summary reports for later clinician review, with no live user present.

Table 6-2 maps each stage's constituent steps onto the task profiles defined above.

6.3.3 Modality-Hardware Capability and State Space

The IoM's existing tier hierarchy already represents heterogeneity in compute capacity. Accommodating externally originated IoS tasks introduces a second, structurally distinct axis: some nodes are physically incapable of processing a given modality regardless of available compute. A tier-level rule excludes Consumer nodes from audio processing entirely, reflecting a compute-ceiling limitation rather than an audio-specific one. Within the Professional and Hub tiers, audio-processing capability is instead provisioned per node, drawn independently at topology instantiation. The state therefore represents the information available to the Double DQN when making an allocation

Table 6-2: Use-Case Pipeline Mapping

Pipeline step	**Task profile invoked**
Mirror watches the user's face during the exercise	Expression Detection (LI)
Mirror checks mouth shape against the target sound	Lip Movement Tracking
Speaker compares spoken pitch to the target	Pitch Scoring (LI)
Speaker judges whether speech came out clearly	Speech Clarity Classification
Between repetitions: confirms user still present/audible	Expression Detection + Pitch Scoring (AM, low-rate)
After the session trends compiled for clinician review	Session Summary Report + Vocal Trend Report

decision. It combines dynamic system conditions with the characteristics of the current task, allowing the learned policy to distinguish not only between different levels of system load, but also between tasks originating from different ecosystems and requiring different processing modalities:

$$\phi(n, \text{vis}) = 1, \quad \forall n \in N \tag{6-1}$$

$$\phi(n, \text{aud}) = 0, \quad \forall n \in C \tag{6-2}$$

$$\phi(n, \text{aud}) \sim \text{Bernoulli}(p_{aud}), \quad \forall n \in P \cup H \tag{6-3}$$

where N is the set of all IoM processing nodes, comprising the Consumer (C), Professional (P), and Hub (H) tiers; $n \in N$ denotes an individual processing node; and M is the set of task modalities, with vis and aud denoting visual and audio modalities, respectively. The function $\phi : N \times M \to \{0, 1\}$ indicates whether node n has the hardware processing capability required for modality m. Visual processing is assumed to be supported across all three IoM tiers, whereas Consumer nodes are excluded from audio processing under the modelled capability assumptions. For Professional and Hub nodes, audio-processing capability is assigned according to Bernoulli(p_{aud}), where p_{aud} is the probability that an individual node is provisioned with the required audio-processing capability.

The provisioning probability p_{aud} is not fixed across the complete evaluation. It is varied across evaluation runs in Section 6.4 to examine how task completion and defer behaviour change as audio-processing capability becomes increasingly scarce across the Professional and Hub tiers. This provides the experimental mechanism through which the effect of cross-ecosystem modality heterogeneity is evaluated.

The state observed by the orchestration layer at each decision step combines this node-

level information with system- and task-level information:

$$s_t = \big\langle \{u(n)\}, \{q(n)\}, \{b(w)\}, \{R(w)\}, \\ \{\phi(n, \text{aud})\}_{n \in P \cup H}, m(t), e(t), \chi(t), \text{mode}(t) \big\rangle \tag{6-4}$$

$u(n)$ and $q(n)$ give per-node utilisation and queue depth; $b(w)$ and $R(w)$ give available bandwidth and reliability per link. Ecosystem origin $e(t)$ and modality $m(t)$ are carried explicitly, rather than folded into a single task-type label, because a node's eligibility for an externally originated task depends on both – not on system load alone.

where s_t is the system state observed at decision step t; $u(n)$ and $q(n)$ denote the utilisation and queue depth of node n, respectively; $b(w)$ and $R(w)$ denote the currently available bandwidth and reliability of network link w; and $\phi(n, \text{aud})$ indicates whether Professional- or Hub-tier node n has audio-processing capability. For the task currently being considered, $m(t)$ denotes its modality, $e(t)$ its ecosystem of origin, $\chi(t)$ its trigger type, and mode(t) its associated context mode.

The state therefore combines two forms of information available at each task-allocation decision. The node utilisation, queue depth, bandwidth, reliability, and provisioned modality capabilities describe the current operating condition of the IoM infrastructure and may change between successive task arrivals. In contrast, $m(t)$, $e(t)$, $\chi(t)$, and mode(t) describe the task currently being considered. The Double DQN consequently receives both the current system condition and the characteristics of the task for which a routing decision is required, rather than learning from infrastructure state alone. Ecosystem origin $e(t)$ and modality $m(t)$ are carried explicitly rather than folded into a single task-type label because the feasibility and suitability of a routing decision depend on both, instead of system load alone.

6.3.4 Context Mode Determination

Generic IoT offloading formulations typically derive task urgency from system-state signals such as queue depth or resource load, signals that describe how busy the system is, not why a task is running or what the consequence of a missed deadline would be. In a cross-ecosystem and multimodal setting of IoM-IoS, the same computational task can carry very different urgency depending on the point in a session at which it occurs. A task that must resolve within a live user interaction is not commensurable with the same kind of task compiled once, after that interaction has ended. Task urgency is therefore derived from application-domain context rather than system state.

The three context modes introduced here follow from two properties of a session that jointly determine urgency, whether a live user interaction is currently active, and, if so, how time-sensitive the task's outcome is. Table 6-3 summarises the resulting structure. Each mode corresponds to a genuinely distinct operational phase of a live-interaction session, instantiated concretely for the vocal rehabilitation scenario in Table 6-2.

Table 6-3: Context Modes

Mode	Session active	Urgency	Example tasks
LI	Yes	High	Lip tracking, speech clarity
AM	Yes	Low	Low-rate expression/pitch checks
BB	No	Low	Session/vocal trend reports

Rather than being inferred by a learned classifier, the context mode is computed deterministically from the task's trigger type:

$$\alpha(t) = 1[\chi(t) \in \{\text{live, ambient}\}] \tag{6-5}$$

$$\text{mode}(t) = \begin{cases} \text{LI} & \text{if } \alpha(t) = 1 \wedge \chi(t) = \text{live} \\ \text{AM} & \text{if } \alpha(t) = 1 \wedge \chi(t) = \text{ambient} \\ \text{BB} & \text{if } \alpha(t) = 0 \end{cases} \tag{6-6}$$

where $\alpha(t)$ is a binary indicator of whether task t is generated during an active session; $\chi(t)$ is the trigger type associated with task t; and $1[\cdot]$ is the indicator function, equal to one when the stated condition is true and zero otherwise. mode(t) is the resulting context mode assigned to the task, taking one of three values: LI, AM, or BB.

Because $\alpha(t)$ is itself a function of the already-observed $\chi(t)$, mode(t) is fully determined once a task's trigger is known, leaving no residual ambiguity for the learning agent to infer.

The context mode does not itself determine the destination node. Its role is to represent the operational urgency associated with the current task and to select the corresponding deadline and reward priorities used later in the formulation. Node selection remains the responsibility of the Double DQN after feasibility constraints have been applied.

6.3.5 Action Space and Constraint Masking

Existing offloading formulations typically treat data modality as a soft factor shaping routing quality rather than a hard feasibility constraint. This does not extend to a cross-ecosystem setting, where an externally originated task's eligible nodes are limited by hardware capability that no amount of available compute can substitute for. At each decision step, the orchestration layer selects a specific processing node, or defers:

$$\mathcal{A} = N \cup \{\text{defer}\} \tag{6-7}$$

where $\mathcal{A}$ is the complete action space available to the orchestration mechanism, N is the set of IoM processing nodes, and defer represents the action of postponing the allocation of task t rather than immediately assigning it to a processing node. The complete action space is subsequently reduced to the feasible subset of $\mathcal{A}$ at state s_t:

modality-hardware eligibility, and stakeholder-tier access:

$$N_m(t) = \{n \in N : \phi(n, m(t)) = 1\} \tag{6-8}$$

$$N_{acc}(t) = \{n \in N : \text{tier}(n) \in \text{Access}(\text{tier}(\sigma_r(t)))\} \tag{6-9}$$

where $N_m(t)$ is the set of processing nodes capable of supporting the modality $m(t)$ required by task t, as determined by the capability function $\phi(n, m(t))$. $N_{acc}(t)$ is the set of nodes permitted by the stakeholder-tier access constraint; $\text{tier}(n)$ identifies the IoM tier to which candidate node n belongs; $\sigma_r(t)$ is the IoM entry node associated with task t; and $\text{Access}(\cdot)$ defines the destination tiers that may be accessed from the tier containing that entry node. The first constraint therefore represents physical modality-processing feasibility, while the second preserves the stakeholder-oriented access structure of the IoM hierarchy.

The feasible action set is their intersection, with defer unioned in unconditionally so that $\mathcal{A}(s_t)$ is never empty:

$$\mathcal{A}(s_t) = \big(N_m(t) \cap N_{acc}(t)\big) \cup \{\text{defer}\} \tag{6-10}$$

where $\mathcal{A}(s_t)$ is the feasible action set in state s_t, formed by the intersection of the modality-compatible node set $N_m(t)$ and the stakeholder-accessible node set $N_{acc}(t)$, with the defer action included unconditionally so that $\mathcal{A}(s_t)$ is never empty.

$\sigma_r(t)$ is the entry node, the requesting IoM node for native tasks, or the co-deployed entry node for externally originated IoS tasks. Ecosystem origin $e(t)$ does not appear explicitly in the constraint equations, cross-ecosystem handling is concentrated entirely in how $\sigma_r(t)$ is set, so the same constraint structure applies regardless of a task's origin. This formulation is a state-wise constrained decision process, constraints are enforced by masking the action set directly, guaranteeing zero infeasible routing at every step, rather than through an expectation-budget constraint as in a classical constrained MDP.

The masking operation is applied before action selection. The Double DQN therefore does not learn whether a node is physically or architecturally permissible for a task; those conditions are enforced directly through $\mathcal{A}(s_t)$. Learning is restricted to selecting between the feasible nodes remaining after masking, together with the defer action. This separates non-negotiable system constraints from the routing preferences learned from experience.

6.3.6 Packet Loss and End-to-End Latency

Packet loss and latency are tracked explicitly because they are the primary channel through which the added IoS-IoM link, and the cross-ecosystem load it carries, are felt by the rest of the system. Packet loss on a link is derived from its reliability and its

current congestion:

$$U_{\text{link}}(w,t) = \min\left(\frac{Load(w)}{B(w)}, 1\right) \tag{6-11}$$

$$PL(w,t) = 1 - R(w)\cdot(1 - U_{\text{link}}(w,t))^{\xi} \tag{6-12}$$

$$PL(t) = 1 - \prod_{w\in \text{Path}(t)} (1 - PL(w,t)) \tag{6-13}$$

where $U_{\text{link}}(w,t)$ is the utilisation of network link w while handling task t; $Load(w)$ is the current traffic load carried by link w; $B(w)$ is the bandwidth capacity of the link; $R(w)$ is its reliability; and ξ is the congestion-sensitivity exponent used in the packet-loss model. $PL(w,t)$ denotes the packet-loss probability for task t on link w, while $PL(t)$ is the resulting end-to-end packet-loss probability across the links forming Path(t), the end-to-end service path followed by the task.

$U_{\text{link}}(w,t)$ is clamped to $[0,1]$ so that an overloaded link does not invert the loss calculation.

End-to-end latency is measured to delivery at the user-facing endpoint, over a path from source, to the processing node, to the delivery endpoint:

$$\begin{aligned}\text{Path}(t) = {} & [Src(t) \to \sigma_r(t)] \oplus [\sigma_r(t) \to n(t)] \\ & \oplus [n(t) \to Dst(t)]\end{aligned} \tag{6-14}$$

$$\begin{aligned}\ell(t) = {} & \ell_{uplink}(t) + \ell_{route}(t) \\ & + \ell_{proc}(t) + \ell_{downlink}(t)\end{aligned} \tag{6-15}$$

where Path(t) is the complete end-to-end service path followed by task t; $Src(t)$ and $Dst(t)$ are its source and delivery destination, respectively; $\sigma_r(t)$ is the IoM entry node; $n(t)$ is the processing node assigned to the task; and $\oplus$ denotes the concatenation of successive path segments. The end-to-end latency $\ell(t)$ comprises the uplink latency $\ell_{uplink}(t)$, routing latency $\ell_{route}(t)$, processing latency $\ell_{proc}(t)$, and downlink latency $\ell_{downlink}(t)$.

For a native IoM task, $Src(t) = Dst(t) = \sigma_r(t)$; for an IoS-originated task, both endpoints are the IoS device, with $\sigma_r(t)$ the co-deployed entry node.

Each mode carries a deadline $D(t)$ reflecting the differing consequence of delay under each. Live-Interactive tasks are held to a 200 ms end-to-end budget. The ITU-T G.114 recommendation identifies approximately 150 ms one-way delay as an important reference for interactive real-time communication [231], corresponding to approximately 300 ms for a symmetric round trip. The 200 ms budget adopted here is deliberately more stringent, reflecting the tighter responsiveness targeted for the closed feedback loop between a user's live action and the system's response rather than directly adopting a voice-communication threshold. Ambient Monitoring's 1000 ms budget and Background-Batch's 30 s budget are set successively looser in accordance with

the reduced urgency of these modes.

6.3.7 Reward Function

Task urgency, once derived from context mode, must also shape how outcomes are scored: a single fixed weighting cannot represent the differing priorities of a live interaction versus a batch process. The reward is therefore composed of four signed components and conditioned on context mode:

$$R_{lat}(t) = -\min\left(\frac{\ell(t)}{D(t)}, C_{lat}\right) \tag{6-16}$$

$$R_{dead}(t) = -\psi(\text{mode}(t)) \cdot 1[\ell(t) > D(t)] \tag{6-17}$$

$$R_{eff}(t) = 1 - \frac{\text{rank}(n(t)) - 1}{2} \tag{6-18}$$

$$\begin{aligned} R_{loss}(t) = & -\lambda_{loss} \cdot \max(0, PL(t) - \theta_{m(t)}) \\ & \cdot 1[\chi(t) \in \{\text{live, ambient}\}] \end{aligned} \tag{6-19}$$

where $R_{lat}(t)$ is the latency component of the reward; $R_{dead}(t)$ is the deadline-satisfaction component; $R_{eff}(t)$ is the resource-efficiency component; and $R_{loss}(t)$ is the packet-loss component. $\ell(t)$ is the end-to-end latency experienced by task t, $D(t)$ is its mode-dependent deadline, and C_{lat} is the upper clamp applied to the latency-ratio reward term. $\psi(\text{mode}(t))$ is the deadline-miss penalty associated with the task's context mode, and $1[\cdot]$ is the indicator function. $n(t)$ is the processing node assigned to the task and $\text{rank}(n(t))$ denotes its IoM processing-tier rank, where $\text{rank}(C)$=1, $\text{rank}(P)$=2, and $\text{rank}(H)$=3. λ_{loss} is the packet-loss penalty coefficient, $PL(t)$ is the end-to-end packet-loss probability for task t, and $\theta_{m(t)}$ is the packet-loss threshold associated with its modality $m(t)$.

R_{eff} rewards the lowest-capability tier that satisfies the task. R_{lat} is clamped at C_{lat} to bound the penalty for a repeatedly deferred or very late task. R_{loss} is structurally zero for Background-Batch tasks, formalising their exemption from packet-loss penalty.

Because these weights are manually specified, the learned policy is sensitive to their relative magnitudes, changing the weights changes the scalarised trade-off presented to the agent and may therefore alter the routing behaviour learned during training. The present study evaluates whether the selected weights produce the intended mode-specific behaviour, but does not perform an exhaustive reward-weight sensitivity analysis or optimise the weight vectors as part of the learning process. The reported policy should therefore be interpreted as conditional on the selected reward design, with systematic optimisation of these weights representing a direction for further investigation.

The composite reward is piecewise, since a deferred task has no assigned node:

$$r_t = \begin{cases} \sum_k w_k(t) \cdot R_k(t) & \text{if } a_t \in N \\ -\psi_{defer}(\text{mode}(t)) & \text{if } a_t = \text{defer} \end{cases} \tag{6-20}$$

$$w_k(t) = w_k^{(\text{mode}(t))}, \quad w_k^{(i)} \geq 0, \quad \sum_k w_k^{(i)} = 1 \ \ \forall i \tag{6-21}$$

where r_t is the scalar reward obtained at decision step t; a_t is the action selected for task t; $R_k(t)$ denotes reward component k; and $w_k(t)$ is the weight applied to that component for the current task. $\psi_{defer}(\text{mode}(t))$ is the mode-dependent penalty applied when the selected action is defer. The term $w_k^{(i)}$ denotes the weight assigned to reward component k under context mode i, with the component weights constrained to be non-negative and to sum to one for each mode.

A deferred task retains its mode and deadline, so deferral does not exempt it from urgency: ψ_{defer} is itself mode-scaled, and the accumulating wait is reflected in $\ell(t)$ once the task is eventually assigned. Table 6-4 gives the per-mode weight vectors used in this work.

Table 6-4: Per-Mode Reward Weights

Mode	R_{lat}	R_{dead}	R_{eff}	R_{loss}
LI	0.35	0.35	0.10	0.20
AM	0.25	0.20	0.30	0.25
BB	0.10	0.10	0.80	0.00

These weight vectors are a designed prior rather than a derived optimum: the ordering within each mode follows directly from the mode semantics defined in Section 6.3.4. Live-Interactive's tight deadline and closed feedback loop demand that latency and deadline-satisfaction dominate its weighting (0.70 combined), even at the cost of resource efficiency; Background-Batch's lack of real-time urgency instead allows efficiency to dominate, since there is no live feedback loop to protect. The exact magnitudes are tuned rather than derived from a first-principles argument. This follows standard practice in multi-objective RL, where linear scalarisation is the most widely used approach because it avoids the added complexity of multi-policy methods [232], at the cost of only representing the convex portion of the trade-off space those objectives define. Section 6.4's behavioural validation checks that the trained policy's actual behaviour matches each mode's intended weighting.

The reward should therefore be interpreted as feedback on the consequence of a routing decision rather than as an additional routing rule. During training, the observed outcome of each selected action is converted into this scalar learning signal, allowing the Double DQN to update the relative value assigned to feasible actions under similar task and system conditions. The reward weights encode the intended priorities of each context mode, while the routing behaviour that follows from those priorities is learned.

6.3.8 Optimisation Objective

The state, feasible action set, and reward combine into the orchestration layer's optimisation objective:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=1}^{K} r_t\right], \quad \text{s.t. } a_t \in \mathcal{A}(s_t) \ \ \forall t \tag{6-22}$$

where π denotes a task-orchestration policy and π^* the policy that maximises the expected cumulative reward; $\mathbb{E}_{\pi}[\cdot]$ denotes expectation under policy π; K is the number of decision steps in one bounded session; r_t is the reward obtained at decision step t; a_t is the selected action; s_t is the system state observed at that decision step; and $\mathcal{A}(s_t)$ is the corresponding feasible action set.

This states the orchestration layer's objective in full to find the policy π, a mapping from the current state s_t to a choice of action in $\mathcal{A}(s_t)$, that maximises the total reward accumulated over one session's sequence of tasks, in expectation, while never selecting an action outside the feasible set at any step. Because $\mathcal{A}(s_t)$ already excludes any node that cannot process the task's modality or violates access rules, learning affects *which* feasible node is best given context, not whether a chosen node is permitted.

K is the number of decision steps in one bounded session. An undiscounted, finite-horizon return is used, since each episode is a short, bounded session rather than an ongoing process. Both the feasible action set and the reward's weighting change with every task, the former with modality eligibility and the random provisioning of node capability, the latter with context mode, making this a moving target rather than a fixed assignment solvable once at design time. Section 6.4 describes the Double DQN agent trained to solve this objective under these shifting conditions.

6.4 Methodology and Simulation Design

This section describes how the formulation in Section 6.3 is realised and evaluated. This includes the training algorithm used to solve the optimisation objective, the episode and simulation structure it is trained within, the baseline it is compared against, and the design of that comparison.

6.4.1 Training Algorithm

The orchestration problem uses a discrete action space comprising the candidate processing nodes together with the defer action, making a value-based reinforcement-learning method a direct fit for the routing decision. Double DQN was selected over standard DQN because its separation of action selection and target evaluation re-

duces the value-overestimation bias associated with using a single Q-network [182]. Continuous-action methods were not selected because the decision formulated here is a discrete routing choice rather than continuous resource allocation, while multi-agent approaches would represent a different decentralised coordination problem from the single orchestration agent considered in this work.

The orchestration layer is therefore trained using Double DQN, with two departures from a generic DQN loop. Firstly, hard masking is applied before action selection, and the same mask is reused when computing the target. The agent maintains two networks: an online network $Q(s, a; \theta)$, parameterised by weights θ, used both to select actions and to compute the greedy next action a^*; and a target network $Q(s, a; \theta^-)$, a periodically-synchronised copy of θ used only to evaluate that action when forming the training target y. Separating selection through the online network from evaluation through the target network reduces the value overestimation that can arise when the same estimates are used for both operations. Transitions are stored in and sampled from a replay buffer $\mathcal{B}$.

Algorithm 2 summarises the training procedure, where $\gamma \in [0, 1]$ is the discount factor applied to the target value.

Algorithm 2 Double DQN with Action Masking

Initialise online network $Q(s, a; \theta)$, target network $Q(s, a; \theta^-) \leftarrow Q(s, a; \theta)$
Initialise replay buffer $\mathcal{B}$
for each training episode **do**
    Reset environment: redraw $\phi(n, \text{aud})$ (Eq. 3); reset queues and link states
    for each task arrival t **do**
        Observe state s_t
        mode$(t) \leftarrow$ deterministic mode from $\chi(t)$
        $\mathcal{A}(s_t) \leftarrow$ feasible actions via hard masking
        $a_t \leftarrow \varepsilon$-greedy over $Q(s_t, \cdot; \theta)$, restricted to $\mathcal{A}(s_t)$
        Execute a_t; observe r_t and s_{t+1}
        Store $(s_t, a_t, r_t, s_{t+1}, \mathcal{A}(s_{t+1}))$ in $\mathcal{B}$
        Sample mini-batch from $\mathcal{B}$
        $a^* \leftarrow \arg\max_{a \in \mathcal{A}(s_{t+1})} Q(s_{t+1}, a; \theta)$ ▷ online net selects
        $y \leftarrow r_t + \gamma \cdot Q(s_{t+1}, a^*; \theta^-)$ ▷ target net evaluates
        Update θ by gradient descent on $(y - Q(s_t, a_t; \theta))^2$
        Periodically sync $\theta^- \leftarrow \theta$
    end for
end for

Masking the target's max-over-actions matters: without it, the network would be trained to value actions it was never permitted to take, corrupting the learned values for infeasible state-action pairs. For a finite-horizon episode return, γ is set to 1. The agent was trained with ε-greedy exploration, ε decayed exponentially from 1.0 to a 0.05 floor, over 1,200 episodes, using a standard multi-layer perceptron as

the Q-network; hyperparameters were selected via a small sensitivity comparison on a held-out validation seed split, distinct from the seeds used for reported results.

6.4.2 Episode Structure

An episode mirrors one vocal-rehabilitation session, an active exercise phase with inter-repetition gaps, followed by session close and post-session reporting. Task instances are drawn from the six profiles (Section 6.3.2) according to their representative triggers: Live-Interactive tasks during exercise repetitions, Ambient-Monitoring instances during inter-repetition gaps, and Background-Batch tasks once at session close. Episode length is set to 15–20 minutes of simulated session time.

An episode terminates at session close, after Background-Batch tasks are generated and processed, there is no early termination on task failure, since failures are penalised through reward rather than by ending the episode. At reset, queues, node loads, and link states are reinitialised. Audio provisioning $\phi(n, \text{aud})$ is redrawn per episode during training, so the agent does not overfit to a single hardware layout, but held fixed per topology seed during evaluation, so that any performance difference between methods reflects the policy rather than a favourable draw of which nodes support audio.

6.4.3 Simulation Setup

The IoM is simulated as a three-tier network of IoM nodes deployed over a 5×5 km urban area, matching the topology used in Chapter 5, so that results remain comparable with that wider body of work rather than introducing a second, scenario-specific deployment scale. The home-based scenario in this chapter is one entry point into this shared topology: the Consumer-tier mirror and co-deployed IoS speaker are not served by dedicated infrastructure of their own, but route into the same area wide Professional and Hub tiers as any other IoM deployment.

Fig. 6-2 shows the resulting simulation architecture, including the environment's reset and step interface, the constraint and reward modules described in Section 6.3, and the agent-training loop of Algorithm 2. The three stages map directly onto the episode structure above:

- INPUT is built once per episode reset: the topology, the provisioning draw $\phi(n, \text{aud})$, and the task-arrival stream for that session
- PROCESS runs once per task arrival, mirroring `step()`: state and context are assembled, the feasible action set is computed via hard masking, and the Routing Switch dispatches the task to the RL Router (the trained Double DQN agent, Algorithm 2) under the orchestrated condition, or to the Cloud Router under the baseline process.

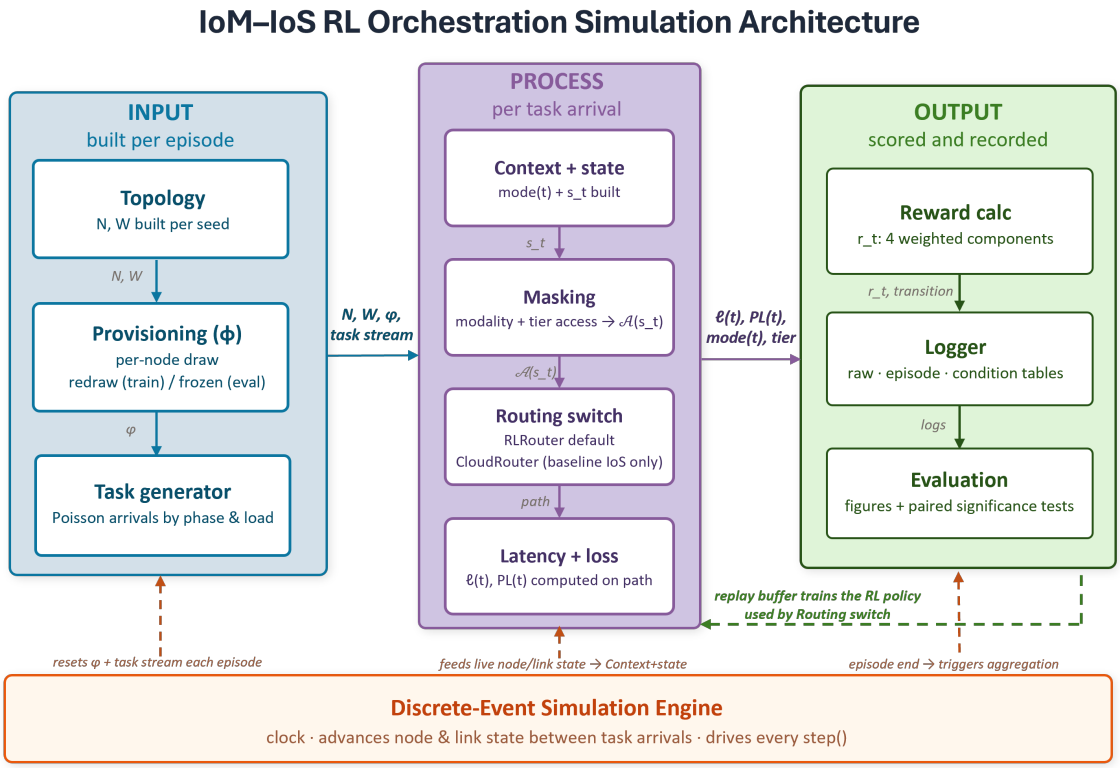


Figure 6-2: Simulation architecture, showing the environment interface, constraint and reward modules, and the agent-training loop.

- OUTPUT scores and logs the outcome of each task; during training, this transition is written to the replay buffer $\mathcal{B}$, which in turn trains the policy used by the Routing Switch, closing the learning loop shown at the bottom of the figure. A discrete-event simulation engine coordinates all three stages, owning the simulated clock and triggering the OUTPUT stage's aggregation at each episode's end.

6.4.4 Baseline

The comparison baseline is a cloud-offload configuration. IoS audio tasks are routed to a remote Google Cloud endpoint rather than into the IoM tier hierarchy, while native IoM visual tasks remain routed by the same trained agent in both conditions. This isolates the comparison to the decision this work is concerned with, whether to orchestrate IoS tasks through the IoM topology or offload them to the cloud.

The network latency segment of the cloud-offload path uses empirically measured round-trip latency from an actual Google Cloud deployment accessed over Wi-Fi, following the same edge-cloud measurement methodology established in 4.3. Both the orchestrated and cloud-offload paths are measured to the same user-delivery endpoint, and both face an identical task-arrival sequence within a given condition, since the task-generator seed is shared across methods rather than drawn independently, making this a paired comparison rather than two independent runs.

6.4.5 Evaluation Design

The evaluation proceeds in three stages. This is by first confirming the trained policy behaves as designed, then comparing it against the cloud-offload baseline, and finally stress-testing it on the axis with no baseline analogue.

Validation. Before any baseline comparison, the trained policy is checked against the behaviour the formulation intends by assessing the assigned-tier distribution by task profile, deadline-satisfaction and defer-rate by context mode as load increases.

Baseline Comparison Load in the form of task arrival rate is evaluated across low, medium, and high conditions, defined relative to the IoM topology's empirically calibrated saturation point ($\lambda_{rel} = 30\%, 60\%, 85\%$) rather than as arbitrary task counts. Audio pipeline provisioning is held fixed at $p_{aud} = 1.0$ so this stage carries no modality-scarcity confound. This is the primary comparison against the cloud-offload baseline, using task completion rate, end-to-end latency, and packet-loss rate as key metrics, each additionally split by ecosystem origin.

Modality-Heterogeneity Sensitivity. Audio-provisioning probability p_{aud} is swept from 10% to 100% in 10% increments, with load held fixed, tracking overall, native-IoM, and IoS completion rate alongside defer rate as p_{aud} varies. This stage directly tests adaptation to data-modality heterogeneity.

6.5 Results and Discussion

6.5.1 Validation

Before comparing the trained policy against the cloud-offload baseline, its intended behaviour is first validated.

Fig. 6-3 shows the assigned-tier distribution by task profile under the orchestrated condition at medium load, grouped by ecosystem origin. The two groups reveal the same underlying logic applied consistently across two very different task populations. Within native IoM tasks, routing tracks urgency. Session Summary Report, a background reporting task, lands on the Consumer tier the majority of the time, while the two live-interactive profiles, Expression Detection and Lip Movement Tracking, are pushed toward Professional and Hub nodes instead. Within IoS tasks, the pattern is starker, every audio-gated profile is entirely absent from the Consumer column, and each one clears Professional or Hub in over ninety-eight percent of instances. This is a stronger result than urgency-based routing alone, it demonstrates that the constraint and reward structure generalise across ecosystems without any ecosystem-specific routing rule. The policy did not need to be told that audio tasks behave differently from visual ones, the modality-hardware constraint alone was sufficient to produce that separation, and the mode-conditioned reward alone was sufficient to produce urgency-appropriate placement within each ecosystem.

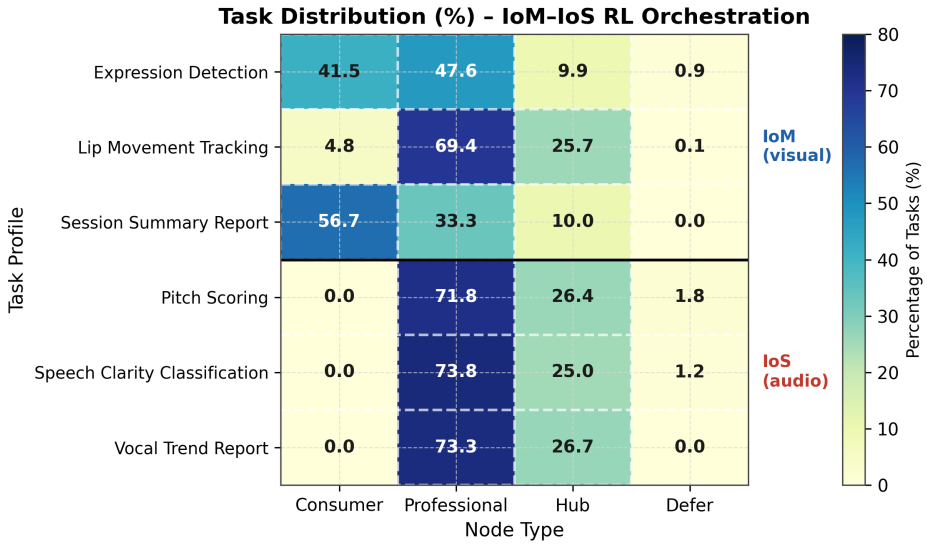


Figure 6-3: Assigned-tier distribution by task profile under the orchestrated condition at medium load, grouped by ecosystem of origin.

The same separation holds at the level of outcomes. Background-Batch tasks never miss their deadline or defer, regardless of load. Ambient-Monitoring degrades only mildly. Live-Interactive is the only mode under real pressure, deadline satisfaction drops from 0.992 to 0.933 and defer rate rises more than fivefold between low and high load, confirming that the per-mode weighting translates into intended behaviour rather than remaining an untested prior e.g. LI's combined 0.70 weight on latency and deadline is exactly the mode under strain, BB's 0.80 weight on efficiency is exactly the mode that never strains. This sets the terms for what follows, a favourable baseline comparison is only meaningful once the policy has shown it treats tasks differently according to need.

6.5.2 Load vs. Baseline

The trained policy is now compared against the cloud-offload baseline across low, medium, and high load, holding audio provisioning fixed so the comparison isolates load alone.

Fig. 6-4 reports completion rate split by ecosystem origin. At low load, orchestration achieves a higher completion rate (99.2% vs. 95.3%, $p < 0.001$) while native-IoM is statistically indistinguishable between conditions (99.5% vs. 99.4%, $p = 0.815$), with the topology nowhere near saturated, taking on the IoS workload costs the native system nothing. As load rises, native-IoM completion becomes significantly worse under orchestration at both medium (89.6% vs. 95.3%, $p < 0.001$) and high load (87.3% vs. 92.5%, $p < 0.001$), since native tasks are routed identically in both conditions and the difference is congestion, not routing quality. IoS completion remains significantly higher under orchestration at every load (medium: +2.7 pp, $p = 0.018$; high: +8.7 pp, $p < 0.001$), the gap widening as load increases. Orchestration is buying IoS performance by

spending native-task headroom, a trade worth making here since the vocal exercise IoS is scoring and the facial expression IoM is tracking are evidence of the same repetition, not two competing workloads.

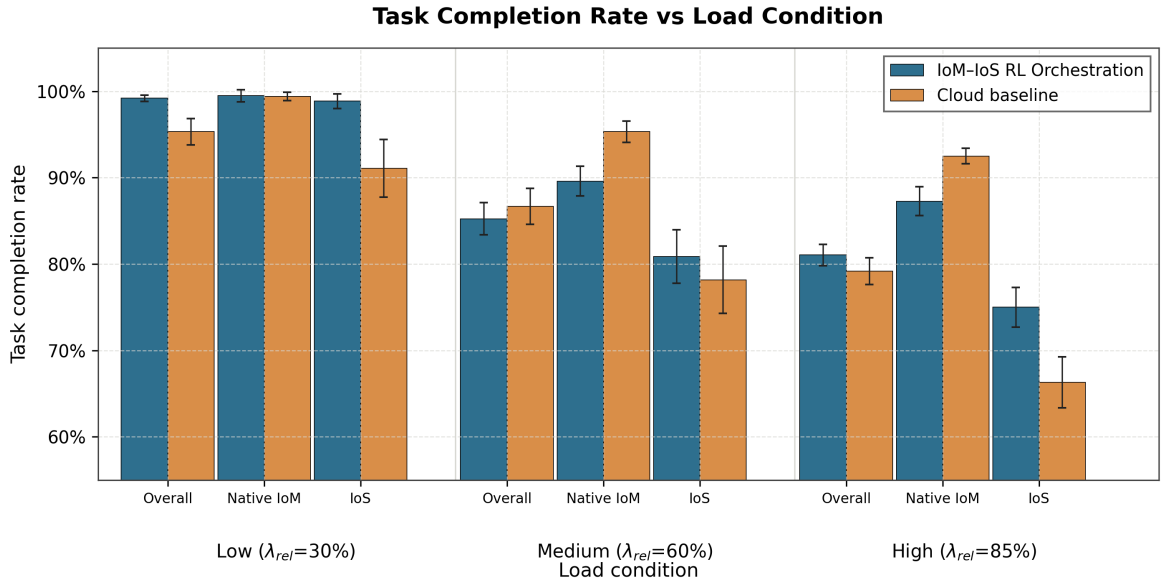


Figure 6-4: Task completion rate versus load condition, orchestrated system compared against the cloud-offload baseline.

This trade-off produces a pooled effect that is non-monotonic in load and worth stating explicitly, overall completion favours orchestration at low load (+3.9 pp), reverses in cloud's favour at medium load (−1.4 pp, $p = 0.037$), then reverses back at high load (+1.9 pp, $p = 0.033$). This follows directly from the two component trends growing at different rates, between medium and high load the IoS advantage more than triples (2.7 to 8.7 pp) while the native disadvantage stays roughly flat (−5.7 to −5.2 pp), so once the IoS gain outweighs the native loss the pooled average swings back even though native performance has not recovered. The overall metric is a weighted composite of two ecosystems moving in opposite directions.

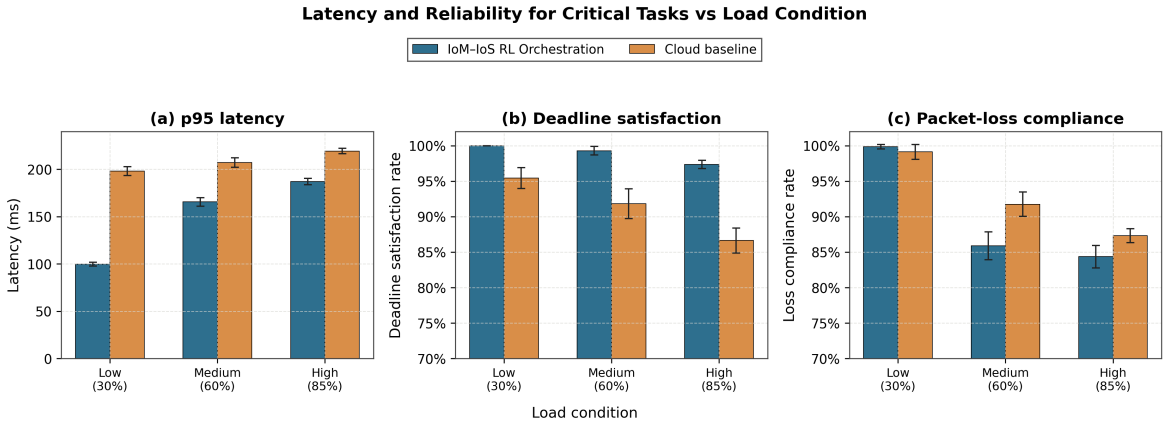


Figure 6-5: Latency and reliability for Live-Interactive tasks versus load condition: p95 latency, deadline satisfaction, and packet-loss compliance.

Fig. 6-5 narrows the comparison to Live-Interactive tasks. Orchestration achieves lower

p95 latency and higher deadline satisfaction than the cloud-offload baseline across all three load conditions. As load increases, p95 latency rises under both approaches, although orchestration retains a latency advantage. Deadline satisfaction also declines with increasing load for both approaches, but the reduction is more pronounced for the cloud baseline, resulting in a progressively clearer advantage for orchestration under higher load.

Packet-loss compliance exhibits a different trend. Orchestration performs marginally better at low load but falls below the cloud baseline at medium and high load. This reflects the additional contention experienced when IoS traffic is retained within the shared IoM topology as system load increases, whereas cloud offloading provides greater isolation from local network congestion at the cost of the additional remote communication path. The results therefore demonstrate a trade-off that cross-ecosystem orchestration provides lower latency and stronger deadline satisfaction for Live-Interactive tasks, while increased local contention can reduce packet-loss compliance under heavier load.

6.5.3 Modality-Heterogeneity Sensitivity

The final evaluation scales how many nodes can physically process audio, at fixed medium load, with no cloud-offload baseline, since a cloud path does not face a hardware-availability constraint of this kind.

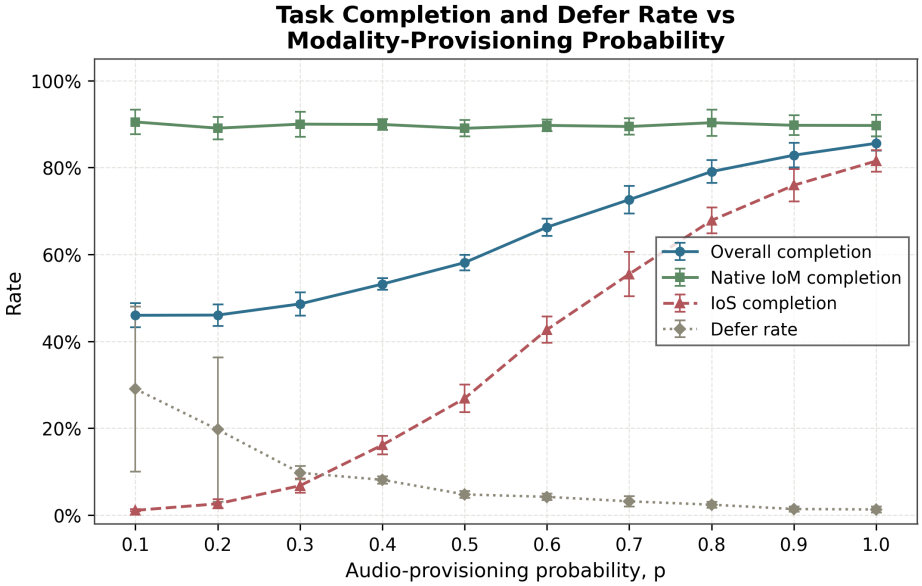


Figure 6-6: Task completion and defer rate versus audio-provisioning probability, at fixed medium load.

Fig. 6-6 shows IoS completion rising monotonically from 1.1% at $p_{aud} = 0.1$ to 81.5% at $p_{aud} = 1.0$, while native-IoM completion stays flat throughout (89.0–90.3%) and overall completion rises correspondingly, from 46.0% to 85.6%. This is the modality-hardware constraint doing exactly what it was built to do: IoS tasks are gated by audio capability and native tasks are not, so scarcity on one side does not infringe on the

other. Defer rate falls smoothly across the same range, from 29.1% at $p_{aud} = 0.1$ to 1.3% at $p_{aud} = 1.0$. A hard constraint could easily have produced a system that simply fails, or silently misroutes, whenever the hardware it depends on is scarce. Instead, the policy defers tasks it genuinely cannot place, and does so proportionally less often as the constraint binds less frequently, this is the reward treating a deferred task as an acceptable outcome rather than a failure, exactly as the mode-conditioned reward in Section 6.3.7 was designed to do.

One caveat applies at the scarce end of the sweep: defer-rate variance is high at $p_{aud} = 0.1$ (SD=0.194 against a mean of 0.291), because provisioning at that point is bimodal across seeds, some topology draws end up with no audio-capable nodes at all, others with a few. This reflects two different operating regimes, not measurement noise, and is a property of the provisioning process rather than a weakness in the constraint itself.

6.6 Summary

This chapter addressed Q_4 and O_4 by extending IoM task coordination beyond its native computational hierarchy to an ecosystem-to-ecosystem interaction with the IoS. Building on the context-aware distribution developed in Chapter 5, the orchestration problem was expanded to account for task characteristics that arise when different IoT ecosystems interact, including ecosystem origin, sensing modality, modality-specific hardware capability, application urgency, and the availability of appropriate processing resources.

An Intelligent Orchestration Layer was introduced as a control plane through which native IoM and externally originated IoS tasks can be coordinated across the existing Consumer, Professional, and Hub hierarchy. The formulation combines modality-hardware and stakeholder-access constraints with three application-context modes representing Live-Interactive, Ambient-Monitoring, and Background-Batch operation. These modes determine the relative importance of latency, deadline satisfaction, resource efficiency, and packet-loss behaviour within a mode-conditioned reward. Hard action masking ensures that the learned policy operates only over feasible execution nodes, while Double DQN enables selection between those resources according to the current task and system state.

The formulation was instantiated through a representative home-based vocal rehabilitation scenario in which visual IoM and audio IoS tasks contribute complementary information to the same user interaction. The trained policy demonstrated the intended distinction between task types, excluding audio workloads from nodes without the required processing capability while assigning tasks differently according to their urgency. Live-Interactive workloads experienced the greatest pressure as system load increased, whereas Background-Batch tasks retained their substantially looser timing behaviour, demonstrating that the context modes affected the learned allocation policy

rather than acting only as descriptive task labels.

Comparison with the cloud-offload baseline further demonstrated the trade-off introduced by cross-ecosystem orchestration. Routing IoS workloads through the IoM hierarchy provided higher IoS task completion across the evaluated load conditions and lower p95 latency together with higher deadline satisfaction for Live-Interactive tasks. These benefits were accompanied by increased contention within the shared IoM topology as load increased, reducing native-IoM completion relative to the cloud condition at medium and high load and producing lower packet-loss compliance under the more congested conditions. Cross-ecosystem orchestration should therefore not be interpreted as universally preferable to remote offloading. Rather, the results demonstrate that sharing IoM computational resources can improve latency-sensitive external services while introducing a measurable resource-sharing trade-off for the host ecosystem.

The modality-heterogeneity evaluation additionally showed that the orchestration framework responds predictably when compatible hardware becomes scarce. IoS completion increased as the proportion of audio-capable Professional and Hub nodes increased, while native-IoM completion remained comparatively stable. Where an audio task could not be feasibly assigned, the action-masking and defer mechanism prevented incompatible routing rather than forcing execution on an unsuitable node. This demonstrates that interoperability between IoT ecosystems requires consideration not only of computational capacity and network conditions, but also of the specialised capabilities associated with different sensing and processing modalities.

Collectively, the findings extend the IoM from a distributed ecosystem capable of coordinating its own heterogeneous applications to an architecture capable of participating in broader cross-ecosystem task orchestration. Chapters 4 and 5 established the importance of computational placement and context-aware distribution within the IoM, while the present chapter demonstrates how this decision process can incorporate external ecosystem origin, multimodal requirements, application context, and learned adaptation. Although instantiated here through the IoS and a representative multimodal rehabilitation scenario, this evaluation provides an initial operational demonstration of the broader IoX vision established in Chapter 2. By enabling an external ecosystem to access the IoM's distributed computational hierarchy while preserving modality-specific feasibility and the operation of native IoM services, the framework demonstrates how the IoM can begin to function as an interoperable participant within a wider landscape of specialised IoT ecosystems. The resulting progression provides an initial operational basis for the wider IoX vision, in which specialised IoT ecosystems can retain their individual roles while sharing computational capabilities to support services that span ecosystem boundaries.

Chapter 7

Conclusion and Future Work

This chapter concludes the thesis by synthesising the principal findings in relation to the research problems and questions established at the outset of the work. Section 7.1 summarises how the challenges identified for the IoM were addressed through the contributions of the thesis, before Section 7.2 outlines directions for extending the work towards larger-scale deployment, broader application validation, cross-ecosystem interoperability, and secure operation.

7.1 Reflection on the Research Objectives

This thesis investigated the conceptual, architectural, and operational foundations required to establish the IoM as an interconnected IoT ecosystem capable of supporting personalised and assistive services across diverse users, applications, and connected environments. The research progressed from defining the IoM and the requirements associated with its interconnection to evaluating application execution across distributed infrastructure, developing mechanisms for context-aware task distribution, and extending computational coordination beyond the IoM towards interaction with an external IoT ecosystem. Collectively, these contributions move the IoM from a prospective architectural concept towards an experimentally and computationally evaluated distributed ecosystem.

The starting challenge concerned the absence of an architectural foundation through which independently deployed smart mirrors could operate as part of a coherent interconnected ecosystem. The literature review identified that existing smart mirror systems were predominantly developed as standalone or application-specific platforms, providing limited support for interaction between users, service providers, computational resources, and connected environments. This motivated Q_1, which asked what requirements and architectural considerations are necessary to establish the IoM as an interconnected ecosystem. The thesis addressed this problem by defining the IoM around a stakeholder-oriented Consumer, Professional, and Hub hierarchy together with Perception, Network, and Application functional layers that describe how sens-

ing, communication, computation, and service delivery interact across the ecosystem. The architectural investigation additionally considered accessibility and inclusive operation to account for differences in user interaction and assistance requirements. This established the conceptual and architectural foundation required for the IoM to operate as more than a collection of independent smart mirror devices and provided the system structure used throughout the subsequent contributions of the thesis.

Establishing an interconnected architecture introduced the subsequent problem of determining how application computation should be distributed across resources with substantially different capabilities and communication paths. The placement of computation can affect processing latency, network transfer, resource demand, and responsiveness, yet these effects had not been characterised for the IoM hierarchy. This motivated Q_2, which examined the performance implications, computational bottlenecks, and computation–communication trade-offs associated with distributed IoM infrastructure. A representative dental smile-analysis application was first developed and integrated within a smart mirror to establish an end-to-end assistive IoM workload. Its execution was then evaluated between local edge and remote cloud resources before being extended to a physical Consumer-Professional-Hub testbed under different network conditions and concurrent demand. The results demonstrated that execution placement affects the complete processing and communication path of an IoM service. Local execution reduces dependence on network transfer but is constrained by device capability and contention, while higher-tier execution can reduce lower-tier computational demand at the cost of additional communication. The suitability of each placement therefore changed with workload characteristics, resource capability, network condition, payload size, and concurrent demand, establishing computational placement in the IoM as a context-dependent computation-communication problem rather than a fixed architectural choice.

The context dependence observed in the physical evaluations introduced a further challenge as the IoM scales to support multiple applications and users simultaneously. A fixed placement policy cannot readily accommodate workloads with different computational, memory, communication, latency, and data-handling requirements while the availability and utilisation of candidate resources also change during operation. This motivated Q_3, which considered how diverse and concurrent computational demands can be accommodated across the heterogeneous resources of the IoM. The CATD framework was developed to address this problem by combining tier-based resource assessment with evaluation of residual computational capability, network quality, geographic proximity, and data sensitivity. Its evaluation across heterogeneous workload compositions, increasing system demand, infrastructure failures, bandwidth degradation, and changing operational conditions demonstrated that integrated contextual assessment provides more resilient task allocation than the evaluated single-factor strategies. The results therefore showed that accommodating application diversity within the IoM requires placement decisions to respond jointly to task requirements and current ecosys-

tem conditions rather than relying on a predetermined preferred execution tier.

Although context-aware distribution enables coordination within the IoM, the wider IoT landscape contains specialised ecosystems with sensing capabilities, data modalities, and application requirements that extend beyond those native to smart mirrors. Supporting the broader IoM vision therefore introduces an interoperability problem in which tasks originating outside the IoM may need to make use of its distributed computational resources while retaining requirements associated with their original ecosystem and modality. This motivated Q_4, which examined how the IoM can operate intelligently and interoperably within the wider IoT landscape. The thesis addressed this through an interaction between the IoM and the IoS, introducing an Intelligent Orchestration Layer capable of coordinating native visual IoM tasks and externally originated audio workloads. Ecosystem origin, task modality, hardware compatibility, system state, and application context were incorporated into the orchestration formulation, with hard action masking preventing infeasible allocation and a DDQN policy selecting between the remaining resources. Evaluation through a representative multimodal rehabilitation scenario demonstrated that external tasks can use the IoM computational hierarchy while retaining modality-specific feasibility and differentiated application urgency. This IoM-IoS implementation consequently provides an initial operational realisation of the broader IoX vision in which specialised IoT ecosystems retain their individual functions while sharing capabilities through interoperable coordination.

Overall, these investigations establish a progression from defining the IoM to demonstrating how it can operate as a distributed and increasingly interoperable IoT ecosystem. The architectural contribution defines the entities, relationships, and functions required for interconnection. The system-level evaluations establish how distributed execution affects application behaviour across heterogeneous resources. CATD converts the resulting context dependence into an explicit mechanism for coordinating diverse computational demands within the IoM, while the reinforcement-learning-based orchestration framework extends this coordination to tasks originating from an external ecosystem. The contributions therefore address different stages of the same underlying problem of moving from independently operating smart mirrors towards an ecosystem capable of coordinating heterogeneous users, applications, computational resources, and connected environments.

The central outcome of this thesis is the establishment of an operational pathway towards the IoM envisioned at the outset of the research. The work demonstrates that an interconnected mirror ecosystem can support user-facing assistive applications while distributing their computational demands across heterogeneous infrastructure, adapting allocation decisions to changing operating context, and beginning to exchange computational capabilities with complementary IoT ecosystems. The investigations also expose several areas in which further development is required before this vision can be realised at larger deployment scales, particularly physical system expansion,

broader application and user validation, multimodal interoperability, and privacy and security. These directions form the basis of the future work presented in the following section.

7.2 Future Work

The contributions of this thesis establish the initial foundational mechanisms for the IoM, while also identifying several areas that require further investigation as the ecosystem progresses towards larger and more diverse real-world deployments. The following directions build directly on the limitations and opportunities exposed through the evaluations presented in this thesis.

7.2.1 Physical Testbed Expansion for Multi-User IoM Deployments

The physical testbed developed in this thesis provides an initial implementation of the Consumer, Professional, and Hub hierarchy and demonstrates how computational placement affects latency, communication overhead, resource utilisation, and concurrent demand. A natural extension is to increase the scale and heterogeneity of this environment to represent larger IoM deployments containing multiple simultaneously active Consumer, Professional, and Hub nodes. Such an environment would enable the behaviour observed under the present testbed conditions to be examined under higher levels of contention, varying node availability, mobility, and heterogeneous network connectivity.

A larger physical deployment would also provide an opportunity to transfer the task-distribution and orchestration mechanisms evaluated through simulation into real infrastructure. CATD could be integrated with live resource and network monitoring so that allocation decisions respond to measured rather than simulated context, while the orchestration framework developed for IoM-IoS interactions could be evaluated using physical multimodal devices and dynamically changing hardware availability. Particular attention should be given to scheduling and queuing behaviour under simultaneous user demand, together with the computational cost and runtime overhead introduced by task-distribution and orchestration decisions. Evaluating these overheads would enable the performance benefits of adaptive coordination to be assessed against the additional processing and communication costs required to achieve them.

7.2.2 Diverse Use Case Development and Validation

The evaluations in this thesis use representative applications and scenarios to exercise different capabilities of the IoM, including the developed dental smile-analysis appli-

cation. Future work should extend this application base to cover a broader range of health, beauty, accessibility, and assisted-living services with differing sensing modalities, computational requirements, interaction patterns, and user populations. This would enable the architectural and computational mechanisms developed in the thesis to be evaluated against a wider range of practical requirements rather than a limited set of representative workloads.

Further application development should be accompanied by domain-specific and user-centred validation. Where IoM services contribute to health-related assessment or monitoring, evaluation with appropriate clinical or professional expertise would be required to establish the validity and limitations of the generated outputs. For assistive and personalised services, participatory user studies could additionally examine usability, accessibility, trust, and the extent to which personalisation mechanisms accommodate differing user needs. Such validation would complement the system-level metrics considered in this thesis by establishing how improvements in computational performance translate into meaningful service quality and user experience.

7.2.3 Multimodal Representation and Cross-Ecosystem Interoperability

The IoM-IoS evaluation demonstrates one mechanism through which tasks originating from different sensing ecosystems can share computational infrastructure. Extending this model to the wider IoX vision will require common representations through which heterogeneous ecosystems can describe the data, capabilities, requirements, and constraints associated with their tasks. Future research should therefore investigate interoperable task and capability descriptions capable of representing visual, audio, physiological, environmental, and other sensing modalities without requiring the orchestration mechanism to be redesigned for each newly connected ecosystem.

Such representations could support automated capability discovery, allowing an IoM node or external ecosystem to advertise the modalities it can process, available computational resources, access restrictions, expected quality-of-service characteristics, and data-handling requirements. Standardised interfaces and semantic descriptions could then enable orchestration decisions to operate across previously unknown devices and services rather than relying on task profiles defined in advance. This would extend the initial IoM-IoS demonstration towards the broader IoX vision established in this thesis, where interoperability is achieved through shared descriptions and interfaces while individual ecosystems retain their own internal architectures and specialised capabilities.

7.2.4 Privacy Preservation and Security Protocols for IoM Deployments

The user-facing and health-related nature of many IoM applications means that large-scale deployment would involve the collection and processing of potentially sensitive visual, behavioural, and health information across multiple devices and stakeholder domains. This thesis incorporates data sensitivity into task-distribution decisions and demonstrates how locality can influence computational placement, but these mechanisms do not themselves provide a complete privacy or security architecture. Future IoM deployments will therefore require explicit protocols governing identity, authentication, authorisation, user consent, secure communication, data storage, and trust between Consumer, Professional, Hub, and external ecosystem nodes.

Further research should investigate privacy-preserving computation alongside these communication and access-control mechanisms. Depending on the application, this may include encrypted communication and storage, privacy-preserving inference, federated or distributed learning, data minimisation, and policies that constrain where particular categories of information may be transferred or processed. Cross-ecosystem operation introduces an additional challenge because trust must be established between devices and services that may belong to different administrative domains. Integrating security and privacy state into the context considered by future task-distribution and orchestration mechanisms would allow execution decisions to account not only for computational and communication suitability, but also for whether a candidate resource satisfies the trust and data-governance requirements of the service.

Bibliography

[1] M. A. Iqbal, S. Hussain, H. Xing, and M. Imran, "Internet of Things (IoT) Fundamentals," in *Enabling the Internet of Things*, pp. 1–28, Wiley, 2021.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[3] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A Survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[4] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of Things: Vision, Applications and Research Challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012.

[5] L. Bariah, L. Mohjazi, S. Muhaidat, P. C. Sofotasios, G. K. Kurt, H. Yanikomeroglu, and O. A. Dobre, "A Prospective Look: Key Enabling Technologies, Applications and Open Research Topics in 6G Networks," *IEEE Access*, vol. 8, pp. 174792–174820, 2020.

[6] F. Yang, S. Wang, J. Li, Z. Liu, and Q. Sun, "An Overview of Internet of Vehicles," *China Communications*, vol. 11, no. 10, pp. 1–15, 2014.

[7] A. Gatouillat, Y. Badr, B. Massot, and E. Sejdić, "Internet of Medical Things: A Review of Recent Contributions Dealing With Cyber-Physical Systems in Medicine," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3810–3822, 2018.

[8] L. Turchet, M. Lagrange, C. Rottondi, G. Fazekas, N. Peters, J. Østergaard, F. Font, T. Bäckström, and C. Fischione, "The Internet of Sounds: Convergent Trends, Insights, and Future Directions," *IEEE Internet of Things Journal*, vol. 10, no. 13, pp. 11264–11292, 2023.

[9] L. Turchet, C. Fischione, G. Essl, D. Keller, and M. Barthet, "Internet of Musical Things: Vision and Challenges," *IEEE Access*, vol. 6, pp. 61994–62017, 2018.

[10] M. Gréaux, M. F. Moro, K. Kamenov, A. M. Russell, D. Barrett, and A. Cieza, "Health Equity for Persons with Disabilities: A Global Scoping Review on Barriers and Interventions in Healthcare Services," *International Journal for Equity in Health*, vol. 22, no. 1, p. 236, 2023.

[11] E. König-Klever, P. R. Bohn, C. de Souza Libânio, F. S. Fogliatto, and L. M. Tonetto, "Design of Health Technologies for Users Facing Limitations: A Scoping Review," *Technology and Disability*, pp. 1–18, 2024.

[12] M. A. Faudzi, M. Ghazali, Z. C. Cob, R. Omar, and S. A. Sharudin, "The Evaluation of Cognitive Load Significance for Mobile Learning Application via User Interface Design Violations," in *Proc. IEEE Int. Conf. Computing (ICOCO)*, pp. 392–397, 2022.

[13] M. Braun, M. Wölfel, G. Renner, and C. Menschik, "Accessibility of Different Natural User Interfaces for People with Intellectual Disabilities," in *Proc. Int. Conf. Cyberworlds (CW)*, pp. 211–218, 2020.

[14] G. Modanwal, S. B. Rai, A. Jaiswal, T. Singh, and K. Sarawadekar, "Can Visually Impaired Use Gestures to Interact With Computers? A Cognitive Load Perspective," *IEEE Transactions on Human-Machine Systems*, vol. 52, no. 2, pp. 267–275, 2022.

[15] Y. J. A. Khasawneh and M. A. S. Khasawneh, "Cognitive Load Analysis of Adaptive Learning Technologies in Special Education Classrooms: A Quantitative Approach," *International Journal of Advanced and Applied Sciences*, vol. 11, no. 12, pp. 34–41, 2024.

[16] N. Vasco Lopes, "Internet of Things Feasibility for Disabled People," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 12, p. e3906, 2020.

[17] M. Haghi Kashani, M. Madanipour, M. Nikravan, P. Asghari, and E. Mahdipour, "A Systematic Review of IoT in Healthcare: Applications, Techniques, and Trends," *Journal of Network and Computer Applications*, vol. 192, p. 103164, 2021.

[18] R. Rayan, C. Tsagkaris, and I. Romash, "The Internet of Things for Healthcare: Applications, Selected Cases and Challenges," in *IoT in Healthcare and Ambient Assisted Living*, pp. 1–15, Springer, 2021.

[19] P. Kokol, H. Blažun Vošner, J. Završnik, and M. Turčin, "How 'Smart' Is Smart Dentistry?," *F1000Research*, vol. 8, p. 183, 2019.

[20] M. Baidachna, H. Fatima, R. Omran, N. Ghadban, M. A. Imran, A. Taha, and L. Mohjazi, "Mirror, Mirror on the Wall: Automating Dental Smile Analysis with AI in Smart Mirrors," *Computing & AI Connect*, vol. 1, no. 1, pp. 1–10, 2024.

[21] M. Yang, A. C. Y. Lo, and W. C. Lam, "Smart Phone Apps Every Ophthalmologist Should Know About," *International Journal of Ophthalmology*, vol. 13, no. 8, pp. 1329–1333, 2020.

[22] J. Lee, H. Yoon, S. Kim, C. Lee, J. Lee, and S. Yoo, "Deep Learning-Based Skin Care Product Recommendation: A Focus on Cosmetic Ingredient Analysis and Facial Skin Conditions," *Journal of Cosmetic Dermatology*, vol. 23, no. 6, pp. 2066–2077, 2024.

[23] C. Li, J. Wang, S. Wang, and Y. Zhang, "A Review of IoT Applications in Healthcare," *Neurocomputing*, vol. 565, p. 127017, 2024.

[24] M. Rocklin, "IoT for Beauty." `https://www.iterate.ai/blog/iot-for-beauty/1043`, 2021. [Online]. Accessed: Sep. 15, 2023.

[25] Accenture, "Consumers Welcome Personalized Offerings but Businesses Are Struggling to Deliver." `https://newsroom.accenture.com/news/consumers-welcome-personalized-offerings-but-businesses-are-struggling-to-deliver-finds-accenture-interactive-personalization-research.htm`, 2016. [Online]. Accessed: Sep. 15, 2023.

[26] CB Insights, "14 Trends Changing the Face of the Beauty Industry in 2021." `https://www.cbinsights.com/research/report/beauty-trends-2021/`, 2021. [Online]. Accessed: Sep. 26, 2023.

[27] Y. J. Lee, J. W. Choi, H. W. Nam, and S. Y. Shin, "A Study on the Direction of Beauty Tech Reflecting the Skin Characteristics of Koreans: Focused on Case Studies," in *Computer and Information Science 2021 – Fall*, pp. 107–118, Springer, 2022.

[28] L. Griebel, H.-U. Prokosch, F. Köpcke, D. Toddenroth, J. Christoph, I. Leb, I. Engel, and M. Sedlmayr, "A Scoping Review of Cloud Computing in Healthcare," *BMC Medical Informatics and Decision Making*, vol. 15, p. 17, 2015.

[29] D. A. Alboaneen, D. Alsaffar, A. Alateeq, A. Alqahtani, A. Alfahhad, B. Alqahtani, R. Alamri, and L. Alamri, "Internet of Things Based Smart Mirrors: A Literature Review," in *Proc. 3rd Int. Conf. Computer Applications & Information Security (ICCAIS)*, pp. 1–6, 2020.

[30] A. D. Rayome, "How Sephora Is Leveraging AR and AI to Transform Retail and Help Customers Buy Cosmetics." `https://www.techrepublic.com/article/how-sephora-is-leveraging-ar-and-ai-to-transform-retail-and-help-customers-buy-cosmetics/`, 2018. TechRepublic. [Online]. Accessed: Sep. 15, 2023.

[31] A. Krajna and T. Kocejko, "Real-Time Skin Quality Assessment System," in *Proc. 16th Int. Conf. Human System Interaction (HSI)*, pp. 1–6, 2024.

[32] R. Miotto, M. Danieletto, J. R. Scelza, B. A. Kidd, and J. T. Dudley, "Reflecting Health: Smart Mirrors for Personalized Medicine," *npj Digital Medicine*, vol. 1, no. 1, pp. 1–7, 2018.

[33] G. Casalino, G. Castellano, V. Pasquadibisceglie, and G. Zaza, "Improving a Mirror-Based Healthcare System for Real-Time Estimation of Vital Parameters," *Information Systems Frontiers*, vol. 27, no. 5, pp. 1925–1941, 2025.

[34] K. Jin, X. Deng, Z. Huang, and S. Chen, "Design of the Smart Mirror Based on Raspberry Pi," in *Proc. 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conf. (IMCEC)*, pp. 1919–1923, 2018.

[35] Global Market Insights, "Smart Mirror Market Size & Share – Trends Report, 2032." `https://www.gminsights.com/industry-analysis/smart-mirror-market`, 2024. [Online]. Accessed: Oct. 1, 2024.

[36] J. Yum, J. Lee, P. Y. Lai, and J.-H. Lee, "Improving User-Centered Interface for Smart Mirror as a Product-Service System Design," in *Distributed, Ambient and Pervasive Interactions. Smart Environments, Ecosystems, and Cities*, pp. 340–360, Springer, 2022.

[37] M. Ghazal, T. Al Hadithy, Y. Al Khalil, M. Akmal, and H. Hajjdiab, "A Mobile-Programmable Smart Mirror for Ambient IoT Environments," in *Proc. 5th Int. Conf. Future Internet of Things and Cloud Workshops (FiCloudW)*, pp. 240–245, 2017.

[38] D. S. Sahana and C. Gopala Krishnan, "Raspbian Magic Mirror: A Smart Mirror System to Assist on IoT Platform," in *IoT Based Smart Applications*, pp. 261–276, Springer, 2023.

[39] D. Gold, D. Sollinger, and Indratmo, "SmartReflect: A Modular Smart Mirror Application Platform," in *Proc. IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conf. (IEMCON)*, pp. 1–7, 2016.

[40] H. Boley and E. Chang, "Digital Ecosystems: Principles and Semantics," in *Proc. Inaugural IEEE-IES Digital EcoSystems and Technologies Conf. (DEST)*, pp. 398–403, 2007.

[41] O. Mazhelis, E. Luoma, and H. Warma, "Defining an Internet-of-Things Ecosystem," in *Internet of Things, Smart Spaces, and Next Generation Networking*, vol. 7469 of *Lecture Notes in Computer Science*, pp. 1–14, Springer, 2012.

[42] G. Paolone, D. Iachetti, R. Paesani, F. Pilotti, M. Marinelli, and P. Di Felice, "A Holistic Overview of the Internet of Things Ecosystem," *IoT*, vol. 3, no. 4, pp. 398–434, 2022.

[43] J. Kasparian, "Your New Favorite Beauty Gadget for 2021: A Smart Mirror." `https://reviewed.usatoday.com/beauty/features/smart-mirrors-new-beauty-trend-2021`, 2021. [Online]. Accessed: Sep. 15, 2023.

[44] T. Loncar-Turukalo, E. Zdravevski, J. Machado da Silva, I. Chouvarda, and V. Trajkovik, "Literature on Wearable Technology for Connected Health: Scoping Review of Research Trends, Advances, and Barriers," *Journal of Medical Internet Research*, vol. 21, no. 9, p. e14017, 2019.

[45] E. Iwabuchi, M. Nakagawa, and I. Siio, "Smart Makeup Mirror: Computer-Augmented Mirror to Aid Makeup Application," in *Human-Computer Interaction. Ambient, Ubiquitous and Intelligent Interaction*, vol. 5613 of *Lecture Notes in Computer Science*, pp. 495–503, Springer, 2009.

[46] A. S. M. M. Rahman, T. Tran, S. A. Hossain, and A. El Saddik, "Augmented Rendering of Makeup Features in a Smart Interactive Mirror System for Decision Support in Cosmetic Products Selection," in *Proc. IEEE/ACM 14th Int. Symp. Distributed Simulation and Real Time Applications*, pp. 203–206, 2010.

[47] M. Teeuw, "Magic Mirror." `https://michaelteeuw.nl/series/MagicMirror`. [Online]. Accessed: Sep. 15, 2023.

[48] R. M. A. Latif, L.-B. Imran, M. Farhan, M. J. Bah, G. Ali, and Y. A. Abid, "Real-Time Simulation of IoT Based Smart Home Live Mirror Using WSN," in *Proc. Int. Conf. Frontiers of Information Technology (FIT)*, pp. 49–495, 2019.

[49] P. Prasad, "Making Smart Mirror and Interactive Using IoT," *International Journal for Research in Applied Science and Engineering Technology*, vol. 9, no. 5, pp. 1582–1585, 2021.

[50] J. Manjrekar, M. Pandit, S. Salvi, and Y. Chavan, "Smart Mirror Using Internet of Things," *International Journal for Research in Applied Science and Engineering Technology*, vol. 10, no. 4, 2022.

[51] A. Barbadekar, S. Bhake, and S. Parekh, "Design of a Cost-Effective Smart Mirror Using Raspberry Pi," in *Proc. Int. Conf. for Advancement in Technology (ICONAT)*, pp. 1–5, 2023.

[52] K. Subramanya Chari, M. Raja, and S. R. Kishore, "Raspberry Pi-Based Smart Mirror," in *Advances in Cognitive Science and Communications*, pp. 211–220, Springer, 2023.

[53] R. J. Sapkal, Y. R. Dhumal, S. Joshi, and A. Jondhale, "Beyond Reflection: Smart Mirrors in the Internet of Things Era," *International Research Journal on Advanced Engineering Hub (IRJAEH)*, vol. 2, no. 7, pp. 2059–2064, 2024.

[54] N. Shekokar, S. Shinde, S. Ambarkar, A. Michalas, M. Mangla, and A. Thomas, "Challenges and solutions in internet of things-based smart applications," Chapman and Hall/CRC, 1st ed., 2025.

[55] K. B. Prajapati, C. Bhatt, and H. Chaouchi, "An Interactive Smart Mirror Using Internet of Things and Machine Learning," in *IoT and Analytics for Sensor Networks*, pp. 1–9, Springer, 2022.

[56] Y. Sun, L. Geng, and K. Dan, "Design of Smart Mirror Based on Raspberry Pi," in *Proc. Int. Conf. Intelligent Transportation, Big Data & Smart City (ICITBS)*, pp. 77–80, 2018.

[57] S. Athira, F. Francis, R. Raphel, N. S. Sachin, S. Porinchu, and S. Francis, "Smart Mirror: A Novel Framework for Interactive Display," in *Proc. Int. Conf. Circuit, Power and Computing Technologies (ICCPCT)*, pp. 1–6, 2016.

[58] R. Akshaya, N. N. Raj, and S. Gowri, "Smart Mirror – Digital Magazine for University Implemented Using Raspberry Pi," in *Proc. Int. Conf. Emerging Trends and Innovations in Engineering and Technological Research (ICETIETR)*, pp. 1–4, 2018.

[59] A. G. Putrada, I. D. Oktaviani, M. N. Fauzan, and N. Alamsyah, "CNN-LSTM for MFCC-Based Speech Recognition on Smart Mirrors for Edge Computing Command," *Journal of Dinda: Data Science, Information Technology, and Data Analytics*, vol. 4, no. 2, pp. 63–74, 2024.

[60] S. Rigling, S. Avdic, M. E. Ozver, and M. Sedlmair, "Reverse Vampire UI: Reflecting on AR Interaction with Smart Mirrors," in *Extended Abstracts of the CHI Conf. Human Factors in Computing Systems*, pp. 1–7, 2025.

[61] Harsha M. S., Nischala G. S., Prajwal H. J., Smitha Gayathri D., Manonmaya Tandula S. V., and Nandish C., "A Privacy-Preserving Multimodal Smart Mirror with Edge AI, Blockchain Security and Quantum-Inspired Optimization for Personalized Wellness," in *Proc. Int. Conf. Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE)*, pp. 1–6, 2026.

[62] A. Rat, C. Stangaciu, and M. Micea, "Energy Consumption Analysis for Raspberry Pi Based Smart Mirrors and Proposed Solutions," in *Proc. IEEE 16th Int. Symp. Applied Computational Intelligence and Informatics (SACI)*, pp. 197–202, 2022.

[63] B. Bargh, "Sephora Introduces AI-Powered Digital Mirror to Spanish Flagship." `https://cosmeticsbusiness.com/news/article_page/Sephora_introduces_AI-powered_digital_mirror_to_Spanish_flagship/153247`, 2019. Cosmetics Business. [Online]. Accessed: Sep. 15, 2023.

[64] HiMirror, "HiMirror: Smart Mirror x Smarter Beauty." `https://www.himirror.com/`. [Online]. Accessed: Sep. 15, 2023.

[65] Memomi, "Augmented Reality & Artificial Intelligence Mirror Software." https://memorymirror.com/. [Online]. Accessed: Sep. 15, 2023.

[66] Charlotte Tilbury, "The Magic Mirror In Store." https://www.charlottetilbury.com/uk/secrets/charlotte-tilbury-magic-mirror. [Online]. Accessed: Sep. 15, 2023.

[67] Shiseido Co., Ltd., "First in Japan: Next Generation Cosmetics Counter with Digital Counseling Mirror." https://corp.shiseido.com/en/news/detail.html?n=00000000002241. [Online]. Accessed: Sep. 15, 2023.

[68] K. Hua, "Panasonic's New Smart Mirror Shows You Your Flaws and Helps You Fix Them." https://www.forbes.com/sites/karenhua/2016/10/07/panasonic-smart-mirror-shows-you-your-face-flaws-and-helps-you-fix-them/, 2016. Forbes. [Online]. Accessed: Sep. 15, 2023.

[69] ModiFace, "ModiFace – Augmented Reality." https://modiface.com/. [Online]. Accessed: Sep. 15, 2023.

[70] Retail Customer Experience, "Skin Care Startup Offers AI-Enabled Skin Care Kiosk." https://www.retailcustomerexperience.com/news/skin-care-startup-offers-ai-enabled-skin-care-kiosk/, 2021. [Online]. Accessed: Sep. 15, 2023.

[71] L. Ulanoff, "This Smart Mirror Will Have All the Answers, and Your Music, Too." https://www.techradar.com/news/this-smart-mirror-will-have-all-the-answers-and-your-music-too, 2022. TechRadar. [Online]. Accessed: Sep. 15, 2023.

[72] L. Chang, "The Morror Is a Smart Mirror That Provides Great Lighting and Makeup Tips." https://www.digitaltrends.com/home/morror-smart-mirror/, 2018. Digital Trends. [Online]. Accessed: Sep. 15, 2023.

[73] simplehuman, "Sensor Mirror." https://www.simplehuman.co.uk/collections/sensor-mirror. [Online]. Accessed: Sep. 15, 2023.

[74] Digital Trends, "Kohler's Alexa-Enabled Smart Mirror Controls the Whole Bathroom." https://www.digitaltrends.com/home/kohler-verdera-mirror-ces-2018/, 2018. [Online]. Accessed: Sep. 15, 2023.

[75] iHome, "iHome Vanity Mirror Speakers with Full-Color Spectrum Lighting." https://ihomeaudiointl.com/beauty/. [Online]. Accessed: Sep. 15, 2023.

[76] K. Holt, "Capstone's Android-Powered Smart Mirror Is Now Available." https://www.forbes.com/sites/krisholt/2020/01/06/capstones-android-powered-smart-mirror-is-now-available/, 2020. Forbes. [Online]. Accessed: Sep. 15, 2023.

[77] R. A. Nadaf and V. Bonal, "Smart Mirror Using Raspberry Pi for Intrusion Detection and Human Monitoring," in *Advances in Artificial Intelligence and Data Engineering*, pp. 1359–1373, Springer, 2021.

[78] N. Prajugjit, P. Silapasuphakornwong, K. Sookhanaphibarn, and K. Uehira, "Friendly Chatbot with IBM Watson Assistant Support Elderly Delightful in Smart Mirror," in *Proc. IEEE 11th Global Conf. Consumer Electronics (GCCE)*, pp. 332–334, 2022.

[79] B. Yang, T. Kang, Y. Choi, and J. Jung, "Error-Resistant Movement Detection Algorithm for the Elderly with Smart Mirror," *Applied Sciences*, vol. 12, no. 14, p. 7024, 2022.

[80] A. E. A. Hippocrate, E. T. Luhanga, T. Masashi, K. Watanabe, and K. Yasumoto, "Smart Gyms Need Smart Mirrors: Design of a Smart Gym Concept Through Contextual Inquiry," in *Proc. ACM Int. Joint Conf. Pervasive and Ubiquitous Computing (UbiComp) Adjunct*, pp. 658–661, 2017.

[81] O. Gomez-Carmona and D. Casado-Mansilla, "SmiWork: An Interactive Smart Mirror Platform for Workplace Health Promotion," in *Proc. 2nd Int. Multidisciplinary Conf. Computer and Energy Science (SpliTech)*, pp. 1–6, 2017.

[82] X. He, Y. Hu, J. Zheng, X. Yang, and X. Gong, "Mirror Face: A Large Language Model-Based Workplace System for Emotion Regulation," *International Journal of Human–Computer Interaction*, pp. 1–19, 2026.

[83] A. Muneer, S. M. Fati, and S. Fuddah, "Smart Health Monitoring System Using IoT Based Smart Fitness Mirror," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 18, no. 1, pp. 317–331, 2020.

[84] S. Ahn and Y. Choi, "Smiley: Designing Smile Recognition Smart Mirror for Premenstrual Syndrome," *International Journal of Human–Computer Interaction*, pp. 1–19, 2022.

[85] G. Kuruppu, N. Wijesinghe, and P. Kumarawadu, "Smart Mirror for Personalized Content Delivery Through Facial Emotion Recognition," in *Proc. 3rd Int. Conf. Sustainable Technologies for Industry 4.0 (STI)*, pp. 1–6, 2021.

[86] R.-P. Yang, Z.-T. Liu, L.-D. Zheng, J.-P. Wu, and C.-C. Hu, "Intelligent Mirror System Based on Facial Expression Recognition and Color Emotion Adaptation – iMirror," in *Proc. 37th Chinese Control Conf. (CCC)*, pp. 3227–3232, 2018.

[87] A. C. Njaka, N. Li, and L. Li, "Voice Controlled Smart Mirror with Multifactor Authentication," in *Proc. IEEE Int. Smart Cities Conf. (ISC2)*, pp. 1–8, 2018.

[88] K. M. M. Uddin, S. K. Dey, G. U. Parvez, A. S. Mukta, and U. K. Acharjee, "MirrorME: Implementation of an IoT Based Smart Mirror Through Facial Recognition and Personalized Information Recommendation Algorithm," *International Journal of Information Technology*, vol. 13, no. 6, pp. 2313–2322, 2021.

[89] T. V. Nguyen and L. Liu, "Smart Mirror: Intelligent Makeup Recommendation and Synthesis," in *Proc. 25th ACM Int. Conf. Multimedia*, pp. 1253–1254, 2017.

[90] A. Dragomir, G. Simion, and I. Ermalai, "Smart Mirror for Skin Type Identification," in *Proc. Int. Symp. Electronics and Telecommunications (ISETC)*, pp. 1–4, 2020.

[91] A. Kumar, A. Yadav, S. Chahar, and E. Uchoi, "Smart Mirror with Deep Learning for Health Evaluation, Skincare, and Virtual Enhancement," in *Proc. Int. Conf. Engineering, Technology & Management (ICETM)*, pp. 1–6, 2025.

[92] X. Guo, S. Yu, X. Cao, Y. Jiang, K. Chen, and H. Guo, "Research on Innovative Design and Evaluation of Smart Dresser Mirror Based on Kano-FBS," in *Proc. Int. Conf. Intelligent Engineering and Next-Gen Healthcare Systems (IEHNS)*, pp. 1–6, 2026.

[93] A. Y. Owda and M. Owda, "Early Detection of Skin Disorders and Diseases Using Radiometry," *Diagnostics*, vol. 12, no. 9, p. 2117, 2022.

[94] M. Villarroel, J. Jorge, C. Pugh, and L. Tarassenko, "Non-Contact Vital Sign Monitoring in the Clinic," in *Proc. 12th IEEE Int. Conf. Automatic Face & Gesture Recognition (FG)*, pp. 278–285, 2017.

[95] S. A. Shah and F. Fioranelli, "RF Sensing Technologies for Assisted Daily Living in Healthcare: A Comprehensive Review," *IEEE Aerospace and Electronic Systems Magazine*, vol. 34, no. 11, pp. 26–44, 2019.

[96] J. S. Alarifi, M. Goyal, A. K. Davison, D. Dancey, R. Khan, and M. H. Yap, "Facial Skin Classification Using Convolutional Neural Networks," in *Image Analysis and Recognition*, pp. 479–485, Springer, 2017.

[97] S. Jain, V. Jagtap, and N. Pise, "Computer Aided Melanoma Skin Cancer Detection Using Image Processing," *Procedia Computer Science*, vol. 48, pp. 735–740, 2015.

[98] N. S. ALKolifi ALEnezi, "A Method of Skin Disease Detection Using Image Processing and Machine Learning," *Procedia Computer Science*, vol. 163, pp. 85–92, 2019.

[99] R. Sharma and V. Mehan, "Skin Disease Detection Using Image Processing and Soft Computing," *ECS Transactions*, vol. 107, no. 1, p. 17051, 2022.

[100] S. Prigent, X. Descombes, D. Zugaj, P. Martel, and J. Zerubia, "Multi-Spectral Image Analysis for Skin Pigmentation Classification," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, pp. 3641–3644, 2010.

[101] I. Kuzmina, I. Diebele, L. Valeine, D. Jakovels, A. Kempele, J. Kapostinsh, and J. Spigulis, "Multi-Spectral Imaging Analysis of Pigmented and Vascular Skin Lesions: Results of a Clinical Trial," in *Proc. SPIE, Photonic Therapeutics and Diagnostics VII*, vol. 7883, pp. 195–201, 2011.

[102] X. Zhang, W. Pan, C. Bontozoglou, E. Chirikhina, D. Chen, and P. Xiao, "Skin Capacitive Imaging Analysis Using Deep Learning GoogLeNet," in *Intelligent Computing*, vol. 1229 of *Advances in Intelligent Systems and Computing*, pp. 395–404, Springer, 2020.

[103] A. Ahmad Tarar, U. Mohammad, and S. K. Srivastava, "Wearable Skin Sensors and Their Challenges: A Review of Transdermal, Optical, and Mechanical Sensors," *Biosensors*, vol. 10, no. 6, p. 56, 2020.

[104] R. Sinhal, K. Singh, and M. M. Raghuwanshi, "An Overview of Remote Photoplethysmography Methods for Vital Sign Monitoring," in *Computer Vision and Machine Intelligence in Medical Image Analysis*, pp. 21–31, Springer, 2020.

[105] H. Ghanadian, M. Ghodratigohar, and H. Al Osman, "A Machine Learning Method to Improve Non-Contact Heart Rate Monitoring Using an RGB Camera," *IEEE Access*, vol. 6, pp. 57085–57094, 2018.

[106] U. Debnath and S. Kim, "A comprehensive review of heart rate measurement using remote photoplethysmography and deep learning," *BioMedical Engineering OnLine*, vol. 24, p. 73, June 2025.

[107] D. Shahmirzadi, W.-F. Wang, and A. M. Rahmani, "Remote photoplethysmography for camera-based vital sign estimation: A systematic literature review, challenges and deployment guidelines," *Array*, vol. 31, p. 100969, Sept. 2026.

[108] J. Verstockt, S. Verspeek, F. Thiessen, W. A. Tjalma, L. Brochez, and G. Steenackers, "Skin Cancer Detection Using Infrared Thermography: Measurement Setup, Procedure and Equipment," *Sensors*, vol. 22, no. 9, p. 3327, 2022.

[109] M. M. Vergilio, G. Gomes, L. M. Aiello, M. Fontana, A. Aldred, J. A. de Souza Ribeiro, T. V. B. Gabbi, and G. R. Leonardi, "Evaluation of Skin Using Infrared Thermal Imaging for Dermatology and Aesthetic Applications," *Journal of Cosmetic Dermatology*, vol. 21, no. 3, pp. 895–904, 2022.

[110] P. Jagadev and L. I. Giri, "Human Respiration Monitoring Using Infrared Thermography and Artificial Intelligence," *Biomedical Physics & Engineering Express*, vol. 6, no. 3, p. 035007, 2020.

[111] P. Jagadev and L. I. Giri, "Non-Contact Monitoring of Human Respiration Using Infrared Thermography and Machine Learning," *Infrared Physics & Technology*, vol. 104, p. 103117, 2020.

[112] M. Švantner, V. Lang, J. Skála, T. Kohlschütter, M. Honner, L. Muzika, and E. Kosová, "Statistical Study on Human Temperature Measurement by Infrared Thermography," *Sensors*, vol. 22, p. 8395, Jan. 2022.

[113] A. Wnuk-Scardaccione and J. Bilski, "Clinical Use of Infrared Thermography: Where Are We and Where Are We Going," *Medicina*, vol. 62, p. 1204, June 2026.

[114] M. Kebe, R. Gadhafi, B. Mohammad, M. Sanduleanu, H. Saleh, and M. Al-Qutayri, "Human Vital Signs Detection Methods and Potential Using Radars: A Review," *Sensors*, vol. 20, no. 5, p. 1454, 2020.

[115] J. Gao, S. Baskar, D. Teng, M. al'Absi, S. Kumar, and E. Ertin, "A New Direction for Biosensing: RF Sensors for Monitoring Cardio-Pulmonary Function," in *Mobile Health: Sensors, Analytic Methods, and Applications*, pp. 289–312, Springer, 2017.

[116] Y. Wu, H. Ni, C. Mao, J. Han, and W. Xu, "Non-intrusive Human Vital Sign Detection Using mmWave Sensing Technologies: A Review," *ACM Transactions on Sensor Networks*, vol. 20, pp. 16:1–16:36, Nov. 2023.

[117] M. Ostrysz, Z. Szczepaniak, and T. Sondej, "Non-Contact Measurement of Human Vital Signs in Dynamic Conditions Using Microwave Techniques: A Review," *Sensors*, vol. 26, p. 359, Jan. 2026.

[118] L. N. Nguyen, P. Susarla, A. Mukherjee, M. L. Canellas, C. A. Casado, X. Wu, O. Silven, D. B. Jayagopi, and M. B. Lopez, "Non-contact multimodal indoor human monitoring systems: A survey," *Information Fusion*, vol. 110, p. 102457, Oct. 2024.

[119] Z. Li, X. Wu, C. Álvarez Casado, V. Lindholm, K. Mikkonen, Z. Xia, X. Feng, and M. Bordallo López, "A comprehensive survey on contactless vital sign monitoring using vision-based, radio-based, and fusion approaches," *Neurocomputing*, vol. 674, p. 132877, Apr. 2026.

[120] N. Sharma, V. Jain, and A. Mishra, "An Analysis of Convolutional Neural Networks for Image Classification," *Procedia Computer Science*, vol. 132, pp. 377–384, 2018.

[121] M. C. Kim and J.-H. Lee, "Color Reproduction in Virtual Lip Makeup Using a Convolutional Neural Network," *Color Research & Application*, vol. 45, no. 6, pp. 1190–1201, 2020.

[122] X. Shen, J. Zhang, C. Yan, and H. Zhou, "An Automatic Diagnosis Method of Facial Acne Vulgaris Based on Convolutional Neural Network," *Scientific Reports*, vol. 8, no. 1, p. 5839, 2018.

[123] T. Zhao, H. Zhang, and J. Spoelstra, "A Computer Vision Application for Assessing Facial Acne Severity from Selfie Images," *arXiv preprint arXiv:1907.07901*, 2019.

[124] A. de Fátima Soares Borges and C. H. Morimoto, "A Virtual Makeup Augmented Reality System," in *Proc. 21st Symp. Virtual and Augmented Reality (SVR)*, pp. 34–42, 2019.

[125] P. R. H. Perera, E. S. S. Soysa, H. R. S. De Silva, A. R. P. Tavarayan, M. P. Gamage, and K. M. L. P. Weerasinghe, "Virtual Makeover and Makeup Recommendation Based on Personal Trait Analysis," in *Proc. 3rd Int. Conf. Advancements in Computing (ICAC)*, pp. 288–293, 2021.

[126] J. Cho, J. Ohya, and S. I. Park, "Makeup Interpolation Based on Color and Shape Parametrization," *Symmetry*, vol. 11, no. 11, p. 1422, 2019.

[127] S. Aly, A. Abdallah, S. Sherif, A. Atef, R. Adel, and M. Hatem, "Toward Smart Internet of Things (IoT) for Apparel Retail Industry: Automatic Customer's Body Measurements and Size Recommendation System Using Computer Vision Techniques," in *Proc. 2nd Int. Conf. Intelligent Data Science Technologies and Applications (IDSTA)*, pp. 99–104, 2021.

[128] M. Ueta, "Improving Mirror Fitness Through Augmented Reality Technology," in *Proc. 3rd Int. Conf. Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, pp. 186–194, 2022.

[129] P. Eisert, P. Fechteler, and J. Rurainsky, "3-D Tracking of Shoes for Virtual Mirror Applications," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 1–6, 2008.

[130] T. Alzahrani, B. Al-Bander, and W. Al-Nuaimy, "Deep Learning Models for Automatic Makeup Detection," *AI*, vol. 2, no. 4, pp. 497–511, 2021.

[131] S. Wang and Y. Fu, "Face Behind Makeup," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, 2016.

[132] M.-C. Hu, H.-T. Wu, L.-Y. Lo, T.-Y. Pan, W.-H. Cheng, K.-L. Hua, and T. Mei, "A Framework of Enlarging Face Datasets Used for Makeup Face Analysis," in *Proc. IEEE 2nd Int. Conf. Multimedia Big Data (BigMM)*, pp. 219–222, 2016.

[133] J. Lee, J. Yum, M. Lee, and J.-H. Lee, "A Cognitive Knowledge-Based System for Hair and Makeup Recommendation Based on Facial Features Classification," in *Proc. Int. Conf. Multimodal Interaction (ICMI)*, pp. 384–394, 2022.

[134] K. Gulati, G. Verma, M. Mohania, and A. Kundu, "BeautifAI – A Personalised Occasion-Oriented Makeup Recommendation System," *arXiv preprint arXiv:2109.06072*, 2022.

[135] M. Li, B. Huang, and G. Tian, "A Comprehensive Survey on 3D Face Recognition Methods," *Engineering Applications of Artificial Intelligence*, vol. 110, p. 104669, 2022.

[136] A. Da'u and N. Salim, "Recommendation System Based on Deep Learning Methods: A Systematic Review and New Directions," *Artificial Intelligence Review*, vol. 53, no. 4, pp. 2709–2748, 2020.

[137] N. M. Villegas, C. Sánchez, J. Díaz-Cely, and G. Tamura, "Characterizing Context-Aware Recommender Systems: A Systematic Literature Review," *Knowledge-Based Systems*, vol. 140, pp. 173–200, 2018.

[138] L. Chettri and R. Bera, "A Comprehensive Survey on Internet of Things (IoT) Toward 5G Wireless Systems," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 16–32, 2020.

[139] K. Shafique, B. A. Khawaja, F. Sabir, S. Qazi, and M. Mustaqim, "Internet of Things (IoT) for Next-Generation Smart Systems: A Review of Current Challenges, Future Trends and Prospects for Emerging 5G-IoT Scenarios," *IEEE Access*, vol. 8, pp. 23022–23040, 2020.

[140] M. Al-Hawawreh, N. Moustafa, and J. Slay, "A Threat Intelligence Framework for Protecting Smart Satellite-Based Healthcare Networks," *Neural Computing and Applications*, 2021.

[141] Q. V. Khanh, N. V. Hoai, L. D. Manh, A. N. Le, and G. Jeon, "Wireless Communication Technologies for IoT in 5G: Vision, Applications, and Challenges," *Wireless Communications and Mobile Computing*, vol. 2022, p. 3229294, 2022.

[142] R. Moraes and H. Sadjadpour, "Wireless Network Protocols," in *Handbook of Wireless Networks and Mobile Computing*, pp. 603–614, 2017.

[143] J. Ding, M. Nemati, C. Ranaweera, and J. Choi, "IoT Connectivity Technologies and Applications: A Survey," *IEEE Access*, vol. 8, pp. 67646–67673, 2020.

[144] H. Shokri-Ghadikolaei, C. Fischione, G. Fodor, P. Popovski, and M. Zorzi, "Millimeter Wave Cellular Networks: A MAC Layer Perspective," *IEEE Transactions on Communications*, vol. 63, no. 10, pp. 3437–3458, 2015.

[145] L. Turchet and P. Casari, "Latency and Reliability Analysis of a 5G-Enabled Internet of Musical Things System," *IEEE Internet of Things Journal*, vol. 11, no. 1, pp. 1228–1240, 2024.

[146] C. Arendt, S. Fricke, S. Böcker, and C. Wietfeld, "Distributed Performance Evaluation of 5G and Wi-Fi for Private Industrial Networks," in *Proc. IEEE 35th Int. Symp. Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 1–7, 2024.

[147] J. Cao, X. Su, B. Finley, A. Pauanne, M. Ammar, and P. Hui, "Evaluating Multimedia Protocols on 5G Edge for Mobile Augmented Reality," in *Proc. 17th Int. Conf. Mobility, Sensing and Networking (MSN)*, pp. 199–206, 2021.

[148] A. Ali and K. Ali, "Multiradio Parallel Offloading in Multiaccess Edge Computing: Optimizing Load Shares, Scheduling, and Capacity," *IEEE Internet of Things Journal*, vol. 11, no. 3, pp. 4047–4062, 2024.

[149] D. Georgakopoulos, P. P. Jayaraman, M. Fazia, M. Villari, and R. Ranjan, "Internet of Things and Edge Cloud Computing Roadmap for Manufacturing," *IEEE Cloud Computing*, vol. 3, no. 4, pp. 66–73, 2016.

[150] M. Satyanarayanan, "The Emergence of Edge Computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[151] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[152] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog Computing: A Platform for Internet of Things and Analytics," in *Big Data and Internet of Things: A Roadmap for Smart Environments*, pp. 169–186, Springer, 2014.

[153] Z. Ahmadi, M. Haghi Kashani, M. Nikravan, and E. Mahdipour, "Fog-Based Healthcare Systems: A Systematic Review," *Multimedia Tools and Applications*, vol. 80, no. 30, pp. 36361–36400, 2021.

[154] P. Moore and H. Van Pham, "FOG Computing and Low Latency Context-Aware Health Monitoring in Smart Interconnected Environments," in *Advances in Internet, Data & Web Technologies*, pp. 29–40, Springer, 2018.

[155] H. El-Sayed, S. Sankar, M. Prasad, D. Puthal, A. Gupta, M. Mohanty, and C.-T. Lin, "Edge of Things: The Big Picture on the Integration of Edge, IoT and the Cloud in a Distributed Computing Environment," *IEEE Access*, vol. 6, pp. 1706–1717, 2018.

[156] P. J. Escamilla-Ambrosio, A. Rodríguez-Mota, E. Aguirre-Anaya, R. Acosta-Bermejo, and M. Salinas-Rosales, "Distributing Computing in the Internet of Things: Cloud, Fog and Edge Computing Overview," in *NEO 2016: Results of the Numerical and Evolutionary Optimization Workshop*, pp. 87–115, Springer, 2018.

[157] H. F. Shahid, B. Akdemir, J. Islam, I. Ahmad, and E. Harjula, "IoT Service Orchestration in Edge-Cloud Continuum with 6G: A Review," *IEEE Internet of Things Journal*, 2026.

[158] N. Fernando, S. Shrestha, S. W. Loke, and K. Lee, "On Edge-Fog-Cloud Collaboration and Reaping Its Benefits: A Heterogeneous Multi-Tier Edge Computing Architecture," *Future Internet*, vol. 17, no. 1, p. 22, 2025.

[159] M. Guo, L. Li, and Q. Guan, "Energy-Efficient and Delay-Guaranteed Workload Allocation in IoT-Edge-Cloud Computing Systems," *IEEE Access*, vol. 7, pp. 78685–78697, 2019.

[160] A. Nainwal, M. Almusawi, S. Bansal, J. Michaelson, S. Kumar, and K. Sangeeta, "Efficient Load Balancing Algorithms for Edge Computing in IoT Environments," in *Proc. Int. Conf. Communication, Computer Sciences and Engineering (IC3SE)*, pp. 1646–1651, 2024.

[161] X. Zhu, W. Yao, and W. Wang, "Load-Aware Task Migration Algorithm Toward Adaptive Load Balancing in Edge Computing," *Future Generation Computer Systems*, vol. 157, pp. 303–312, 2024.

[162] P. Shukla, S. Pandey, P. Hatwar, and A. Pant, "FAT-ETO: Fuzzy-AHP-TOPSIS-Based Efficient Task Offloading Algorithm for Scientific Workflows in Heterogeneous Fog–Cloud Environment," *Proceedings of the National Academy of Sciences, India Section A: Physical Sciences*, vol. 93, no. 2, pp. 339–353, 2023.

[163] E. Al-Masri and S. V. Paruchuri, "FogScheduler: A Resource Optimization Framework for Energy-Efficient Computing in Fog Environments," *Internet of Things*, vol. 32, p. 101609, 2025.

[164] M. S. U. Islam and A. Kumar, "Context-Aware Application Scheduling in Fog Computing Environment," *Concurrency and Computation: Practice and Experience*, vol. 35, no. 26, p. e7836, 2023.

[165] R. Chen and X. Wang, "Situation-Aware Orchestration of Resource Allocation and Task Scheduling for Collaborative Rendering in IoT Visualization," *IEEE Transactions on Sustainable Computing*, vol. 7, no. 4, pp. 935–949, 2022.

[166] K. Wang, J. Jin, Y. Yang, T. Zhang, A. Nallanathan, C. Tellambura, and B. Jabbari, "Task Offloading With Multi-Tier Computing Resources in Next Generation Wireless Networks," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 2, pp. 306–319, 2023.

[167] A. Kiani and N. Ansari, "Towards Hierarchical Mobile Edge Computing: An Auction-Based Profit Maximization Approach," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2082–2091, 2017.

[168] P. Liu, Z. Liu, J. Wang, Z. Wu, P. Li, and H. Lu, "Reinforcement Learning Empowered Multi-AGV Offloading Scheduling in Edge-Cloud IIoT," *Journal of Cloud Computing*, vol. 11, no. 1, pp. 1–14, 2022.

[169] M. R. Raju and S. K. Mothku, "Delay and Energy Aware Task Scheduling Mechanism for Fog-Enabled IoT Applications: A Reinforcement Learning Approach," *Computer Networks*, vol. 224, p. 109603, 2023.

[170] Y. Ma, Y. Zhao, Y. Hu, X. He, and S. Feng, "Multi-Agent Deep Reinforcement Learning for Joint Task Offloading and Resource Allocation in IIoT with Dynamic Priorities," *Sensors*, vol. 25, no. 19, p. 6160, 2025.

[171] T. Shi, T. Zhang, Y. Liu, and R. Huang, "Multimodal Mobile Edge Computing: Multi-Objective Optimization With Synchronization Constraint," *IEEE Transactions on Mobile Computing*, vol. 25, pp. 8034–8049, 2025.

[172] J. Liang, B. Ma, Z. Feng, and J. Huang, "Reliability-Aware Task Processing and Offloading for Data-Intensive Applications in Edge Computing," *IEEE Transactions on Network and Service Management*, 2023.

[173] C. Jiang, X. Cheng, H. Gao, X. Zhou, and J. Wan, "Toward Computation Offloading in Edge Computing: A Survey," *IEEE Access*, vol. 7, pp. 131543–131558, 2019.

[174] S. K. Sahoo and S. K. Mishra, "A Survey on Task Scheduling in Edge-Cloud," *SN Computer Science*, vol. 6, no. 3, p. 217, 2025.

[175] S. Zhang, N. Yi, and Y. Ma, "A Survey of Computation Offloading With Task Types," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 8, pp. 8313–8333, 2024.

[176] J. Chen, Y. Leng, and J. Huang, "An Intelligent Approach of Task Offloading for Dependent Services in Mobile Edge Computing," *Journal of Cloud Computing*, vol. 12, no. 1, p. 107, 2023.

[177] Y. Zhang, J. Chen, Y. Zhou, L. Yang, B. He, and Y. Yang, "Dependent Task Offloading with Energy-Latency Tradeoff in Mobile Edge Computing," *IET Communications*, vol. 16, no. 17, pp. 1993–2001, 2022.

[178] E. Gyamfi and A. Jurcut, "A Robust Security Task Offloading in Industrial IoT-Enabled Distributed Multi-Access Edge Computing," *Frontiers in Signal Processing*, vol. 2, p. 788943, 2022.

[179] I. AlQerm and J. Pan, "I-HARF: Intelligent and Hierarchical Framework for Adaptive Resource Facilitation in Edge-IoT Systems," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 3954–3967, 2023.

[180] W. Chen, X. Qiu, T. Cai, H.-N. Dai, Z. Zheng, and Y. Zhang, "Deep Reinforcement Learning for Internet of Things: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1659–1692, 2021.

[181] L. Lei, Y. Tan, K. Zheng, S. Liu, K. Zhang, and X. Shen, "Deep Reinforcement Learning for Autonomous Internet of Things: Model, Applications and Challenges," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1722–1760, 2020.

[182] H. van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-Learning," in *Proc. AAAI Conf. Artificial Intelligence*, vol. 30, pp. 2094–2100, 2016.

[183] A. Robles-Enciso and A. F. Skarmeta, "A Multi-Layer Guided Reinforcement Learning-Based Tasks Offloading in Edge Computing," *Computer Networks*, vol. 220, p. 109476, 2023.

[184] Y. Chen, Y. Sun, C. Wang, and T. Taleb, "Dynamic Task Allocation and Service Migration in Edge-Cloud IoT System Based on Deep Reinforcement Learning," *IEEE Internet of Things Journal*, vol. 9, pp. 16742–16757, 2022.

[185] C. Sun, X. Li, C. Wang, Q. He, X. Wang, and V. C. M. Leung, "Hierarchical Deep Reinforcement Learning for Joint Service Caching and Computation Offloading in Mobile Edge-Cloud Computing," *IEEE Transactions on Services Computing*, 2024.

[186] P. Zhang, N. Chen, G. Xu, N. Kumar, A. Barnawi, M. Guizani, Y. Duan, and K. Yu, "Multi-Target-Aware Dynamic Resource Scheduling for Cloud-Fog-Edge Multi-Tier Computing Network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 5, pp. 3885–3897, 2024.

[187] S. Shahhosseini, D. Seo, A. Kanduri, T. Hu, S.-S. Lim, B. Donyanavard, A. M. Rahmani, and N. Dutt, "Online Learning for Orchestration of Inference in Multi-User End-Edge-Cloud Networks," *ACM Transactions on Embedded Computing Systems*, vol. 21, no. 6, pp. 1–25, 2022.

[188] K. Moghaddasi, S. Rajabi, and F. S. Gharehchopogh, "An Advanced Deep Reinforcement Learning Algorithm for Three-Layer D2D-Edge-Cloud Computing Architecture for Efficient Task Offloading in the Internet of Things," *Sustainable Computing: Informatics and Systems*, 2024.

[189] S. Shresthamali, M. Kondo, and H. Nakamura, "Multi-Objective Resource Scheduling for IoT Systems Using Reinforcement Learning," *Journal of Low Power Electronics and Applications*, vol. 12, no. 4, p. 53, 2022.

[190] J. Armalaite, M. Jarutiene, A. Vasiliauskas, A. Sidlauskas, V. Svalkauskiene, M. Sidlauskas, and G. Skarbalius, "Smile Aesthetics as Perceived by Dental Students: A Cross-Sectional Study," *BMC Oral Health*, vol. 18, no. 1, p. 225, 2018.

[191] A. Lukez, A. Pavlic, M. Trinajstic Zrinski, and S. Spalj, "The Unique Contribution of Elements of Smile Aesthetics to Psychosocial Well-Being," *Journal of Oral Rehabilitation*, vol. 42, no. 4, pp. 275–281, 2015.

[192] M. Luca, A. Luca, C. M. A. V. Grasso, and C. Calandra, "Nothing to Smile About," *Neuropsychiatric Disease and Treatment*, vol. 10, pp. 1999–2008, 2014.

[193] J. May, P. V. Bussen, and D. M. Steinbacher, *Smile Aesthetics*, ch. 11, pp. 253–287. John Wiley & Sons, Ltd, 2019.

[194] R. Saini, N. Thakur, R. Jindal Goyal, K. Sharma Rai, H. Bagde, and A. Dhopte, "Analysis of Smile Aesthetic Changes with Fixed Orthodontic Treatment," *Cureus*, vol. 14, no. 12, p. e32612, 2022.

[195] M. Khan, S. M. R. Kazmi, F. R. Khan, and I. Samejo, "Analysis of Different Characteristics of Smile," *BDJ Open*, vol. 6, no. 1, p. 6, 2020.

[196] S. Desai, M. Upadhyay, and R. Nanda, "Dynamic Smile Analysis: Changes with Age," *American Journal of Orthodontics and Dentofacial Orthopedics*, vol. 136, no. 3, pp. 310.e1–310.e10, 2009.

[197] G. Arroyo-Cruz, A. Orozco-Varo, M. Vilches-Ahumada, and E. Jiménez-Castellanos, "Comparative Analysis of Smile Esthetics Between Top Celebrity Smile and a Southern European Population," *The Journal of Prosthetic Dentistry*, vol. 128, no. 6, pp. 1336–1341, 2022.

[198] A. W. Machado, "10 Commandments of Smile Esthetics," *Dental Press Journal of Orthodontics*, vol. 19, no. 4, pp. 136–157, 2014.

[199] S. Akyalcin, L. K. Frels, J. D. English, and S. Laman, "Analysis of Smile Esthetics in American Board of Orthodontic Patients," *The Angle Orthodontist*, vol. 84, no. 3, pp. 486–491, 2014.

[200] L. K. Ríos, "Laypeople's perceptions of smile esthetics: Why is it important and what do we need to know?," *Journal of Oral Research*, vol. Perspectives S3, no. 1, pp. 27–29, 2020.

[201] S. H. Sajjadi, B. Khosravanifard, F. Moazzami, V. Rakhshan, and M. Esmaeilpour, "Effects of Three Types of Digital Camera Sensors on Dental Specialists' Perception of Smile Esthetics: A Preliminary Double-Blind Clinical Trial," *Journal of Prosthodontics*, vol. 25, no. 8, pp. 675–681, 2016.

[202] S. H. Sajjadi, B. Khosravanifard, M. Esmaeilpour, V. Rakhshan, and F. Moazzami, "The Effects of Camera Lenses and Dental Specialties on the Perception of Smile Esthetics," *Journal of Orthodontic Science*, vol. 4, no. 4, pp. 97–101, 2015.

[203] S. H. Agou, "Comparison of Digital and Paper Assessment of Smile Aesthetics Perception," *Journal of International Society of Preventive & Community Dentistry*, vol. 10, no. 5, pp. 659–665, 2020.

[204] G. Zhu, Z. Piao, and S. C. Kim, "Tooth Detection and Segmentation with Mask R-CNN," in *Proc. Int. Conf. Artificial Intelligence in Information and Communication (ICAIIC)*, pp. 70–72, 2020.

[205] S. Lee and J.-E. Kim, "Evaluating the Precision of Automatic Segmentation of Teeth, Gingiva and Facial Landmarks for 2D Digital Smile Design Using Real-Time Instance Segmentation Network," *Journal of Clinical Medicine*, vol. 11, no. 3, p. 852, 2022.

[206] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

[207] C. Attig, N. Rauh, T. Franke, and J. F. Krems, "System Latency Guidelines Then and Now – Is Zero Latency Really Considered Necessary?," in *Engineering Psychology and Cognitive Ergonomics: Cognition and Design*, pp. 3–14, Springer, 2017.

[208] Student, "The Probable Error of a Mean," *Biometrika*, vol. 6, no. 1, pp. 1–25, 1908.

[209] M. Hartmann, U. S. Hashmi, and A. Imran, "Edge Computing in Smart Health Care Systems: Review, Challenges, and Research Directions," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, p. e3710, 2022.

[210] Janani R. P., Renuka K., Aruna A., and Lakshmi Narayanan K., "IoT in Smart Cities: A Contemporary Survey," *Global Transitions Proceedings*, vol. 2, no. 2, pp. 187–193, 2021.

[211] S. Bousnina, M. Cesana, J. Ortín, C. Delgado, J. R. Gállego, and M. Canales, "A Greedy Approach for Resource Allocation in Virtual Sensor Networks," in *Proc. Wireless Days*, pp. 15–20, 2017.

[212] O. A. Mahmood, A. R. Abdellah, A. Muthanna, and A. Koucheryavy, "Distributed Edge Computing for Resource Allocation in Smart Cities Based on the IoT," *Information*, vol. 13, no. 7, p. 328, 2022.

[213] D. Liu, A. Hafid, and L. Khoukhi, "Workload Balancing in Mobile Edge Computing for Internet of Things: A Population Game Approach," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 3, pp. 1726–1739, 2022.

[214] P. Yang, R. Ma, M. Yi, Y. Zhang, B. Li, and Z. Bai, "A Computation Offloading Strategy for Multi-Access Edge Computing Based on DQUIC Protocol," *The Journal of Supercomputing*, vol. 80, no. 12, pp. 18285–18318, 2024.

[215] Amazon Web Services, "Amazon EC2 Instance Types." `https://aws.amazon.com/ec2/instance-types/`, 2024. [Online]. Accessed: Jan. 2025.

[216] Google Cloud, "Google Cloud Compute Engine Machine Types." `https://cloud.google.com/compute/docs/machine-types`, 2024. [Online]. Accessed: Jun. 2025.

[217] NVIDIA Corporation, "NVIDIA Jetson Modules." `https://developer.nvidia.com/embedded/jetson-modules`, 2024. [Online]. Accessed: Jun. 2025.

[218] Raspberry Pi Foundation, "Raspberry Pi Products." `https://www.raspberrypi.com/products/`, 2024. [Online]. Accessed: Jun. 2025.

[219] L. A. Barroso, J. Clidaras, and U. Hölzle, "The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines," *Synthesis Lectures on Computer Architecture*, vol. 8, no. 3, pp. 1–154, 2013.

[220] A. Pantelopoulos and N. G. Bourbakis, "A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 40, no. 1, pp. 1–12, 2010.

[221] G. Litjens, T. Kooi, B. E. Bejnordi, *et al.*, "A Survey on Deep Learning in Medical Image Analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.

[222] R. Behringer, S. Feiner, S. Julier, and B. MacIntyre, "Recent Advances in Augmented Reality," *IEEE Computer Graphics and Applications*, vol. 21, no. 6, pp. 34–47, 2001.

[223] J. Nielsen, *Usability Engineering*. Academic Press, 1993.

[224] A. Esteva, A. Robicquet, B. Ramsundar, *et al.*, "A Guide to Deep Learning in Healthcare," *Nature Medicine*, vol. 25, no. 1, pp. 24–29, 2019.

[225] R. P. Dobrow, "Poisson Process," in *Introduction to Stochastic Processes With R*, John Wiley & Sons, 2016.

[226] L. Turchet, G. Fazekas, M. Lagrange, H. S. Ghadikolaei, and C. Fischione, "The Internet of Audio Things: State of the Art, Vision, and Challenges," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10233–10249, 2020.

[227] H. Jiang, X. Dai, Z. Xiao, and A. Iyengar, "Joint Task Offloading and Resource Allocation for Energy-Constrained Mobile Edge Computing," *IEEE Transactions on Mobile Computing*, vol. 22, pp. 4000–4015, 2023.

[228] J. Mills, G. Kernohan, K. Pedlow, and O. Duffy, "Voice-Assisted Technology for People With Parkinson's Disease Experiencing Speech and Voice Difficulties: Co-Designing Solutions Using Design Thinking," *JMIR Rehabilitation and Assistive Technologies*, vol. 13, p. e84364, 2026.

[229] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello Edge: Keyword Spotting on Microcontrollers," *arXiv preprint arXiv:1711.07128*, 2017.

[230] P. M. Sørensen, B. Epp, and T. May, "A Depthwise Separable Convolutional Neural Network for Keyword Spotting on an Embedded System," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2020, no. 1, p. 10, 2020.

[231] ITU-T, "One-Way Transmission Time," Tech. Rep. ITU-T Recommendation G.114, International Telecommunication Union, 2003.

[232] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A Survey of Multi-Objective Sequential Decision-Making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.