Performance Modeling for Deadline-Aware Multi-Task Edge-AI Inference in Wearable IoT Systems

Neil Joshua Limbaga, Osama Amin, *Senior Member, IEEE*, Khaled Nabil Salama, *Senior Member, IEEE*, and Basem Shihada, *Senior Member, IEEE*

***Abstract*—Real-time wearable systems face a fundamental tension between deploying accurate AI inference models on resource-constrained edge processors and guaranteeing that physiological insights are delivered within clinically meaningful deadlines. This work presents a deadline-aware performance model for wearable edge inference using an M/G/1/k priority queue with load-dependent service vacations, characterizing end-to-end delay, stability and energy by linking sensing load, model computational cost and processor capacity. Validation against trace-driven simulations across profiled fatigue detection and oxygen uptake estimation models confirms mean delay prediction errors below 10% under well-provisioned, low-utilization conditions with robustness confirmed under correlated bursty arrivals, and energy predictions accurate across all load regimes. For concurrent multi-task execution, progressive task isolation is formally proven to strictly reduce per-task delay along a specified isolation sequence. Furthermore, a validated energy-competitiveness frontier derived from the closed-form relative energy-overhead expression allows the partitioning of the design space into three regions that help guide initial architectural decisions before hardware prototyping.**

***Index Terms*—Edge-AI, wearable IoT, deadline-aware scheduling, energy-latency tradeoff, multi-task edge computing**

I. Introduction

INTELLIGENT wearable systems have emerged as a critical class of Internet-of-Things (IoT) devices, integrating multimodal sensing, embedded machine learning, and wireless communication to enable continuous physiological monitoring in health and sports environments [1]. The integration of artificial intelligence (AI) into such IoT infrastructure has further enhanced the capabilities of wearable systems to process physiological data at the network edge [2]. Recent advances demonstrate this capability, from IoT-enabled blood glucose prediction on embedded devices [3], to deep learning frameworks for personalized cardiac monitoring via wearable systems [4], showcasing the fundamental impact of intelligent wearable systems.

Within this landscape of AI-enabled healthcare IoT are time-critical physiological monitoring systems that require low-latency processing to prevent severe and often irreversible outcomes, such as detecting cardiac emergencies [5], preventing fatigue-related injuries [6], [7], or enabling preventive monitoring in athletic performance [8], [9]. However, the usefulness of these physiological insights depends entirely on whether wearable systems can infer in real time [10], [11], as even systems with sensors capable of capturing rich multimodal data remain ineffective if embedded models cannot produce timely predictions [12]. Multiple reviews of wearable AI systems emphasize that inference delays directly undermine applications where actionable decisions must occur within strict deadlines [13], yet latency predictability and real-time guarantees remain fundamental challenges across IoT and cyber–physical systems, where heterogeneous sensing and compute workloads must meet strict timing requirements [14], [15]. Related edge-health studies have explored wearable-sensor inference using local edge processing [16] and Mobile Edge Computing (MEC)-enabled decentralized health monitoring for the Internet of Medical Things (IoMT) [17]. However, these approaches primarily address application-level edge deployment, task offloading, or system-wide cost optimization, rather than explicitly characterizing compute-aware queueing delay, deadline feasibility, and architecture-level tradeoffs for resource-constrained wearable inference [18].

Further narrowing to the wearable system use case, real-time inference is also constrained by limited battery capacity and compute budgets where even state-of-the-art deep-learning models may consume several joules per inference, making continuous operation impractical without careful optimization [19], [20]. Generally, improving accuracy metrics, measured by F1-score for classification tasks and R^2 for estimation tasks, typically increases computational cost, which increases latency and energy consumption, whereas reducing the computational cost risks degrading prediction quality until the model becomes operationally ineffective [21], [22].

Developing and validating such systems is further complicated by the high cost of hardware prototyping. Selecting an appropriate processor, estimating its effective throughput, and matching it to the model's computational demands requires iterative testing with real sensor loads, incurring material cost, development time, and firmware complexity. Thus, before committing to hardware, there is a need for a principled analytical framework to estimate whether a candidate edge processor can meet latency and energy requirements under realistic workloads.

Queueing-theoretic approaches have modeled throughput and stability in communication networks [23], and have more recently been extended to IoT systems where heterogeneous traffic and variable service demands introduce non-trivial delay

Neil Joshua Limbaga, Osama Amin, Khaled Nabil Salama, and Basem Shihada are with the Computer, Electrical and Mathematical Science and Engineering Division, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia. E-mail: {neil.limbaga, osama.amin, khaled.salama, basem.shihada}@kaust.edu.sa.

Khaled Nabil Salama is also with the College of Engineering and Computer Science, VinUniversity, Hanoi 100000, Vietnam.

Copyright (c) 20xx IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

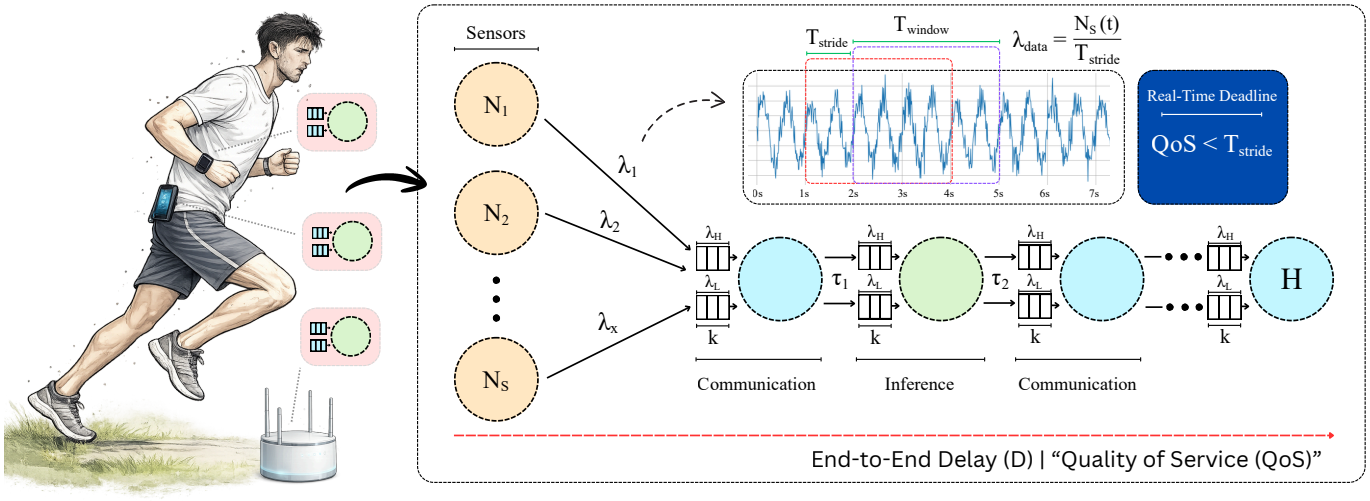


Fig. 1. System architecture of an AI-enabled wearable IoT node modeled as an M/G/1/k queue, capturing stochastic sensing arrivals, embedded compute resources, multi-task interference, and real-time deadline constraints in edge intelligence deployments.

behavior [24], [25]. Further narrowing to the gateway level, Pagliari et al. [26] demonstrate that concurrent services across heterogeneous interfaces introduce queueing contention that produces variable sojourn times, even when individual service rates appear sufficient in isolation. At the node level, the same contention produces load-dependent latency that cannot be captured by static benchmarks, as sensing, preprocessing, and communication services compete on a shared processor [27]. This is further complicated with multi-task interference on wearable platforms, which causes the inference latency to fluctuate unpredictably under realistic workloads [28]. Hence, inference latency is not fixed but emerges from queueing effects driven by competing tasks, stochastic service demands, and fluctuating sensing loads. However, none of these models capture compute-aware inference cost, energy–latency trade-offs, or deadline feasibility within IoT nodes, which are essential under stochastic edge-AI workloads.

This work closes that gap with a deadline-aware queueing performance model for wearable edge inference, linking sensing load, model cost, and processor capacity to end-to-end delay, energy, and stability. The main contributions are:

- **Queue-based wearable system performance model:** Analytical delay and energy predictions derived under Poisson arrival assumptions and validated under bursty arrival conditions across profiled physiological monitoring models.
- **Distributed isolation gain:** A formal proof that progressive task isolation strictly reduces per-task end-to-end delay along a specified isolation sequence in concurrent multi-task inference.
- **Closed-form energy-competitiveness frontier:** By analytically deriving the centralized–distributed energy gap, we obtain a closed-form energy-competitiveness frontier, providing insights on architecture selection.
- **Operating-plane architecture selection:** Combining the energy-competitiveness frontier with the centralized stability boundary, we partition the design space to map configurations for initial architecture decision-making or pre-deployment guidance.

The remainder of this paper is organized as follows: Section II presents the system model and analytical framework. Section III derives the closed-form delay and energy expressions, distributed isolation gain, and energy-competitiveness frontier. Section IV describes the simulation methodology and model profiling approach. Section V presents results on analytical–simulation validation, and multi-task architectural comparison. Section VI concludes the paper.

II. System Design Overview

Fig. 1 illustrates the proposed wearable inference pipeline spanning communication, on-device inference, and downstream buffering stages. The system workflow is as follows: (1) Multiple heterogeneous sensors generate inference windows, fixed-length segments of raw sensor data, to be processed into physiological metrics, at fixed stride intervals T_{stride} where each window constitutes a discrete job submitted to the queueing model for inference. (2) These jobs propagate through H communication and inference stages. (3) Each stage is modeled as an M/G/1/k queue [29] where M denotes Poisson arrivals, G denotes general service times, 1 denotes a single processor or relay, and k denotes finite buffer capacity, with arrivals dropped if the buffer is full. For analytical tractability, the closed-form waiting-time expressions approximate each finite-buffer M/G/1/k node by its corresponding infinite-buffer M/G/1 priority queue, while the finite capacity k and associated overflow behavior are retained explicitly in the simulation. Table I summarizes the notation used throughout this paper.

The system is characterized by three core parameters: sensing load (λ), computational cost (C), and processor capacity (F).

1) Sensing Load (λ): Each sensor outputs one inference window every T_{stride} seconds, producing a time-varying Poisson arrival rate:

$$\lambda_{\text{data}}(t) = \frac{N_s(t)}{T_{\text{stride}}} \tag{1}$$

TABLE I
NOTATION SUMMARY

Symbol	Definition
System Parameters	
T	Number of concurrent inference tasks
τ	Task index, $\tau \in \{1, \ldots, T\}$
θ	Number of deployed processors
H	Number of communication hops
k	Finite buffer capacity per queue
$T_{\text{stride}}^{(\tau)}$	Sensing stride interval for task τ (s)
T_{win}	Inference window duration (s)
f_s	Sensor sampling frequency (Hz)
Arrival and Traffic	
$N_s^{(\tau)}(t)$	Active sensor count for task τ at time t
$\lambda_{\text{data}}^{(\tau)}(t)$	Data arrival rate for task τ (jobs/s)
λ_{control}	Control packet arrival rate (jobs/s)
λ_{H},λ_{L}	High- and low-priority arrival rates (jobs/s)
$\bar{\lambda}_{\text{data}}$	Aggregate multi-task data arrival rate (jobs/s)
$\lambda_{\text{eff}}^{(\tau)}(\theta, t)$	Effective sensing load on task τ's node (jobs/s)
λ^*	Stability limit arrival rate (jobs/s)
p_t	Probability a data packet is low-priority
$\gamma^{(\tau)}$	Fractional arrival weight of task τ
Computation and Service	
$C^{(\tau)}$	Model FLOPs per inference window for task τ
$\bar{C}$	Arrival-weighted average computational cost
F	Processor capacity (FLOPs/s)
α_{model}	Average operations per parameter per sample
P_{model}	Number of model parameters
μ_{comm}	Communication service rate (jobs/s)
μ_{vac}	Vacation end rate (s^{-1})
Queueing and Delay	
ρ, ρ_{H}, ρ_{L}	Total system, high- and low-priority class utilization
$\rho^{(\tau)}(\theta)$	Per-task utilization under architecture θ
R	Mean residual service time (s)
$R_{\text{V}}(\rho_{\text{H}})$	Load-dependent vacation overhead (s)
W_{H}, W_{L}	Mean waiting time, high/low priority (s)
D^{relay}	Relay node mean sojourn time (s)
$D_{\text{data}}^{(\tau)}(\theta)$	Per-task data-window delay under architecture θ (s)
$D^{\text{e2e},(\tau)}(\theta)$	Per-task end-to-end delay under architecture θ (s)
Energy	
P_{active}	Active-state processor power (W)
P_{sleep}	Sleep-state processor power (W)
P_{sensor}	Per-sensor power allowance (W)
P_{tx}	Transmission power per hop (W)
$\Phi_\theta^{(\tau)}$	Per-processor power draw under θ (W)
$E_{\text{per job}}(\theta)$	System-level energy per arriving job (J)
Architecture Thresholds and Frontiers	
ε	Engineering tolerance for energy competitiveness
ρ^*	Utilization threshold for energy competitiveness
η^*	C/F coordinate of the energy-competitiveness frontier
E^*	Energy per job at the energy-competitiveness frontier (J)
ΔE	Energy gap: distributed minus centralized (J)
$\bar{N}_s$	Average active sensor count across tasks

where $N_s(t)$ denotes the number of active sensors at time t. When $N_s(t) = 0$, no inference windows are generated and $\lambda_{\text{data}}(t) = 0$, eliminating the data-traffic contribution to utilization. The sensing load is time-varying at runtime as sensors activate or deactivate.

2) Computational Cost (C): For a model with P_{model} parameters operating on a window of duration T_{win} sampled at f_s, the total FLOPs per inference are

$$C = \alpha_{\text{model}}\, P_{\text{model}}\, f_s\, T_{\text{win}}, \tag{2}$$

where $L = f_s T_{\text{win}}$ denotes the input sequence length and α_{model} is the effective model-level operation factor, normalized by the model parameter count and input sequence length for the specified architecture and input configuration.

3) Processor Capacity (F): The embedded processor's capability is summarized by its effective compute rate F (FLOPs/s) which determines the mean service time per inference,

$$\mathbb{E}[S] = \frac{C}{F}, \qquad \mu = \frac{1}{\mathbb{E}[S]} = \frac{F}{C}. \tag{3}$$

Queue stability requires

$$\rho = \frac{\lambda}{\mu} = \lambda \frac{C}{F} < 1 \quad \Longrightarrow \quad F > \lambda C, \tag{4}$$

showing that the tuple (λ, C, F) fundamentally determines whether real-time performance is achievable.

A. Modeling Assumptions

The analytical model assumes independent Poisson arrival for data (λ_{data}) and control (λ_{control}) traffic, a standard model for sensor-driven IoT systems where sensing events occur independently across heterogeneous devices [25], [30]. Service times follow a general distribution: communication latency per hop is exponential $S_{\text{comm}} \sim \text{Exp}(\mu_{\text{comm}})$, consistent with wireless hop latency models in IoT gateways [26], so that the aggregate communication latency across H independent hops follows an $\text{Erlang}(H, \mu_{\text{comm}})$ distribution, and inference time is deterministic $S_{\text{inf}} = C/F$, appropriate for fixed-graph models on dedicated hardware, where every inference job executes the same computation graph at a fixed clock frequency, with negligible execution time variance under deterministic RTOS scheduling on embedded wearable processors [28]. The system operates under non-preemptive priority scheduling as each fixed-graph neural network inference job is executed to completion once dispatched on the target ARM Cortex-M-class processor, making mid-execution preemption impractical under embedded memory and compute constraints. The processor alternates between active and sleep states with exponential vacation times $V \sim \text{Exp}(\mu_{\text{vac}})$. For multi-stage pipelines, the underlying system retains the M/G/1/k finite-buffer representation, while each stage is treated as an independent M/G/1 node in the closed-form analysis; output processes remain approximately Poisson under short vacation durations, preserving analytical tractability — a standard decomposition result for lightly-loaded multi-stage queueing networks [29].

B. Priority Traffic Model

Inspired by the analytical framework of Alabdulmohsin *et al.* [30], the system handles two traffic classes: high-priority control packets, λ_{control} (synchronization, MCU coordination, safety), and low-priority data packets, $\lambda_{\text{data}}(t)$ (inference windows). Control packets are protocol-driven and always assigned high priority, as they carry system-level coordination signals that must be processed immediately. Data packets are

application-driven: a sensing window that crosses a clinical threshold warrants immediate inference and may be promoted to high priority, while routine monitoring windows are processed at low priority. Data windows may be elevated to high-priority under urgent conditions with probability $(1 - p_t)$, yielding:

$$\lambda_H = \lambda_{\text{control}} + (1 - p_t)\,\lambda_{\text{data}}(t) \tag{5}$$

$$\lambda_L = p_t\,\lambda_{\text{data}}(t) \tag{6}$$

Although control packets and urgent data windows share the same high-priority queue, their service requirements differ. Control packets require communication service only, whereas data windows, regardless of priority queue, require both communication and inference service. Hence, the mean service time for control packets is $E[S_{\text{control}}] = 1/\mu_{\text{comm}}$, while the mean service time for data windows is $E[S_{\text{data}}] = 1/\mu_{\text{comm}} + C/F$, respectively.

The priority promotion mechanism accommodates urgent physiological events by elevating inference windows to high priority. Under non-preemptive scheduling, an arriving high-priority job may experience additional waiting due to the residual service time of a job already in service, which includes the applicable communication and inference service components. Since control packets and urgent data windows have distinct service requirements, the high-priority utilization is written directly as $\rho_H = \lambda_{\text{control}} E[S_{\text{control}}] + (1 - p_t)\lambda_{\text{data}}(t)E[S_{\text{data}}]$, while the low-priority utilization is $\rho_L = p_t \lambda_{\text{data}}(t) E[S_{\text{data}}]$, such that $\rho = \rho_H + \rho_L$. Expanding these terms recovers the total system utilization:

$$\rho = \frac{\lambda_{\text{control}}}{\mu_{\text{comm}}} + \lambda_{\text{data}}(t)\left(\frac{1}{\mu_{\text{comm}}} + \frac{C}{F}\right) \tag{7}$$

such that $\rho = \rho_H + \rho_L < 1$ is within stable conditions. Here, urgent high-priority data windows contribute both communication and inference service to ρ_H, routine data windows contribute both communication and inference service to ρ_L, while control packets contribute communication service only.

C. Multi-Task Extension

The wearable system executes $\mathcal{T} = \{1, 2, \ldots, T\}$ concurrent inference tasks, each characterized by a sensing stride $T_{\text{stride}}^{(\tau)}$, model complexity $C^{(\tau)}$, and active sensor count $N_s^{(\tau)}(t)$. The execution architecture is governed by $\theta \in \{1, \ldots, T\}$, the number of deployed distributed processing nodes, which is an architectural decision independent of T.

Centralized, distributed, and hybrid task-allocation strategies have been considered in edge-computing systems [31]. Here, the closed-form comparison focuses on two boundary configurations: centralized execution ($\theta = 1$), where all tasks share one processor, and fully distributed execution ($\theta = T$), where each task has a dedicated processor, eliminating cross-task interference. Intermediate hybrid configurations ($1 < \theta < T$) remain admissible, but their performance depends on task-to-processor grouping, requiring assignment-aware optimization that is left for future work.

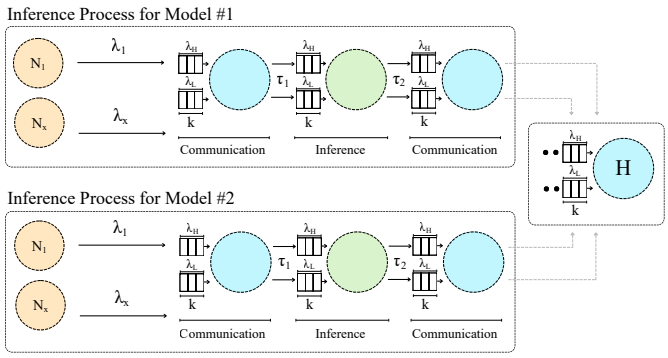


Fig. 2. Distributed multi-task execution architectures for AI-enabled wearable systems for two tasks ($T = 2$), where each task has its own designated processor, $\theta = T$

The effective sensing load seen by the processor handling task τ under architecture θ is:

$$\lambda_{\text{eff}}^{(\tau)}(\theta, t) = \sum_{\tau' \text{ co-located with } \tau} \lambda_{\text{data}}^{(\tau')}(t) \tag{8}$$

where the sum is over all tasks sharing the same node as τ, including τ itself. As θ increases, fewer tasks share each processor, reducing cross-task interference.

III. Performance Analysis

A. Analytical Delay Analysis

The end-to-end pipeline consists of H sequential hops: one inference node and $H-1$ communication-only relay nodes. For tractable closed-form analysis, the finite-buffer M/G/1/k system is approximated by its corresponding infinite-buffer M/G/1 priority queue; such approximations are commonly adopted in the analysis of finite-capacity M/G/1/k systems [32]. Consequently, the finite-capacity parameter k does not appear explicitly in the analytical waiting-time expressions. For the inference node delay, under non-preemptive priority with mean residual service time R and utilization-dependent vacation overhead $R_V(\rho_H)$, the mean waiting times are determined by the mixed control and data traffic.

The mean residual-service term explicitly accounts for their distinct service-time distributions as,

$$R = \frac{1}{2}\left[\lambda_{\text{control}} E[S_{\text{control}}^2] + \lambda_{\text{data}}(t) E[S_{\text{data}}^2]\right]. \tag{9}$$

with the adopted exponential communication service and deterministic inference service, $E[S_{\text{control}}^2] = 2/\mu_{\text{comm}}^2$ and $E[S_{\text{data}}^2] = 2/\mu_{\text{comm}}^2 + 2C/(F\mu_{\text{comm}}) + (C/F)^2$. Since urgent and routine data windows have the same service-time distribution, their contributions combine through the total data arrival rate $\lambda_{\text{data}}(t)$. The mean waiting times are then,

$$W_{\text{H}} = \frac{R + R_{\text{V}}(\rho_{\text{H}})}{1 - \rho_{\text{H}}} \tag{10}$$

$$W_{\text{L}} = \frac{R + R_{\text{V}}(\rho_{\text{H}})}{(1 - \rho_{\text{H}})(1 - \rho)} \tag{11}$$

where the load-dependent vacation overhead is:

$$R_{\text{V}}(\rho_{\text{H}}) = \frac{\rho_{\text{H}}}{\mu_{\text{vac}}(1 - \rho_{\text{H}})} \tag{12}$$

This vanishes as $\rho_H \to 0$, reflecting negligible queue buildup during vacations at low load, and diverges as $\rho_H \to 1$, capturing the vacation-induced backlog near saturation consistent with embedded processor sleep-state behavior.

Under the adopted non-preemptive priority discipline, a low-priority job already in service is not interrupted. However, queued low-priority data windows are delayed by both high-priority control packets and urgent high-priority data windows. This interaction is explicitly captured through the dependence of W_L on ρ_H in Eq. (11). Since the high-priority queue contains both control packets and urgent inference windows with different service requirements, their sojourn times are distinguished explicitly according to traffic type:

$$D_{\text{H,control}} = W_{\text{H}} + E[S_{\text{control}}], \tag{13}$$

$$D_{\text{H,data}} = W_{\text{H}} + E[S_{\text{data}}], \tag{14}$$

$$D_{\text{L,data}} = W_{\text{L}} + E[S_{\text{data}}]. \tag{15}$$

Thus, control packets and urgent data windows share the same high-priority waiting time W_H, while their service requirements are retained through $E[S_{\text{control}}]$ and $E[S_{\text{data}}]$, respectively. Routine data windows experience the low-priority waiting time W_L with service requirement $E[S_{\text{data}}]$.

For communication-only relay nodes, all packets are served FIFO with no inference stage. The relay utilization ρ_{relay} is simply the total arrival rate normalized by the communication service rate. For exponential relay service, the second moment of the service time is $E[S^2_{\text{relay}}] = 2/\mu^2_{\text{comm}}$. With $\rho_{\text{relay}} = (\lambda_{\text{control}} + \lambda_{\text{data}}(t))/\mu_{\text{comm}}$, the residual-work term becomes $R_{\text{relay}} = \rho_{\text{relay}}/\mu_{\text{comm}}$. Accordingly, the mean relay sojourn time is

$$D^{\text{relay}} = \frac{\rho_{\text{relay}}/\mu_{\text{comm}}}{1-\rho_{\text{relay}}} + \frac{1}{\mu_{\text{comm}}} = \frac{1}{\mu_{\text{comm}}(1-\rho_{\text{relay}})} \tag{16}$$

The overall processing-node delay is obtained by arrival-weighting the traffic-specific sojourn times of control packets, urgent data windows, and routine data windows:

$$\begin{aligned} D^{\text{node}} = {} & \frac{\lambda_{\text{control}} D_{\text{H,control}} + (1-p_t)\lambda_{\text{data}}(t) D_{\text{H,data}}}{\lambda_{\text{control}} + \lambda_{\text{data}}(t)} \\ & + \frac{p_t \lambda_{\text{data}}(t) D_{\text{L,data}}}{\lambda_{\text{control}} + \lambda_{\text{data}}(t)}. \end{aligned} \tag{17}$$

The total end-to-end (e2e) delay then accumulates the processing-node delay and the delay across the subsequent $H-1$ relay hops:

$$D^{\text{e2e}} = D^{\text{node}} + (H-1)D^{\text{relay}}. \tag{18}$$

For inference-window timing, we separately define the data-window delay by averaging only urgent and routine data-window sojourn times according to their respective priority probabilities:

$$D_{\text{data}} = (1-p_t)D_{\text{H,data}} + p_t D_{\text{L,data}} + (H-1)D^{\text{relay}}. \tag{19}$$

This separates inference-window latency from the overall system-level delay D^{e2e} by excluding control-packet sojourn times from the reported data-window average. Control traffic nevertheless continues to affect D_{data} through the residual-service term R, high-priority utilization ρ_H, and the resulting waiting times W_H and W_L.

Since T_{stride} specifies the timing requirement associated with successive inference windows, deadline feasibility is evaluated using the data-window delay:

$$D_{\text{data}} \le T_{\text{stride}} \tag{20}$$

The analysis is valid when each hop operates below saturation ($\rho \ll 1$) and vacation durations are short relative to service times, so that output processes approximate independent Poisson streams across hops. The approximation degrades near $\rho \to 1$, where departure processes become bursty and correlated, or under long vacation times that introduce synchronized wake-up patterns violating the independence assumption. Real-time feasibility is achieved whenever Eq. (20) holds across all operating points in the $(\bar{\lambda}_{\text{data}}, C/F)$ plane.

The single-task delay characterization above extends naturally to the multi-task setting by substituting the per-architecture effective load, replacing $\lambda_{\text{data}}(t) \leftarrow \lambda^{(\tau)}_{\text{eff}}(\theta, t)$ and $C \leftarrow C^{(\tau)}$ into Eq. (7) yields the per-architecture utilization. For the centralized case ($\theta = 1$), all T tasks aggregate onto a single node, giving:

$$\rho(\theta=1) = \frac{\lambda_{\text{control}}}{\mu_{\text{comm}}} + \bar{\lambda}_{\text{data}}(t)\left(\frac{1}{\mu_{\text{comm}}} + \frac{\bar{C}}{F}\right) \tag{21}$$

where $\bar{\lambda}_{\text{data}} \triangleq \sum_{\tau=1}^{T} \lambda^{(\tau)}_{\text{data}}$ denotes the aggregate multi-task data arrival rate, and $\bar{C}$ denotes the arrival-weighted average computational cost across co-located tasks. The arrival-weighted average computational cost $\bar{C}$ is sufficient for utilization in Eq. (21), which depends on the mean service time. However, the residual-service term in Eq. (9) depends on the second moment, which is computed by weighting the individual task-specific second moments by their respective arrival rates, rather than using the average computational cost. For the distributed case ($\theta = T$), each node sees only its own task load $\lambda^{(\tau)}_{\text{data}}$ and $C^{(\tau)}$, so $\rho^{(\tau)}(T)$ follows directly by substitution. The full delay analysis of (10)–(18) applies under each architecture. Likewise, the data-window delay $D^{(\tau)}_{\text{data}}(\theta)$ follows from Eq. (19) using the corresponding task-specific effective load and computational cost. The per-task inference-window deadline constraint is therefore:

$$D^{(\tau)}_{\textbf{data}}(\theta) \le T^{(\tau)}_{\text{stride}} \quad \forall\, \tau \in \mathcal{T} \tag{22}$$

A key consequence of the multi-task formulation is that, for a fixed task τ, delay strictly decreases along a progressive-isolation sequence that removes positive-load co-located tasks from its processor at each step. Intuitively, adding a dedicated processor removes cross-task interference, reducing both the residual service time R and the effective utilization seen by each task. Hence, we introduce the first theorem in this work,

Theorem 1 (Distributed Isolation Gain)**.** *For $T > 1$ concurrent inference tasks, consider a progressive isolation sequence in which increasing θ by one removes at least one co-located task τ' with $\lambda^{(\tau')}_{data}(t) > 0$ from the processor serving task*

τ. Under stable operation, for a fixed task τ along this progressive-isolation sequence, the per-task end-to-end delay is strictly monotone decreasing in θ:

$$D^{e2e,(\tau)}(1) > D^{e2e,(\tau)}(2) > \cdots > D^{e2e,(\tau)}(T). \tag{23}$$

At $\theta = T$, task τ has no co-located inference tasks, so $D^{e2e,(\tau)}(T)$ reduces to the single-task result of Eq. (18)*.*

The monotonicity follows because each incremental node removes at least one positive-load co-located task τ', reducing $\lambda_{\text{eff}}^{(\tau)}$ by $\lambda_{\text{data}}^{(\tau')}(t) > 0$. This strictly decreases the residual service term R, the high-priority utilization ρ_H, and the total utilization ρ in Eq. (10)–(11). Since the waiting-time expressions are strictly increasing in these quantities over the stable region, $D^{e2e,(\tau)}$ strictly decreases. Accordingly, Theorem 1 characterizes the structural effect of progressive task isolation along a specified sequence of task assignments. The intermediate terms $D(2), \ldots, D(T-1)$ represent the delay values associated with the corresponding assignments along this isolation sequence and do not imply optimization over all possible task assignments for each intermediate value $1 < \theta < T$. The formal proof is provided in Appendix A.

B. Analytical Energy Analysis

The energy consumed per arriving job accounts for processor state, active sensor power, and transmission across H hops. The power draw $\Phi_\theta^{(\tau)}$ follows a two-state active/sleep model widely adopted in edge computing task allocation and hardware-aware neural architecture search [33], [34]:

$$\begin{aligned}\Phi_\theta^{(\tau)} \triangleq\ & P_{\text{active}}\,\rho^{(\tau)}(\theta) + P_{\text{sleep}}\Big(1 - \rho^{(\tau)}(\theta)\Big) \\ & + N_s^{(\tau)}(t)\,P_{\text{sensor}}\end{aligned} \tag{24}$$

where the sensing term $N_s^{(\tau)}(t)\,P_{\text{sensor}}$ introduces a direct trade-off between sensing sampling and energy efficiency where sensor power is consumed continuously as long as $N_s^{(\tau)}(t) > 0$, independent of processor load or arrival rate. The sensing workloads remain identical across centralized and distributed configurations. For energy accounting, the analytical model applies a fixed per-sensor power allowance at each deployed processor to represent the assumed overhead of sharing sensor data in the distributed configuration.

Since θ nodes are deployed, the total modeled system power is the sum of the power draw across deployed nodes, $\sum_{j=1}^{\theta} \Phi_\theta^{(j)}$, which reduces to $\theta\Phi_\theta^{(\tau)}$ under the symmetric-task setting. Dividing by the total job arrival rate $(\lambda_{\text{control}} + \bar{\lambda}_{\text{data}})$ gives the system-level energy per arriving job — the energy consumed by all deployed processors amortized over each arrival into the system:

$$E_{\text{per job}}(\theta) = \frac{\sum_{j=1}^{\theta} \Phi_\theta^{(j)}}{\lambda_{\text{control}} + \bar{\lambda}_{\text{data}}(t)} + \frac{H\,P_{\text{tx}}}{\mu_{\text{comm}}}. \tag{25}$$

For symmetric tasks, this reduces to

$$E_{\text{per job}}(\theta) = \frac{\theta \cdot \Phi_\theta^{(\tau)}}{\lambda_{\text{control}} + \bar{\lambda}_{\text{data}}(t)} + \frac{H\,P_{\text{tx}}}{\mu_{\text{comm}}}. \tag{26}$$

Both the centralized ($\theta = 1$) and distributed ($\theta = T$) cases use the same denominator, ensuring a consistent per-arriving-job comparison across architectures. Here $HP_{\text{tx}}/\mu_{\text{comm}}$ captures the cumulative transmission energy across all H hops, independent of θ. This formulation captures utilization-driven energy variation across configurations and is suitable for pre-hardware architecture comparison, but assumes static power states and does not account for dynamic voltage/frequency scaling, memory hierarchy effects, or peripheral power; platform-specific calibration is required for absolute energy predictions.

C. Energy-Competitiveness Frontier for Architecture Selection

A natural question arising from Eq. (26) is: under what conditions does the delay-optimal distributed architecture become energy-competitive with the centralized one? The following theorem establishes that for symmetric tasks ($C^{(\tau)} = \bar{C}$, $\lambda_{\text{data}}^{(\tau)} = \bar{\lambda}_{\text{data}}/T$), this energy-competitiveness frontier admits a closed-form characterization.

Theorem 2 (Closed-Form Energy-Competitiveness Frontier)**.** *For $T > 1$ symmetric concurrent inference tasks where each task contributes $\bar{\lambda}_{data}/T$ to the aggregate arrival rate $\bar{\lambda}_{data}$ and $C^{(\tau)} = \bar{C}$, with $N_s^{(\tau)} = \bar{N}_s$, while $\theta = T$ is strictly delay-optimal (Theorem 1), the aggregate active-utilization contribution is identical under centralized and fully distributed execution. Consequently, the additional deployed-node baselines yield a strictly positive distributed-minus-centralized energy gap, while for a fixed aggregate arrival rate, the relative distributed energy overhead decreases monotonically with utilization. For any tolerance ε for which the energy-competitive set is nonempty, there exists a utilization threshold $\rho^* \in [0, 1)$ such that $\theta = T$ is energy-competitive whenever $\rho(1) \geq \rho^*$, in the sense that $\Delta E / E_{per\ job}(1) \leq \varepsilon$. From ρ^*, a corresponding frontier energy E^* and a C/F threshold η^* follow by direct analytical projection, jointly partitioning the $(\bar{\lambda}_{data}, C/F)$ operating plane into unambiguous architecture regimes.*

The energy gap follows from Eq. (26) by noting that the transmission terms cancel:

$$\begin{aligned}\Delta E\big(\rho(1)\big) &\triangleq E_{\text{per job}}(T) - E_{\text{per job}}(1) \\ &= \frac{\sum_{\tau=1}^{T} \Phi_T^{(\tau)} - \Phi_1}{\lambda_{\text{control}} + \bar{\lambda}_{\text{data}}} \\ &= \frac{(T-1)\left(P_{\text{sleep}} + \bar{N}_s P_{\text{sensor}}\right)}{\lambda_{\text{control}} + \bar{\lambda}_{\text{data}}} > 0.\end{aligned} \tag{27}$$

Under the convention $\Delta E = E_{\text{per job}}(T) - E_{\text{per job}}(1)$, $\Delta E > 0$ means distributed execution is more energy-expensive, while $\Delta E < 0$ means distributed execution is cheaper. For the symmetric case considered here, $\sum_{\tau=1}^{T} \rho_T^{(\tau)} = \rho(1)$, so the aggregate active-utilization contribution is identical under centralized and fully distributed execution. With $N_s^{(\tau)} = \bar{N}_s$, the additional deployed-node sleep-state and sensing baselines therefore produce the strictly positive gap in Eq. (27). At low utilization ($\rho(1) \to 0$), the T distributed processors

collectively spend most of their time in sleep state, making distributed execution more expensive due to the aggregate baseline overhead of T idle nodes. As $\rho(1)$ increases, the relative energy overhead decreases toward the energy-competitiveness frontier. The threshold ρ^* is defined as the first utilization at which distributed execution becomes energy-competitive within engineering tolerance ε, such that:

$$\rho^* \triangleq \min\left\{\rho(1) \in [0,1) \ : \ \frac{\Delta E\big(\rho(1)\big)}{E_{\text{per job}}(1, \rho(1))} \leq \varepsilon\right\} \tag{28}$$

Crucially, ρ^* is a utilization-space threshold. Because the centralized energy reference includes the transmission-energy term, ρ^* depends on the aggregate arrival rate $\lambda_{\text{control}} + \bar{\lambda}_{\text{data}}$, while remaining independent of the particular C/F value used to realize a given centralized utilization $\rho(1)$. However, ρ^* still depends on the fixed power, sensing, architecture, and tolerance parameters $(P_{\text{active}}, P_{\text{sleep}}, P_{\text{sensor}}, \bar{N}_s, T, \varepsilon)$, as well as the communication parameters $(H, P_{\text{tx}}, \mu_{\text{comm}})$. Similarly, η^* is a projection of ρ^* onto the workload plane and therefore necessarily depends on workload-related terms such as $\bar{\lambda}_{\text{data}}$ and λ_{control}. The monotonicity proof and existence condition for ρ^* are provided in Appendix B.

From ρ^*, the corresponding centralized energy per job at the energy-competitiveness frontier E^* follows directly:

$$\begin{aligned} E^* &\triangleq E_{\text{per job}}(1,\, \rho^*) \\ &= \frac{P_{\text{active}}\, \rho^* + P_{\text{sleep}}\,(1-\rho^*) + \bar{N}_s\, P_{\text{sensor}}}{\lambda_{\text{control}} + \bar{\lambda}_{\text{data}}} + \frac{H\, P_{\text{tx}}}{\mu_{\text{comm}}} \end{aligned} \tag{29}$$

A C/F threshold η^* follows by inverting $\rho(\theta = 1) = f(C/F, \bar{\lambda}_{\text{data}}, N_s)$ at ρ^*:

$$\eta^* \triangleq \frac{\rho^* - \lambda_{\text{control}}/\mu_{\text{comm}} - \bar{\lambda}_{\text{data}}/\mu_{\text{comm}}}{\bar{\lambda}_{\text{data}}} \tag{30}$$

The stability limit λ^* gives the maximum feasible arrival rate for $\theta = 1$ before the centralized shared processor reaches saturation:

$$\lambda^* \triangleq \frac{1 - \lambda_{\text{control}}/\mu_{\text{comm}}}{\displaystyle\sum_{\tau=1}^{T} \gamma^{(\tau)} \left(\frac{1}{\mu_{\text{comm}}} + \frac{C^{(\tau)}}{F}\right)} \tag{31}$$

Tightening ε or increasing T raises ρ^*, while higher arrival rate lowers η^*, meaning a distributed architecture becomes energy-competitive at smaller C/F ratios under heavier load. The thresholds ρ^*, η^*, and λ^* provide an initial pre-hardware architecture-selection guide, as detailed in Corollary 1. The analytical thresholds hold for any $T > 1$ under these symmetric extremes; intermediate distributions ($\theta \neq T$, $T > 1$), where the task-to-processor assignment introduces a combinatorial problem, are identified as a direction for future work.

Corollary 1 (Unified Architecture Selection Rule)**.** *Given the thresholds ρ^*, η^*, and λ^* from Eqs.* (28)*,* (30)*, and* (31)*, the operating plane partitions into three regimes:*

1) ***Designer trade-off*** *($\rho(1) < \rho^*$, equivalently $C/F < \eta^*$): $\theta = 1$ is energy-cheaper while $\theta = T$ is faster; the optimal choice depends on application requirements. A delay-critical application favors $\theta^* = T$; an energy-critical application favors $\theta^* = 1$.*
2) ***Distributed Energy-Competitive*** *($\rho^* \leq \rho(1) < 1$, equivalently $C/F \geq \eta^*$ within the centralized stable region): $\theta = T$ is strictly faster and its relative energy overhead satisfies $\Delta E / E_{\text{per job}}(1) \leq \varepsilon$; $\theta^* = T$ is the preferred choice under the adopted energy-tolerance criterion.*
3) ***Stability-constrained*** *($\bar{\lambda}_{data}(t) \geq \lambda^*$): $\theta = 1$ is infeasible; provided the distributed nodes remain stable, $\theta^* = T$ is the only feasible architecture.*

IV. Simulation Methodology

The proposed infinite-buffer M/G/1 closed-form approximation is evaluated against an M/G/1/k priority-queue simulator with service vacations, which retains the finite buffer capacity and associated overflow behavior of the modeled system. The evaluation follows a three-stage validation strategy: (i) single-task case under Poisson arrivals ($T = 1$, $\theta = 1$) to validate the analytical model, (ii) MMPP-2 burstiness sensitivity to characterize the robustness boundary under correlated arrivals, and (iii) multi-task settings for architecture comparison ($T = 2$, $\theta \in \{1, T\}$) using the validated model. The simulator sweeps across (λ, C, F), estimating end-to-end delay and energy per job from both simulation and analysis with parameters chosen to reflect realistic embedded system behavior.

To ensure practical relevance, simulation parameters are aligned with representative embedded platforms used in wearable IoT systems. Specifically, processor capacity (F) and power states are selected to reflect characteristics of ARM Cortex-M class microcontrollers and comparable edge processors [35]. The computational cost (C) is obtained via layer-wise FLOP profiling and mapped to hardware execution through effective throughput (FLOPs/s). This abstraction enables hardware-aware performance estimation without requiring platform-specific implementation, while omitting lower-level effects such as memory hierarchy, peripheral overhead, and dynamic voltage and frequency scaling (DVFS).

A. Edge-AI Model Profiling

Systematic performance modeling of deep learning inference has recently been explored in IoT system-on-chip platforms, where layer-level execution behavior is analyzed to predict throughput and hardware utilization under pipeline strategies [36]. In contrast to hardware-level microarchitectural execution modeling, the goal of this study is to construct workload-level computational cost estimates that drive queueing-based delay and energy analysis under stochastic sensing loads.

Floating-point operation counts are estimated from the reconstructed model architectures using the layer-wise operation terms and coefficients summarized in Table II. Here, α_ℓ denotes the operation coefficient associated with layer type ℓ and is distinct from the effective model-level factor α_{model}. For each evaluated architecture, P_{model} is obtained directly from the reconstructed model, while the architecture-specific layer dimensions and coefficients in Table II are used to determine the aggregate operation count. This quantity is

TABLE II
LAYER-WISE OPERATION TERMS AND COEFFICIENTS.

Layer Type	Layer-wise Operation Term	α_ℓ
Fully Connected (Linear)	in_features × out_features × seq_len	2
Convolution	kernel_size × in_channels × out_channels × out_len	2
LSTM	hidden_size (in_size + hidden_size) × seq_len × num_layers	4
Attention	seq_len × embed_dim × (seq_len + embed_dim)	4

normalized by $P_{\text{model}}L$, to obtain the effective α_{model}. The resulting P_{model}, α_{model}, and L then instantiate Eq. (2) to obtain the total inference cost C. Since α_{model} is normalized by L, the sequence length $f_s T_{\text{win}}$ is applied only once in Eq. (2).

The pipeline is instantiated using two recent edge-AI applications that report complete model configurations and input settings: (1) fatigue detection using ECG-based architectures of Mu et al. [37], a binary classification task whose model performance is reported in F1-score, and (2) oxygen uptake (VO_2) estimation using the temporal models of Yang et al. [38], a continuous estimation task whose model performance is reported in R^2. The reported architectures were reconstructed in Python 3.10.18 and PyTorch 2.5.1 using their published layer configurations, model dimensions, and input settings, while f_s and T_{win} were taken directly from the respective source studies. Computational profiling was performed using a custom Python calculation script implemented with PyTorch. The resulting P_{model}, sequence length L, effective α_{model}, and computational cost C are reported in Table III.

Table III summarizes the profiled models, spanning a broad range of sensor modalities and computational costs. The sensing stride $T_{\text{stride}} = T_{\text{win}} \times (1-0.95)$ is adopted consistently to both tasks to maximize temporal resolution while ensuring each stride delivers sufficient new sensor data for inference, as shown in Table III, fixing the per-task deadline constraint $D_{\text{data}}^{(\tau)} \leq T_{\text{stride}}^{(\tau)}$ used throughout the performance analysis.

B. Single-Task Discrete-Event Simulation

The profiled models in Table III are embedded into a discrete-event M/G/1/k simulator implementing the system model of Section II. Each profiled model is simulated in isolation ($T = 1$, $\theta = 1$) under Poisson arrivals to validate the single-task analytical model. The design space is swept across $(\lambda_{\text{data}}, C, F)$, where λ_{data} follows Eq. (1), with $N_s \in \{1, 2, 3, 4, 5, 6, 7, 10, 15, 20\}$ sensors, C taken directly from the model profiles in Table III, and F swept log-uniformly over $[50\,\text{MFLOPs/s},\ 5\,\text{GFLOPs/s}]$ across 12 points. For each configuration $(\lambda_{\text{data}}, C, F)$, 30 independent replications are run with a 100 s warm-up and 1000 s measurement horizon, yielding 95% confidence intervals with half-widths below 5% of the mean for stable configurations.

Table IV summarizes the four event types handled by the simulator. Two Poisson streams are modeled per task: a data stream at rate λ_{data} and a control stream at $\lambda_{\text{control}} = 0.5$ jobs/s, corresponding to one control packet every 2 s, consistent with BLE connection supervision intervals and MCU coordination overhead typical of ARM Cortex-M class wearable processors [35].

Upon arrival, each data job joins Q_H with probability $1-p_t$ or Q_L with probability p_t; the baseline value $p_t = 0.9$ is adopted as a modeling assumption, corresponding to 10% of data windows being promoted for urgent high-priority processing. Sensitivity analysis over $p_t \in \{0.70, 0.80, 0.90, 0.95\}$ showed mean absolute delay differences below 0.11% for fatigue detection and 0.08% for VO_2 estimation; control jobs always join Q_H. Both queues have finite capacity $k = 5$, reflecting typical embedded memory constraints where larger buffers amplify queueing delays that violate real-time deadlines. All packet drops in the simulation arise exclusively from finite-buffer overflow; wireless channel effects and packet erasure are not modeled, with communication latency captured entirely by the exponential service time $S_{\text{comm}} \sim \text{Exp}(\mu_{\text{comm}})$.

The processor uses exponential vacations with $\mu_{\text{vac}} = 10\,\text{s}^{-1}$, with $P_{\text{active}} = 50\,\text{mW}$ and $P_{\text{sleep}} = 10\,\mu\text{W}$, consistent with ARM Cortex-M7 active core power of 58.5–72.2 μW/MHz at 100 MHz and ultra-low-power sleep mode via the Wake-up Interrupt Controller (WIC), as reported in the ARM Cortex-M7 datasheet [35].

The following metrics are evaluated at each operating point: end-to-end delay (deadline feasibility), energy per job (energy efficiency), server utilization (stability), and packet drop rate and deadline miss rate (QoS degradation under finite buffer constraints), where together are essential to characterize the system's objectives for pre-deployment decisions, especially in resource-constrained wearable systems.

C. MMPP-2 Burstiness Sensitivity

To simulate realistic wearable workloads beyond the Poisson assumption, the discrete-event simulator is extended to support two-state Markov-Modulated Poisson Process (MMPP-2) arrivals [39], characterizing the robustness boundary of the analytical model under correlated bursty traffic. The MMPP-2 model alternates between a slow state (rate λ_0) and a fast state (rate λ_1) governed by a symmetric Markov chain. The process is parameterized via the burstiness coefficient c_b, preserving the mean arrival rate $\bar{\lambda}_{\text{data}}$ exactly, as follows:

$$\lambda_0 = \bar{\lambda}_{\text{data}}(1-d), \quad \lambda_1 = \bar{\lambda}_{\text{data}}(1+d),$$
$$d = \sqrt{\frac{2q(c_b^2-1)}{\bar{\lambda}_{\text{data}}^2}} \tag{32}$$

where $q = \bar{\lambda}_{\text{data}}/r$ with $r = 50$. At $c_b = 1$, both states fire at the same rate and the process reduces to standard Poisson arrivals, recovering the single-task Poisson case. For $c_b > 1$, arrivals become increasingly clustered relative to the Poisson case. The switching rate ratio $r = 50$ places the process in the slow-switching regime, where burst durations are long relative to individual inter-arrival times. Hence, this represents the worst-case scenario for Poisson approximation because temporal correlations persist over extended periods. The burstiness coefficient is swept over $c_b \in \{1.0, 1.5, 2.0, 3.0, 5.0\}$ across the full design space for both tasks, with the Poisson-based analytical model used as the analytical reference throughout,

TABLE III
PROFILED EDGE-AI MODELS AND COMPUTATIONAL PARAMETERS USED FOR M/G/1/K SIMULATION.

Task	Model Name	Input Modality	P_{model}	f_s (Hz)	T_{win} (s)	L	α_{model}	Comp. Cost (C)	T_{stride} (s)	Reported Performance
Fatigue	1D CNN [37]	ECG	161,314	500	20	10,000	1.7814	2.87×10^9	1	0.719 (F1)
	LSTM [37]	ECG	133,250	500	20	10,000	0.8626	1.15×10^9	1	0.751 (F1)
	Bi-LSTM [37]	ECG	4,862	500	20	10,000	0.4476	2.18×10^7	1	0.774 (F1)
	CNN–LSTM Hybrid [37]	ECG	278,114	500	20	10,000	1.2708	3.53×10^9	1	0.742 (F1)
	LSTM–CSA [37]	ECG	100,482	500	20	10,000	0.6574	6.61×10^8	1	0.788 (F1)
	Hierarchical Transformer [37]	ECG	967,330	500	20	10,000	0.00337	3.26×10^7	1	0.937 (F1)
Oxygen Uptake (VO_2)	TCN [38]	HR, HRR, BF, VT	38,529	0.5	200	100	3.9667	1.53×10^7	10	0.81 (R^2)
	Stacked LSTM [38]	HR, HRR, BF, VT	201,345	0.5	200	100	1.3554	2.73×10^7	10	0.82 (R^2)
	TFCN [38]	HR, HRR, BF, VT	194,394	0.5	200	100	3.9428	7.66×10^7	10	0.91 (R^2)
	TFLSTM [38]	HR, HRR, BF, VT	211,546	0.5	200	100	3.3134	7.01×10^7	10	0.92 (R^2)

TABLE IV
EVENT TYPES IN THE DISCRETE-EVENT M/G/1/k SIMULATOR.

Event	Server Action
Data arrival	Enqueue in Q_H (prob. $1-p_t$) or Q_L (prob. p_t); drop if full. Wake server if sleeping.
Control arrival	Enqueue in Q_H; drop if full. Wake server if sleeping.
Departure	Record $D = t_{\text{current}} - t_{\text{arrival}}$. Serve head of Q_H, then Q_L; if both empty, enter sleep and schedule wake-up at $t + \text{Exp}(\mu_{\text{vac}})$.
Wake-up	Serve head of Q_H then Q_L if non-empty; else reschedule wake-up.

so that the error between simulation and analytical prediction directly quantifies the sensitivity of the closed-form expressions with regard to the Poisson assumption.

D. Multi-Task Simulation Extension

Using the validated analytical model, the system is extended to $T = 2$ concurrent tasks to compare centralized ($\theta = 1$) and fully distributed ($\theta = T = 2$) architectures. Fatigue detection and VO_2 estimation are evaluated jointly across the $(\lambda_{\text{data}}, C, F)$ design space, with Theorem 1 characterizing the delay benefit of task isolation and Theorem 2 providing a symmetric-workload analytical reference for energy competitiveness. In the distributed configuration, control traffic is split proportionally by task arrival rate across the two processors. All other simulation parameters are identical to those of the Single-Task Discrete-Event Simulation.

V. RESULTS AND DISCUSSION

A. Analytical–Simulation Evaluation and Baseline Feasibility

Fig. 3 plots normalized analytical estimates against simulation outcomes across all profiled models. An aggregated summary table is also presented in Table V. The analytical model closely tracks simulation results for both end-to-end delay and energy per arrival across the full design space. VO_2 estimation workloads exhibit particularly strong performance, with an average delay prediction error of 3.5%, reflecting their lower arrival rates and more stable service behavior. Fatigue detection workloads exhibit larger variance under high-load configurations, where finite-buffer priority effects amplify queueing delay and packet loss; only 26.5% of configurations remain stable compared to 96.5% for VO_2 estimation (Table V).

In terms of prediction quality, delay achieves strong agreement under low load ($R^2 = 0.991$ for fatigue, 0.998 for VO_2, at $\rho < 0.75$), but degrades sharply near saturation: fatigue reaches $R^2 = 0.678$ at $\rho \geq 0.75$, reflecting the improvement from the corrected load-dependent vacation overhead, while VO_2 reaches $R^2 = -0.186$ at high load — indicating that the analytical model performs worse than a mean baseline for VO_2 estimation at high load. This degradation reflects the sensitivity of queueing-theoretic models to high-variance service times and finite-buffer effects near saturation, where small changes in arrival rate or computational cost produce large, non-linear increases in delay. For unstable configurations ($\rho \geq 1$), analytical formulas correctly return unbounded delays, as queues grow without bound when arrival rates exceed service capacity. Energy prediction, by contrast, remains accurate across all regimes ($R^2 \geq 0.974$ for both tasks), since energy depends primarily on utilization rather than queueing dynamics and remains tractable even when $\rho > 1$.

Fig. 3(d) and (e) show the delay prediction error distributions stratified by load regime for fatigue detection and VO_2 estimation respectively. For fatigue detection, errors are tightly concentrated below 10% at low load ($\rho < 0.75$, $n = 168$, $\mu = 8.7\%$), with the distribution broadening at high load ($n = 23$, $\mu = 32.8\%$) consistent with finite-buffer saturation effects. For VO_2 estimation, errors remain below 10% across nearly all low-load configurations ($n = 456$, $\mu = 3.5\%$). Together, these results indicate that the infinite-buffer approximation has limited impact in the intended low-load operating region, while the increasing error near saturation reflects the growing influence of finite-buffer effects.

Table VI reports the worst-case prediction errors stratified by load regime, including mean, 95th-percentile, and maximum errors. For safety-critical deployment, these results confirm that the analytical model provides reliable delay predictions when the processor is provisioned to operate below $\rho = 0.75$, and that designers should treat high-load predictions as conservative upper bounds rather than precise estimates.

The Shapiro–Wilk normality test (Table VII) across 30 replications confirms that average delay and energy estimates are normally distributed across all profiled models ($p > 0.05$), with the exception of TFLSTM delay under high-load conditions, consistent with heavy-tail effects near queue saturation.

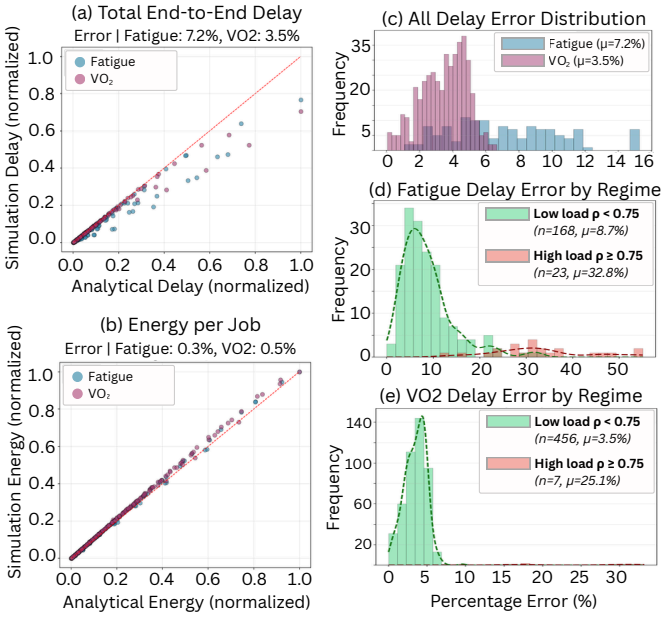


Fig. 3. Comparison of analytical predictions and simulation outputs for delay and energy per job across all model configurations. (a) End-to-end delay; (b) Energy per arrival; (c) Overall distribution of delay prediction errors; (d) Fatigue delay error by regime ($\rho < 0.75$ vs. $\rho \geq 0.75$); (e) VO_2 delay error by regime.

TABLE V
AGGREGATE SIMULATION STATISTICS AND ANALYTICAL VALIDATION QUALITY OVER THE FULL DESIGN SPACE (MEAN ± 95% CI)

Metric	Fatigue	VO_2
Stable Configurations	26.5%	96.5%
Packet Drop Rate	57.3 ± 2.9%	1.0 ± 0.5%
Deadline Miss Rate	67.5 ± 3.1%	0.5 ± 0.4%
Avg Utilization	84.3 ± 2.1%	17.8 ± 2.1%
Avg Delay	20.55 ± 2.87 s	0.31 ± 0.07 s
R^2 ($\rho < 1$)	0.877	0.923
R^2 ($\rho < 0.75$)	0.991	0.998
R^2 ($\rho \geq 0.75$)	0.678	-0.186
Avg Energy	869.2 ± 55.9 mJ	688.9 ± 68.4 mJ
R^2 ($\rho < 1$)	0.999	0.999
R^2 ($\rho < 0.75$)	1.000	1.000
R^2 ($\rho \geq 0.75$)	0.999	0.974

B. Burstiness Sensitivity Analysis

Fig. 4 and Table VIII report prediction errors across $c_b \in \{1.0, 1.5, 2.0, 3.0, 5.0\}$. Delay prediction error remains below 8% for fatigue detection and below 11% for VO_2 estimation across all evaluated c_b values, confirming robustness of the Poisson-based analytical model under correlated bursty arrivals. Fatigue delay error decreases slightly with increasing c_b as burst-induced packet drops remove high-load configurations from the stable region. Energy prediction error remains below 1.2% for fatigue and below 3.5% for VO_2, consistent with the energy model's dependence on mean utilization rather than queueing dynamics. Deadline miss rates remain negligible ($< 0.5\%$) across all c_b values, confirming that the analytical model provides reliable pre-deployment design predictions under realistic bursty wearable workloads.

TABLE VI
WORST-CASE DELAY PREDICTION ERRORS BY LOAD REGIME. LOW LOAD: $\rho < 0.75$; HIGH LOAD: $\rho \geq 0.75$.

Task	Regime	n	Mean	P95	Max
Fatigue	Low ($\rho < 0.75$)	168	8.7%	21.4%	32.3%
Fatigue	High ($\rho \geq 0.75$)	23	32.8%	52.9%	55.0%
VO_2	Low ($\rho < 0.75$)	456	3.5%	5.4%	9.6%
VO_2	High ($\rho \geq 0.75$)	7	25.1%	32.8%	33.2%

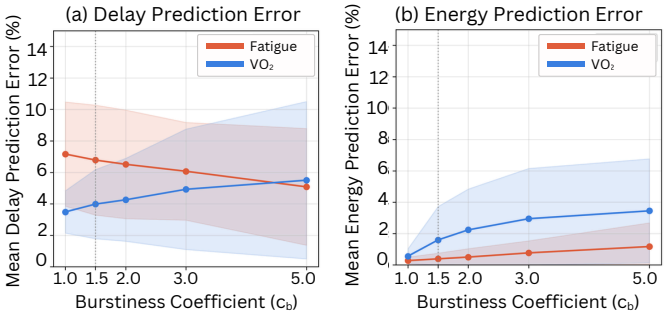


Fig. 4. Mean delay and energy prediction errors as a function of burstiness coefficient c_b under MMPP-2 arrivals ($r = 50$). Shaded regions indicate ± 1 standard deviation. (a) Delay prediction error. (b) Energy prediction error.

C. Delay–Energy Trade-offs and Pareto Front Analysis

Given the validation of the single-task Poisson arrival ($T = 1$, $\theta = 1$, $c_b = 1$) in the previous subsections, this subsection provides its design space results. Fig. 5 presents volumetric 3D visualizations of the complete design space, showing delay and energy as functions of λ_{data}, C, and F. A red surface marks the inference-window feasibility boundary $D_{\text{data}}(\lambda_{\text{data}}, C, F) = T_{\text{stride}}$ (Eq. (20)), separating feasible configurations (below the plane, $D_{\text{data}} \leq T_{\text{stride}}$) from infeasible ones (above the plane, $D_{\text{data}} > T_{\text{stride}}$); the boundary is applied per-task independently at its respective T_{stride}: 1 s for fatigue detection and 10 s for VO_2 estimation, since Fig. 5 presents single-task results ($T = 1$, $\theta = 1$). Fatigue detection occupies a large infeasible region, with deadline violations concentrated at high C or insufficient F. VO_2 estimation lies almost entirely below the deadline surface, confirming that latency is not its limiting constraint. The energy surfaces show complementary behavior where the deadline plane is copied from the delay surfaces to show how much energy was consumed per arrival to meet the task's deadline.

Pareto-optimal configurations are identified by first filtering the full design space to deadline-feasible configurations ($D_{\text{data}} \leq T_{\text{stride}}$), then selecting configurations that achieve the best delay–energy trade-off, where no other feasible configuration simultaneously achieves lower delay and lower energy per job.

Table IX reports results across two processor regimes: the full design space ($0 < F \leq 5$ GFLOPs/s) and the wearable-constrained regime ($0 < F < 175$ MFLOPs/s). The globally Pareto-optimal configurations reveal a sharp contrast between tasks. For fatigue detection, the front collapses to two points — Bi-LSTM at $N_s = 1$ and $N_s = 20$ — reflecting the narrow feasible region imposed by the tight $T_{\text{stride}} = 1$ s real-time constraint; Bi-LSTM is the only architecture light enough

TABLE VII
SHAPIRO–WILK NORMALITY TEST FOR DELAY AND ENERGY DISTRIBUTIONS ACROSS 30 REPLICATIONS.

Task	Model	Delay Mean (s)	Delay Std (s)	p (Delay)	Energy Mean (mJ)	Energy Std (mJ)	p (Energy)
Fatigue	1D CNN	45.8890	0.9343	✓ 0.3247	1112.77	0.83	✓ 0.1953
	LSTM	19.1679	0.1882	✓ 0.9295	1057.72	0.99	✓ 0.4682
	LSTM–CSA	11.5010	0.1033	✓ 0.2049	1000.09	1.02	✓ 0.8003
	CNN–LSTM Hybrid	55.4398	1.5268	✓ 0.9649	1119.11	0.72	✓ 0.3735
	Bi-LSTM	0.4586	0.0009	✓ 0.5800	428.86	0.90	✓ 0.8737
	Hierarchical Transformer	0.7190	0.0023	✓ 0.0943	497.46	0.90	✓ 0.4382
Oxygen Uptake (VO_2)	TCN	0.0785	0.0003	✓ 0.6448	353.72	1.25	✓ 0.9532
	Stacked LSTM	0.1397	0.0008	✓ 0.7963	497.99	1.94	✓ 0.5528
	TFCN	0.5527	0.0052	✓ 0.4240	980.79	4.997	✓ 0.9892
	TFLSTM	0.4876	0.0045	× 0.0003	923.98	4.28	✓ 0.6596

TABLE VIII
ANALYTICAL MODEL PREDICTION ERRORS, DEADLINE MISS RATE, AND UTILIZATION UNDER MMPP-2 BURSTY ARRIVALS (MEAN ± STD).

Metric	Fatigue	VO_2
Delay Prediction Error (%)		
$c_b = 1.0$	7.16 ± 3.31	3.49 ± 1.35
$c_b = 1.5$	6.78 ± 3.49	3.99 ± 2.20
$c_b = 2.0$	6.51 ± 3.45	4.26 ± 2.63
$c_b = 3.0$	6.07 ± 3.10	4.93 ± 3.82
$c_b = 5.0$	5.08 ± 3.70	5.50 ± 5.00
Energy Prediction Error (%)		
$c_b = 1.0$	0.28 ± 0.22	0.56 ± 0.47
$c_b = 1.5$	0.39 ± 0.36	1.60 ± 2.13
$c_b = 2.0$	0.50 ± 0.53	2.23 ± 2.61
$c_b = 3.0$	0.77 ± 0.77	2.94 ± 3.20
$c_b = 5.0$	1.17 ± 1.51	3.44 ± 3.32
Deadline Miss Rate (%)		
$c_b = 1.0$	0.43 ± 2.19	0.00 ± 0.00
$c_b = 1.5$	0.35 ± 1.96	0.00 ± 0.01
$c_b = 2.0$	0.13 ± 0.71	0.00 ± 0.01
$c_b = 3.0$	0.13 ± 0.72	0.00 ± 0.00
$c_b = 5.0$	0.04 ± 0.28	0.00 ± 0.00
Avg Utilization (%)		
$c_b = 1.0$	25.91 ± 13.32	11.54 ± 11.31
$c_b = 1.5$	25.29 ± 12.83	11.07 ± 10.52
$c_b = 2.0$	24.31 ± 12.13	10.32 ± 9.35
$c_b = 3.0$	22.73 ± 11.02	9.65 ± 8.25
$c_b = 5.0$	20.76 ± 9.63	9.03 ± 7.32

TABLE IX
GLOBALLY PARETO-OPTIMAL CONFIGURATIONS FOR FATIGUE DETECTION AND VO_2 ESTIMATION ACROSS THE FULL DESIGN SPACE.

Task	Model	Sensors	Energy (mJ)	Delay (ms)
Full design space ($0 < F \leq 5$ *GFLOPs/s*)				
Fatigue	Bi-LSTM	20	170.13	75.66
Fatigue	Bi-LSTM	1	170.59	33.97
VO_2	TCN	1	169.93	30.81
Wearable-constrained ($0 < F < 175$ *MFLOPs/s*)				
Fatigue	Bi-LSTM	6	762.27	945.11
Fatigue	Bi-LSTM	1	788.04	186.13
VO_2	TCN	1	280.71	53.79

to meet the deadline across the full design space. For VO_2 estimation, the front collapses to a single point — TCN at $N_s = 1$ — confirming that latency is not the binding constraint and energy efficiency governs architecture selection. Under the wearable-constrained regime ($0 < F < 175$ MFLOPs/s), the feasible region contracts further: Fatigue remains dominated by Bi-LSTM at $N_s = 1$ and $N_s = 6$, while VO_2 retains TCN at $N_s = 1$, though energy per job increases substantially due to higher active-state utilization at lower processor speeds. Overall, delay–energy optimality is strongly task-dependent: fatigue detection admits only compact recurrent architectures within the feasible region, while VO_2 estimation offers flexibility governed by processor capacity rather than model selection.

Fig. 6 shows the delay and energy distributions at two representative operating points per task within the wearable-constrained regime ($0 < F < 175$ MFLOPs/s): the Pareto-optimal configuration and the most-expensive stable configuration. Shapiro–Wilk normality tests confirm normally distributed estimates at all four operating points ($p > 0.05$), validating the use of mean ± 95% confidence intervals even at both extreme cases of optimality and instability.

In practice, a designer first filters candidate models by a minimum accuracy threshold — e.g., F1 ≥ 0.9 for fatigue detection or $R^2 \geq 0.9$ for VO_2 estimation — then applies the framework to identify feasible operating points, with the Pareto front representing the accuracy-constrained optimal configurations.

D. Utilization-Dependent Energy Scaling

To explain the Pareto structures summarized in Table IX, we examine how energy per arrival scales with processor utilization as sensor count increases. Fig. 7 shows the percentage change in energy per arrival relative to the single-sensor baseline at $F = 5.0$ GFLOPs/s — the highest processor capacity in the design search space, chosen to span the widest utilization range as sensor count increases from $N_s = 1$ to $N_s = 20$, independently of processor-induced saturation. Unlike static delay–energy projections, this view reveals how workload aggregation and architectural scaling properties interact under realistic compute budgets.

For fatigue detection, most architectures exhibit clear economies of scale: energy per arrival decreases substantially (up to 90%) as sensor count grows, reflecting amortization of fixed inference overhead across larger aggregated workloads. These gains occur well below saturation, indicating efficiency improvements rather than queueing effects. Consequently, fa-

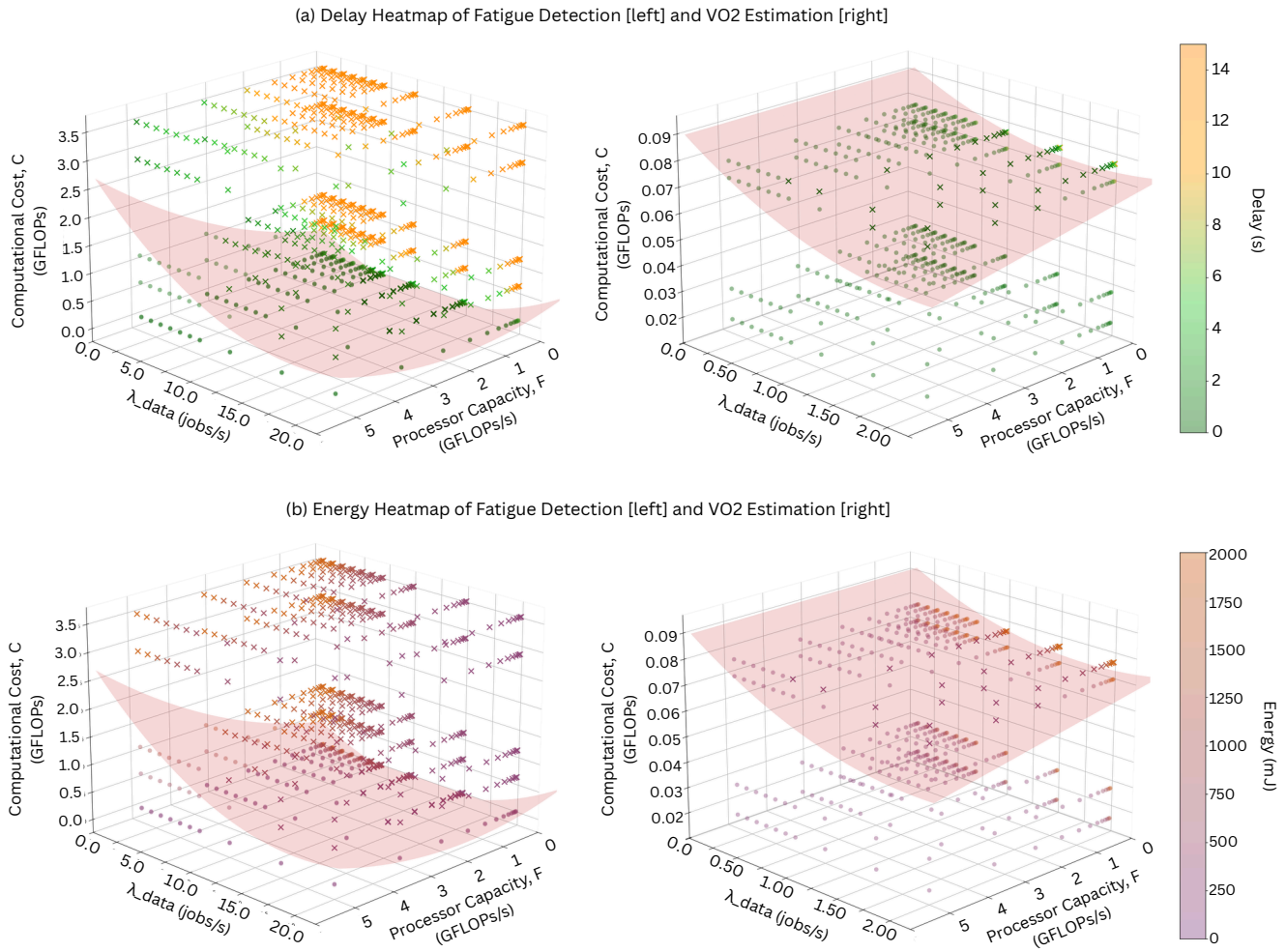


Fig. 5. Delay–energy behavior of all evaluated models across sensing load λ_{data}, computational cost C, and processor capacity F. The red surface marks the per-task inference-window feasibility boundary $D_{\text{data}} = T_{\text{stride}}$ (Fatigue: 1 s, VO_2: 10 s) with feasible configurations ($\circ$) and delay-violating configurations ($\times$). (a) 3D delay surfaces for fatigue detection (left) and VO_2 estimation (right). (b) Corresponding 3D energy surfaces.

tigue sensor selection is governed by accuracy requirements rather than energy constraints.

For VO_2 estimation, architectural divergence dominates. TCN maintains near-flat scaling ($\leq$ 3% variation) across the full sensor range, whereas TFCN and TFLSTM exhibit moderate energy growth (22–25%) and Stacked LSTM shows negligible degradation ($\approx$ 5%) — a direct consequence of attention-based fusion scaling superlinearly with input dimensionality versus the linear scaling of temporal convolutions. This explains why TCN uniquely dominates the VO_2 Pareto front across all sensor counts in Table IX.

TABLE X
ANALYTICAL VS. SIMULATED ENERGY-COMPETITIVENESS FRONTIER AT $\varepsilon = 0.05$.

Threshold	Definition	Analytical	Simulation	Error
ρ^*	Utilization threshold	0.373	0.331	11.4%
η^* (s)	C/F coordinate of frontier	1.841×10^{-1}	1.561×10^{-1}	15.2%
E^* (J)	Energy at frontier	1.122×10^{-2}	1.095×10^{-2}	2.4%

E. Architectural Impact on Multi-Task Inference Performance

The evaluation of centralized execution (one shared processor) versus distributed execution (two independent processors) is presented in this subsection. Fig. 8(a) compares the stability regions of both architectures across model pairs and processor capacities. Under centralized execution, several model pairs exhibit deadline misses or instability when higher-complexity VO_2 models share a processor with delay-sensitive fatigue inference. Long VO_2 service times and high-rate fatigue arrivals interact on the same server, amplifying waiting-time variability and shrinking the feasible design space. Distributed execution eliminates this contention by isolating the two pipelines onto independent processors, making previously infeasible pairings stable and deadline-compliant at moderate processor capacities. The heatmap directly reveals the isolation benefit: distributed execution expands the feasible region most significantly for lighter fatigue detection models (Bi-LSTM, Hierarchical Transformer), whose FLOPs are within the same order of magnitude as the VO_2 models. This load balance

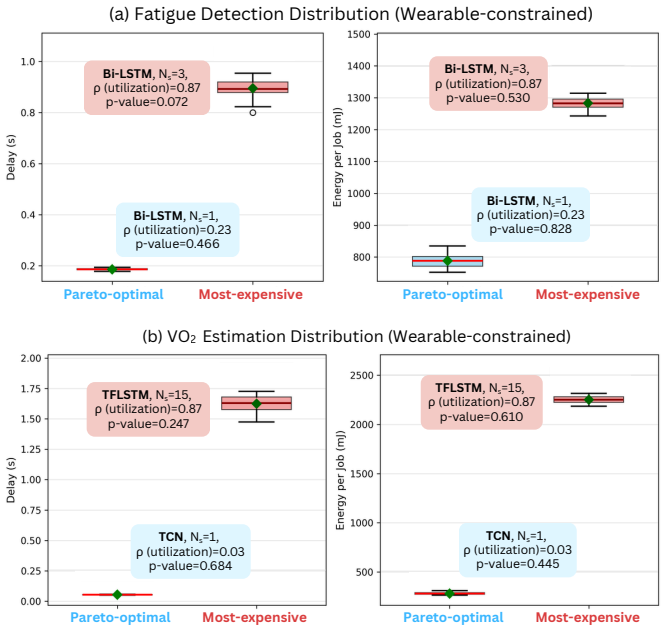


Fig. 6. Delay and energy distributions at the Pareto-optimal and most-expensive stable configurations within the wearable-constrained regime ($0 < F < 175$ MFLOPs/s). (a) Fatigue Detection. (b) VO_2 Estimation.

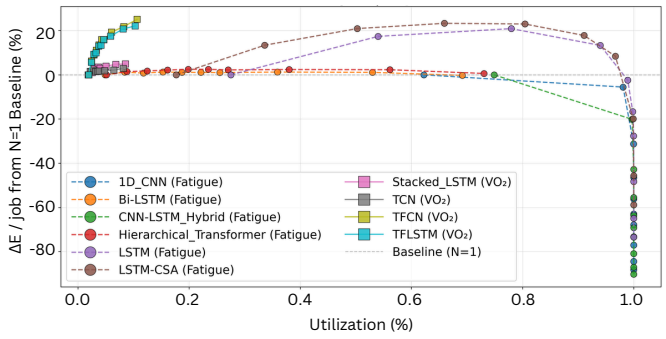


Fig. 7. Energy–utilization scaling at $F = 5.0$ GFLOPs/s as sensor count increases. The x-axis shows processor utilization ρ, while the y-axis reports the percentage change in energy per arrival relative to the single-sensor baseline.

across the two distributed processors enables even the most accurate VO_2 models to remain feasible at moderate processor capacities (50 MFLOPs/s). Fig. 8(b) further illustrates this effect: under centralized execution, many configurations satisfying the fatigue deadline nonetheless exhibit inflated VO_2 delay due to residual service interference. Distributed execution causes VO_2 delay to collapse toward intrinsic service time, enlarging the jointly feasible region.

Table X compares the analytical thresholds from Theorem 2 (derived under symmetric-workload assumptions) against simulation at $\varepsilon = 0.05$. The C/F threshold η^* differs by 15.2%, while ρ^* differs by 11.4%. The corresponding energy at the energy-competitiveness frontier E^* exhibits particularly close agreement, with a deviation of only 2.4% between the analytical and simulated values. The corrected analytical thresholds therefore remain in reasonable agreement with simulation, while preserving the same energy-competitiveness behavior predicted by Theorem 2. The residual error is attributed to the finite-buffer M/G/1/k approximation and the use of median $\bar{\lambda}_{\text{data}}$ in the analytical model, which smooths over the operating-point variation visible in simulation.

Fig. 8(c) illustrates the energy-competitiveness frontier between the two architectures across the full C/F operating range. At low C/F, both architectures operate at low utilization and spend most of their time in the sleep state. However, the distributed architecture ($\theta = 2$) consistently incurs higher $E_{\text{per job}}$ in this regime because two idle nodes each drawing P_{sleep} must amortize their cost over the same total job arrival rate — the idle overhead of $T = 2$ processors is not recovered at low load.

As C/F increases, utilization $\rho(1)$ rises and the centralized per-job energy reference increases. For a fixed aggregate arrival rate, the positive distributed-minus-centralized energy gap remains constant, so relative energy overhead $\Delta E/E_{\text{per job}}(1)$ shrinks monotonically until it falls below the engineering tolerance $\varepsilon = 0.05$ at the analytical threshold $\eta^* = 1.84 \times 10^{-1}$ s, with a corresponding analytical frontier energy of $E^* = 1.12 \times 10^{-2}$ J, marked by the orange star. The simulation frontier (green diamond) occurs at $\eta_{\text{sim}} = 1.56 \times 10^{-1}$ s with $E_{\text{sim}} = 1.09 \times 10^{-2}$ J. These correspond to deviations of 15.2% in the C/F threshold and 2.4% in frontier energy, respectively. Despite the quantitative shift in η^*, the analytical and simulated results preserve the same energy-competitiveness behavior across the operating range.

Beyond the stability limit λ^* ($C/F \geq 5.26 \times 10^{-1}$ s, purple line), the centralized architecture saturates: $\rho(1) \to 1$ and $E_{\text{per job}}(\theta = 1)$ flattens as the queue can no longer keep pace with arrivals. The distributed architecture, by contrast, continues to accommodate increasing computational demand while its processors remain below saturation, with energy per job increasing as their active utilization rises.

The three shaded regimes in Fig. 8(c) provide a direct design guide: (1) operating points to the left of η^* (Designer trade-off) require an application-driven choice where centralized execution is energy-cheaper while distributed execution is faster; (2) between η^* and λ^* (Distributed Energy-Competitive), distributed execution remains faster while its relative energy overhead is within the prescribed tolerance ε; and (3) beyond λ^* (Stability-constrained) the centralized architecture is infeasible regardless of energy preference.

Fig. 9 shows that the relative distributed energy overhead is large at low C/F due to idle-node power, but decreases with utilization until reaching the prescribed tolerance $\varepsilon = 0.05$. As sensing load increases from 0.5 to 3.0 jobs/s, the frontier η^* shifts from 4.960×10^{-1} s to 7.367×10^{-2} s, a 6.7× reduction, making distributed execution energy-competitive across a broader range of model–processor configurations. The corresponding frontier energy E^* decreases from 2.02×10^{-2} J to 5.77×10^{-3} J as baseline power is amortized over more jobs. At the nominal $\bar{\lambda} = 1.3$ jobs/s, simulation follows the analytical gap envelope, with a 15.2% deviation in the frontier coordinate.

By characterizing energy competitiveness analytically, a complementary view quantifies the realized benefit of distributed execution across the full simulated design space. The

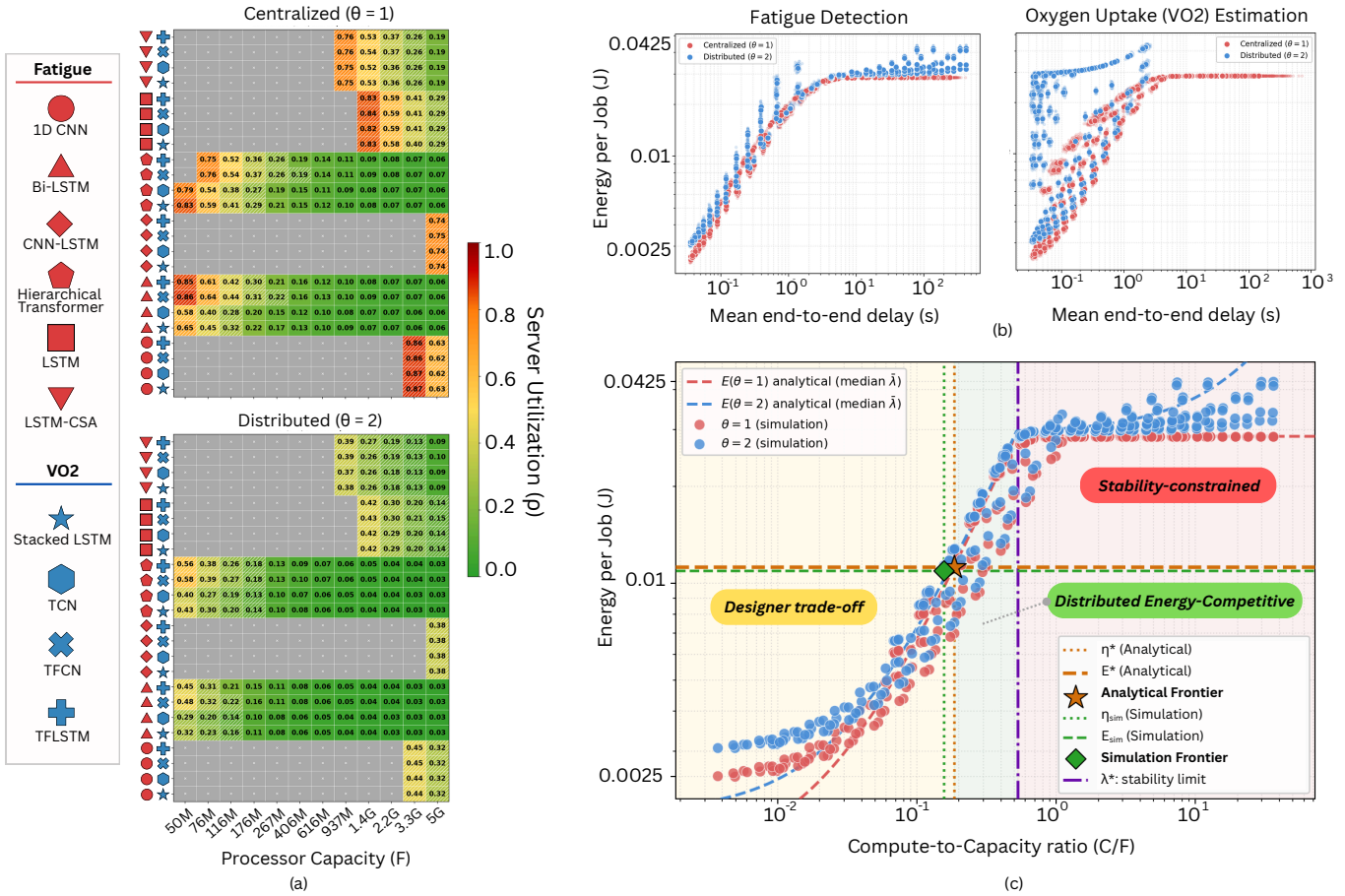


Fig. 8. Centralized vs. distributed multi-task inference performance. (a) Stability map across model pairs and processor capacity F; colors indicate utilization, white dashed lines denote deadline misses, model identities are encoded by marker shape, with red markers denoting fatigue-detection models and blue markers denoting VO_2-estimation models. (b) Delay feasibility for fatigue (left) and oxygen uptake (right). (c) Energy per job $E_{\text{per job}}$ vs. compute-to-capacity ratio C/F for centralized ($\theta = 1$) and distributed ($\theta = 2$) architectures. Dashed curves show analytical predictions; scatter points show simulation results. The orange star marks the analytical frontier point (η^*, E^*) at $\varepsilon = 0.05$, the green diamond marks the simulation frontier point, and the purple dash-dot line marks the stability limit λ^*.

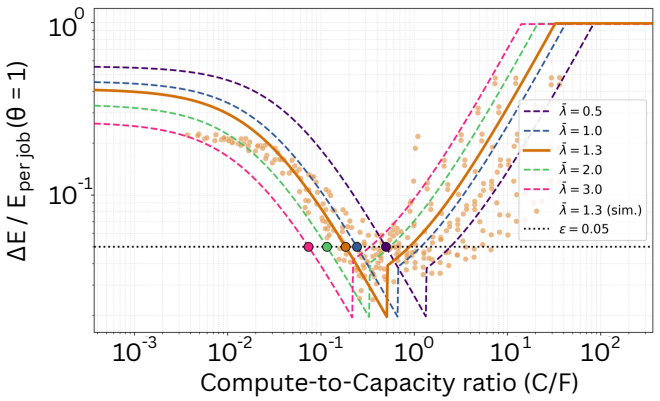


Fig. 9. Relative distributed energy overhead $\Delta E/E_{\text{per job}}(\theta = 1)$ for $\bar{\lambda}_{\text{data}} \in \{0.5, 1.0, 1.3, 2.0, 3.0\}$ jobs/s. Each curve's first intersection with the $\varepsilon = 0.05$ tolerance line (black dotted line) within the centralized stable region defines the frontier coordinate $\eta^*(\bar{\lambda})$, marked by a filled circle. Scatter points show simulation results at the nominal load $\bar{\lambda} = 1.3$. η^* shifts left as load increases.

normalized improvement metric,

$$\Delta(\%) = \frac{M_{\text{cent}} - M_{\text{dist}}}{M_{\text{cent}}} \times 100 \tag{33}$$

TABLE XI
AVERAGE IMPROVEMENT OF DISTRIBUTED EXECUTION RELATIVE TO CENTRALIZED EXECUTION.

Metric	Avg % Impr	Avg Abs Impr	Impr Ratio (D:C)
Fatigue Delay	+5.31%	+0.034 s	113:29
VO_2 Delay	+33.19%	+0.196 s	142:0
Energy	-2.37%	-12.6 mJ	7:135

Avg=Average; Abs=Absolute; Impr=Improvement; D=Distributed; C=Centralized.

quantifies this benefit across configurations, where positive values indicate improvement under the distributed architecture as summarized in Table XI and visualized in Fig. 10.

Across the full design space, distributed execution consistently reduces fatigue delay and substantially improves VO_2 delay, with the average reduction exceeding 30%, reflecting the elimination of cross-task queueing effects. Energy remains largely comparable between architectures, with distributed execution incurring a modest average energy increase of 2.37% across the evaluated configurations. By decoupling delay-sensitive and compute-intensive workloads, distributed architectures improve schedulability and expand the feasible operating space for heterogeneous edge-AI systems.

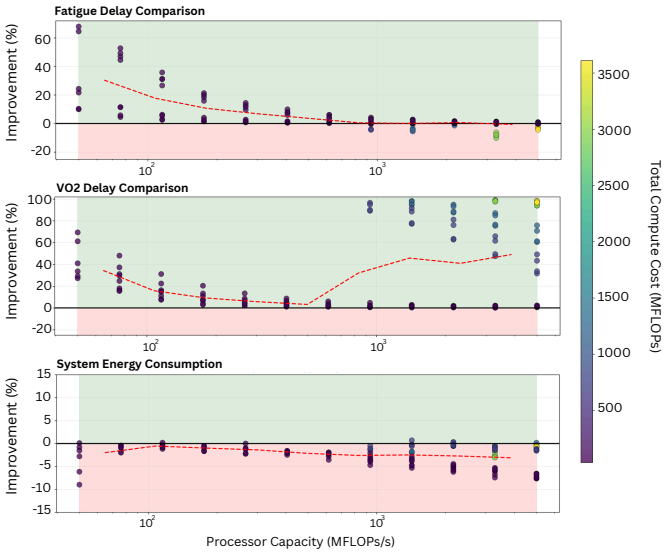


Fig. 10. Relative improvement of distributed execution over centralized execution across processor capacities. Positive values indicate reductions under distributed execution. Points are colored by total compute cost, and the dashed curve highlights the trend across F.

VI. Conclusion

This paper presented a deadline-aware performance model for resource-constrained AI-enabled wearable edge inference, grounded in an M/G/1/k priority queue with service vacations, using an infinite-buffer M/G/1 priority approximation for closed-form analysis, and extended to centralized and distributed processor architectures for concurrent multi-task execution. By linking sensing load (λ), model cost (C), and processor capacity (F), the model yields closed-form predictions for end-to-end delay, stability, and energy consumption under realistic edge-AI workloads with non-preemptive priority scheduling, while finite-buffer effects are retained in the M/G/1/k simulation — before committing to hardware prototyping.

Simulation validation confirms mean delay prediction errors below 10% for well-provisioned configurations ($\rho < 0.75$) and energy predictions accurate across all load regimes ($R^2 \geq 0.974$). Burstiness sensitivity under MMPP-2 arrivals ($c_b \leq 5.0$) further confirms delay error below 8% for fatigue and 11% for VO_2, validating the model under realistic correlated wearable workloads. Pareto analysis reveals strongly task-dependent delay–energy optimality: fatigue detection admits only compact recurrent architectures within the feasible region, while VO_2 estimation admits broad flexibility governed by processor capacity rather than model selection.

For multi-task execution, Theorem 1 formally establishes that per-task delay strictly decreases along the specified progressive-isolation sequence as cross-task queueing interference is removed, while Theorem 2 characterizes the energy-competitiveness frontier via ρ^*, η^*, and E^*. Together, the energy-competitiveness frontier and the centralized stability boundary partition the $(\bar{\lambda}_{\text{data}}, C/F)$ operating plane into three architecture-selection regimes (Corollary 1): a designer trade-off region where application context governs the choice, a distributed energy-competitive region where $\theta = T$ remains faster while its relative energy overhead is within the prescribed tolerance, and a stability-constrained region where distributed execution is the only feasible architecture.

From these analytical and simulation findings, three co-design rules follow: provision F with margin below the deadline boundary, prefer architectures with linear compute scaling under increasing sensor density, and default to distributed execution for heterogeneous concurrent workloads.

The model's predictions degrade near saturation ($\rho \to 1$) and when service times exhibit heavy-tailed variance beyond the deterministic $S_{\text{inf}} = C/F$ assumption. These limitations motivate future extensions of the framework to intermediate hybrid architectures with $1 < \theta < T$, where task grouping across processing nodes becomes an additional design variable requiring assignment-aware optimization beyond the closed-form treatment considered here. Future work will also investigate dynamic task partitioning and runtime architecture-switching policies that leverage ρ^* and η^* as triggers for adapting to time-varying sensing loads and more complex heterogeneous computing architectures.

Appendix A
Formal Proof of Theorem 1

When $\theta = T$, each task is assigned to a dedicated processor. Hence, no other inference tasks are co-located with task τ, and

$$\lambda_{\text{eff}}^{(\tau)}(T,t) = \lambda_{\text{data}}^{(\tau)}(t). \tag{A1}$$

Therefore, Eq. (18) reduces to the single-task delay expression independently for each task.

For strict monotonicity, it suffices to show

$$D^{\text{e2e},(\tau)}(\theta) > D^{\text{e2e},(\tau)}(\theta+1), \qquad \theta \in \{1,\dots,T-1\}. \tag{A2}$$

Under the progressive isolation sequence assumed in Theorem 1, increasing θ by one removes at least one co-located task τ' with $\lambda_{\text{data}}^{(\tau')}(t) > 0$ from the processor serving task τ. This strictly reduces the effective arrival rate seen by task τ:

$$\lambda_{\text{eff}}^{(\tau)}(\theta,t) - \lambda_{\text{eff}}^{(\tau)}(\theta+1,t) \geq \lambda_{\text{data}}^{(\tau')}(t) > 0. \tag{A3}$$

The removal of task τ' induces strictly positive reductions in the residual service term, the high-priority utilization, and the total utilization:

$$\Delta_{\tau'} R = \frac{1}{2}\lambda_{\text{data}}^{(\tau')}(t)\,\mathbb{E}\left[\left(S_{\text{data}}^{(\tau')}\right)^2\right] > 0, \tag{A4}$$

$$\Delta_{\tau'} \rho_{\text{H}} = (1-p_t)\,\lambda_{\text{data}}^{(\tau')}(t)\left(\frac{1}{\mu_{\text{comm}}} + \frac{C^{(\tau')}}{F}\right) > 0, \tag{A5}$$

$$\Delta_{\tau'} \rho = \lambda_{\text{data}}^{(\tau')}(t)\left(\frac{1}{\mu_{\text{comm}}} + \frac{C^{(\tau')}}{F}\right) > 0. \tag{A6}$$

Hence,

$$R(\theta+1) < R(\theta), \tag{A7}$$

$$\rho_{\text{H}}(\theta+1) < \rho_{\text{H}}(\theta), \tag{A8}$$

$$\rho(\theta+1) < \rho(\theta). \tag{A9}$$

Over the stable region $0 \le \rho_H \le \rho < 1$, the waiting-time expressions

$$W_{\mathrm{H}} = \frac{R + R_{\mathrm{V}}(\rho_{\mathrm{H}})}{1 - \rho_{\mathrm{H}}}, \tag{A10}$$

$$W_{\mathrm{L}} = \frac{R + R_{\mathrm{V}}(\rho_{\mathrm{H}})}{(1 - \rho_{\mathrm{H}})(1 - \rho)} \tag{A11}$$

are strictly increasing in R, ρ_{H}, and ρ, where

$$R_{\mathrm{V}}(\rho_{\mathrm{H}}) = \frac{\rho_{\mathrm{H}}}{\mu_{\mathrm{vac}}(1 - \rho_{\mathrm{H}})} \tag{A12}$$

is also strictly increasing in ρ_{H} for $0 \le \rho_{\mathrm{H}} < 1$. Therefore, the strict reductions in R, ρ_{H}, and ρ imply

$$W_{\mathrm{H}}(\theta) > W_{\mathrm{H}}(\theta + 1), \tag{A13}$$

$$W_{\mathrm{L}}(\theta) > W_{\mathrm{L}}(\theta + 1). \tag{A14}$$

The service time of task τ itself is unchanged across architectures, since its communication service rate μ_{comm}, computational cost $C^{(\tau)}$, and processor capacity F are fixed. The relay contribution is also common under the same communication assumptions. Thus, the end-to-end delay strictly decreases:

$$D^{\mathrm{e2e},(\tau)}(\theta) > D^{\mathrm{e2e},(\tau)}(\theta + 1). \tag{A15}$$

Applying this argument for all $\theta \in \{1, \dots, T-1\}$ establishes Eq. (23).

Appendix B
Formal Proof of Theorem 2

Define the relative distributed energy overhead

$$f(\rho(1)) \triangleq \frac{\Delta E(\rho(1))}{E_{\text{per job}}(1, \rho(1))}. \tag{B1}$$

The centralized architecture assigns all T tasks to one processor, whereas the fully distributed architecture assigns each task to a dedicated processor. In the symmetric case, each task contributes $\bar{\lambda}_{\text{data}}/T$ to the aggregate data-arrival rate $\bar{\lambda}_{\text{data}}$ and has computational cost $C^{(\tau)} = \bar{C}$.

For the centralized architecture, the utilization is

$$\rho(1) = \frac{\lambda_{\text{control}} + \bar{\lambda}_{\text{data}}}{\mu_{\text{comm}}} + \bar{\lambda}_{\text{data}} \frac{\bar{C}}{F}. \tag{B2}$$

For the fully distributed architecture, the utilization of node τ is

$$\rho_T^{(\tau)} = \frac{\lambda_{\text{control}}/T + \bar{\lambda}_{\text{data}}/T}{\mu_{\text{comm}}} + \frac{\bar{\lambda}_{\text{data}}}{T} \frac{\bar{C}}{F} = \frac{\rho(1)}{T}. \tag{B3}$$

Thus, each distributed node carries $1/T$ of the centralized utilization and $\sum_{\tau=1}^{T} \rho_T^{(\tau)} = \rho(1)$, while the aggregate control-arrival rate remains λ_{control}.

Let $\bar{N}_s = \frac{1}{T} \sum_{\tau=1}^{T} N_s^{(\tau)}$ denote the average active sensor count across tasks. Under the symmetric setting, $N_s^{(\tau)} = \bar{N}_s$ for all τ. The centralized power term is

$$\Phi_1(\rho(1)) = P_{\text{active}} \rho(1) + P_{\text{sleep}}\big(1 - \rho(1)\big) + \bar{N}_s P_{\text{sensor}}. \tag{B4}$$

When both architectures use the same physical sensors, their intrinsic sensing power is the same. Under the symmetric workload assumptions, the centralized architecture applies the per-sensor power allowance once, while the fully distributed architecture applies it at each of the T processing nodes. The $T - 1$ additional allowances serve as a proxy for the power required to make the same sensor data available to the additional processors. By contrast, $HP_{\mathrm{tx}}/\mu_{\mathrm{comm}}$ accounts for the transmission energy per job across the H communication hops that the job traverses. This term is common to both architectures and cancels from the absolute energy gap in Eq. (27). It still contributes to the centralized energy baseline used to compute the relative overhead and therefore affects the energy-competitiveness frontier.

Similarly, the aggregate distributed power term is

$$\Phi_T^{\text{sum}} = \sum_{\tau=1}^{T} \left[P_{\text{active}} \rho_T^{(\tau)} + P_{\text{sleep}}\big(1 - \rho_T^{(\tau)}\big) + N_s^{(\tau)} P_{\text{sensor}} \right]. \tag{B5}$$

Since the transmission term in Eq. (26) is common to both architectures, the energy gap is

$$\Delta E(\rho(1)) = \frac{\Phi_T^{\text{sum}} - \Phi_1(\rho(1))}{\lambda_{\text{control}} + \bar{\lambda}_{\text{data}}}. \tag{B6}$$

Under the convention $\Delta E = E_{\text{per job}}(T) - E_{\text{per job}}(1)$, $\Delta E > 0$ means that distributed execution is more energy-expensive, whereas $\Delta E < 0$ means that distributed execution is lower energy.

Using $\rho_T^{(\tau)} = \rho(1)/T$ and $N_s^{(\tau)} = \bar{N}_s$, the aggregate distributed power term simplifies exactly to

$$\begin{aligned} \Phi_T^{\text{sum}} = {} & P_{\text{active}} \rho(1) + P_{\text{sleep}}\big(T - \rho(1)\big) \\ & + T \bar{N}_s P_{\text{sensor}}. \end{aligned} \tag{B7}$$

Subtracting the centralized power term gives

$$\begin{aligned} \Phi_T^{\text{sum}} - \Phi_1 &= (T-1) P_{\text{sleep}} \\ &\quad + (T-1) \bar{N}_s P_{\text{sensor}} \\ &= (T-1)\left(P_{\text{sleep}} + \bar{N}_s P_{\text{sensor}}\right). \end{aligned} \tag{B8}$$

For compactness in the remaining derivation, define $\Lambda \triangleq \lambda_{\text{control}} + \bar{\lambda}_{\text{data}}$, $P_0 \triangleq P_{\text{sleep}} + \bar{N}_s P_{\text{sensor}}$, and $\Delta P \triangleq P_{\text{active}} - P_{\text{sleep}}$.

Using these definitions, the exact distributed-minus-centralized energy gap becomes

$$\Delta E = \frac{(T-1) P_0}{\Lambda} > 0. \tag{B9}$$

Hence, the energy ordering does not reverse under the symmetric model.

The centralized energy per job can then be written as

$$E_{\text{per job}}(1, \rho) = \frac{P_0 + \Delta P\, \rho}{\Lambda} + \frac{H P_{\text{tx}}}{\mu_{\text{comm}}}. \tag{B10}$$

Combining Eqs. (B9) and (B10), the relative distributed energy overhead becomes

$$f(\rho) = \frac{(T-1) P_0}{P_0 + \Delta P\, \rho + \Lambda H P_{\text{tx}}/\mu_{\text{comm}}}. \tag{B11}$$

For a fixed aggregate arrival rate Λ and $P_{\text{active}} > P_{\text{sleep}}$, and hence $\Delta P > 0$, differentiation with respect to utilization gives

$$\frac{\partial f(\rho)}{\partial \rho} = -\frac{(T-1)P_0\Delta P}{\left(P_0 + \Delta P\,\rho + \Lambda H P_{\text{tx}}/\mu_{\text{comm}}\right)^2} < 0. \tag{B12}$$

Therefore, for a fixed aggregate arrival rate, although the distributed energy overhead remains positive, its magnitude relative to centralized energy decreases monotonically with utilization.

The energy-competitive operating set is

$$\mathcal{S}_\varepsilon \triangleq \{\rho \in [0,1) : f(\rho) \le \varepsilon\}. \tag{B13}$$

If $\mathcal{S}_\varepsilon$ is nonempty, the utilization threshold is defined as

$$\rho^* = \min \mathcal{S}_\varepsilon. \tag{B14}$$

Since $f(\rho)$ is continuous over the stable operating interval, Eq. (B14) is well-defined whenever $\mathcal{S}_\varepsilon$ is nonempty.

For an interior threshold, $f(\rho^*) = \varepsilon$. Solving Eq. (B11) for ρ^* gives

$$\rho^* = \frac{(T-1)P_0/\varepsilon - P_0 - \Lambda H P_{\text{tx}}/\mu_{\text{comm}}}{\Delta P}. \tag{B15}$$

Hence, $\rho^* = 0$ if $f(0) \le \varepsilon$; no stable threshold exists if $f(1^-) > \varepsilon$; otherwise, Eq. (B15) gives the unique interior threshold.

Substituting $\rho(1) = \rho^*$ into Eq. (26) with $\theta = 1$ gives the centralized energy per job at the energy-competitiveness frontier E^* in Eq. (29). Likewise, substituting $\rho(1) = \rho^*$ into Eq. (21) and solving for $\bar{C}/F$ gives the workload-plane projection η^* in Eq. (30).

The resulting η^* defines the C/F coordinate of the energy-competitiveness frontier separating the Designer trade-off and Distributed Energy-Competitive regions.

Finally, setting $\rho(1) = 1$ in Eq. (21) gives the centralized stability boundary λ^* in Eq. (31). Therefore, ρ^*, η^*, and λ^* jointly partition the $(\bar{\lambda}_{\text{data}}, C/F)$ operating plane into the architecture-selection regimes of Corollary 1.

REFERENCES

[1] L. Yang, O. Amin, and B. Shihada, "Intelligent Wearable Systems: Opportunities and Challenges in Health and Sports," *ACM Computing Surveys*, vol. 56, no. 7, art. 190, Apr. 2024, doi: 10.1145/3648469.

[2] C. Bollineni, M. Sharma, A. Hazra, P. Kumari, S. Manipriya, and A. Tomar, "IoT for Next-Generation Smart Healthcare: A Comprehensive Survey," *IEEE Internet of Things Journal*, vol. 12, no. 16, pp. 32616–32639, Aug. 2025, doi: 10.1109/JIOT.2025.3570188.

[3] T. Zhu, L. Kuang, J. Daniels, P. Herrero, K. Li, and P. Georgiou, "IoMT-Enabled Real-Time Blood Glucose Prediction With Deep Learning and Edge Computing," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 3706–3719, Mar. 2023, doi: 10.1109/JIOT.2022.3143375.

[4] G. Feng, S. Manimurugan, B. Yi, and Y. Feng, "Toward Precision Cardiac Healthcare: Deep Learning and IoT Integration for Real-Time Monitoring and Personalized Diagnosis," *IEEE Internet of Things Journal*, vol. 12, no. 22, pp. 46104–46118, Nov. 2025, doi: 10.1109/JIOT.2025.3579122.

[5] A. Vahedian-Azimi, I. F. Hassan, F. Rahimi-Bashar, H. Elmelliti, M. Salesi, H. Alqahwachi, F. Albazoon, A. Akbar, A. L. Shehata, A. S. Ibrahim, and A. Ait Hssain, "Prognostic effects of cardiopulmonary resuscitation (CPR) start time and the interval between CPR to extracorporeal cardiopulmonary resuscitation (ECPR) on patient outcomes under extracorporeal membrane oxygenation (ECMO): a single-center, retrospective observational study," *BMC Emergency Medicine*, vol. 24, no. 1, Art. no. 36, Mar. 2024, doi: 10.1186/s12873-023-00905-8.

[6] R. D. Hawkins, M. A. Hulse, C. Wilkinson, A. Hodson, and M. Gibson, "The association football medical research programme: an audit of injuries in professional football," *British Journal of Sports Medicine*, vol. 35, no. 1, pp. 43–47, Feb. 2001, doi: 10.1136/bjsm.35.1.43.

[7] T. J. Gabbett, "The training—injury prevention paradox: should athletes be training smarter and harder?" *British Journal of Sports Medicine*, vol. 50, pp. 273–280, Jan. 2016, doi: 10.1136/bjsports-2015-095788.

[8] B. Carrier, M. M. Helm, K. Cruz, B. Barrios, and J. W. Navalta, "Validation of Aerobic Capacity (VO2max) and Lactate Threshold in Wearable Technology for Athletic Populations," *Technologies*, vol. 11, no. 3, Art. no. 71, 2023, doi: 10.3390/technologies11030071.

[9] S. L. Halson, "Monitoring training load to understand fatigue in athletes," *Sports Medicine*, vol. 44, no. Suppl 2, pp. S139–S147, Nov. 2014, doi: 10.1007/s40279-014-0253-z.

[10] A. Amzil, M. Abid, M. Hanini, A. Zaaloul, and S. El Kafhali, "Stochastic analysis of fog computing and machine learning for scalable low-latency healthcare monitoring," *Cluster Computing*, vol. 27, no. 5, pp. 6097–6117, 2024.

[11] L. Tan, K. Yu, A. K. Bashir, *et al.*, "Toward real-time and efficient cardiovascular monitoring for COVID-19 patients by 5G-enabled wearable medical devices: A deep learning approach," *Neural Computing and Applications*, vol. 35, pp. 13921–13934, 2023.

[12] I. Sarkar, M. Adhikari, N. Kumar, and S. Kumar, "Dynamic Task Placement for Deadline-Aware IoT Applications in Federated Fog Networks," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1469–1478, Jan. 15, 2022, doi: 10.1109/JIOT.2021.3088227.

[13] A. Rocha, M. Monteiro, C. Mattos, M. Dias, J. Soares, R. Magalhães, and J. Macedo, "Edge AI for Internet of Medical Things: a literature review," *Computers & Electrical Engineering*, vol. 116, Art. no. 109202, May 2024, doi: 10.1016/j.compeleceng.2024.109202.

[14] I. Behnke, and H. Austad, "Real-Time Performance of Industrial IoT Communication Technologies: A Review," *IEEE Internet of Things Journal*, vol. 11, no. 5, pp. 7399–7410, Mar. 2024, doi: 10.1109/JIOT.2023.3332507.

[15] M. Sharma, A. Tomar, and A. Hazra, "Edge Computing for Industry 5.0: Fundamental, Applications, and Research Challenges," *IEEE Internet of Things Journal*, vol. 11, no. 11, pp. 19070–19093, Jun. 2024, doi: 10.1109/JIOT.2024.3359297.

[16] M. Z. Uddin, "A Wearable Sensor-Based Activity Prediction System to Facilitate Edge Computing in Smart Healthcare System," *Journal of Parallel and Distributed Computing*, vol. 123, pp. 46–53, Jan. 2019, doi: 10.1016/j.jpdc.2018.08.010.

[17] Z. Ning, *et al.*, "Mobile Edge Computing Enabled 5G Health Monitoring for Internet of Medical Things: A Decentralized Game Theoretic Approach," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 2, pp. 463–478, Feb. 2021, doi: 10.1109/JSAC.2020.3020645.

[18] A. I. Awad, M. M. Fouda, M. M. Khashaba, E. R. Mohamed, and K. M. Hosny, "Utilization of Mobile Edge Computing on the Internet of Medical Things: A Survey," *ICT Express*, vol. 9, no. 3, pp. 473–485, Jun. 2023, doi: 10.1016/j.icte.2022.05.006.

[19] M. Hartmann, *et al.*, "Edge computing in smart health care systems: Review, challenges, and research directions," *Transactions on Emerging Telecommunications Technologies*, vol. 33, Art. no. e3710, 2022, doi: 10.1002/ett.3710.

[20] G. Premsankar and B. Ghaddar, "Energy-Efficient Service Placement for Latency-Sensitive Applications in Edge Computing," *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 17926–17937, Sep. 15, 2022, doi: 10.1109/JIOT.2022.3162581.

[21] I. Bisio, C. Garibotto, F. Lavagetto, A. Sciarrone, and M. Zerbino, "Deep at the Edge: AI-Driven Signal Compression for Structural Health Monitoring in Symbiotic IoT Systems," *IEEE Internet of Things Journal*, vol. 12, no. 20, pp. 41538–41549, Oct. 15, 2025, doi: 10.1109/JIOT.2025.3603739.

[22] G. Wang, P. Cheng, Z. Chen, *et al.*, "Inverse reinforcement learning with graph neural networks for full-dimensional task offloading in edge computing," *IEEE Transactions on Mobile Computing*, 2023, doi: 10.1109/TMC.2023.3324332

[23] N. Akar and O. Dogan, "Discrete-Time Queueing Model of Age of Information With Multiple Information Sources," *IEEE Internet of Things Journal*, vol. 8, no. 19, pp. 14531–14542, Oct. 1, 2021, doi: 10.1109/JIOT.2021.3053768.

[24] F. Pouhela, A. Kiggundu, and H. D. Schotten, "Queuing Theory-Based Modeling and Optimization of a Publish/Subscribe IoT Communication System," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2025, pp. 1334–1340, doi: 10.1109/ICC52391.2025.11161937.

[25] M. Emara, H. ElSawy, and G. Bauch, "Prioritized Multistream Traffic in Uplink IoT Networks: Spatially Interacting Vacation Queues," *IEEE*

Internet of Things Journal, vol. 8, no. 3, pp. 1477–1491, Feb. 1, 2021, doi: 10.1109/JIOT.2020.3012515.
[26] E. Pagliari, L. Davoli, and G. Ferrari, "Harnessing Communication Heterogeneity: Architectural Design, Analytical Modeling, and Performance Evaluation of an IoT Multi-Interface Gateway," *IEEE Internet of Things Journal*, vol. 11, no. 5, pp. 8030–8051, Mar. 1, 2024, doi: 10.1109/JIOT.2023.3317672.
[27] D. Fontanelli, L. Greco, and L. Palopoli, "Soft real-time scheduling for embedded control systems," *Automatica*, vol. 49, no. 8, pp. 2330–2338, 2013, doi: 10.1016/j.automatica.2013.04.036.
[28] A. Moin, M. Challenger, A. Badii, and S. Günnemann, "Supporting AI engineering on the IoT edge through model-driven TinyML," in *Proc. IEEE Annual Computers, Software, and Applications Conf. (COMPSAC)*, 2022, pp. 884–893, doi: 10.1109/COMPSAC54236.2022.00140.
[29] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge, U.K.: Cambridge University Press, 2013.
[30] I. Alabdulmohsin, A. Hyadi, L. Afify, and B. Shihada, "End-to-end delay analysis in wireless sensor networks with service vacation," *Proc. IEEE WCNC*, pp. 2799–2804, 2014.
[31] V. Patsias, P. Amanatidis, D. Karampatzakis, T. Lagkas, K. Michalakopoulou, and A. Nikitas, "Task Allocation Methods and Optimization Techniques in Edge Computing: A Systematic Review of the Literature," *Future Internet*, vol. 15, no. 8, Art. no. 254, Jul. 2023, doi: 10.3390/fi15080254.
[32] F. R. B. Cruz, A. R. Duarte, and T. van Woensel, "Buffer Allocation in General Single-Server Queueing Networks," *Computers & Operations Research*, vol. 35, no. 11, pp. 3581–3598, Nov. 2008, doi: 10.1016/j.cor.2007.03.004.
[33] A. Cañete, M. Amor, and L. Fuentes, "Energy-Efficient Deployment of IoT Applications in Edge-Based Infrastructures: A Software Product Line Approach," *IEEE Internet of Things Journal*, vol. 8, no. 22, pp. 16427–16439, Nov. 15, 2021, doi: 10.1109/JIOT.2020.3030197.
[34] O. Krestinskaya, M. E. Fouda, A. Eltawil, and K. N. Salama, "CIMNAS: A joint framework for compute-in-memory-aware neural architecture search," *IEEE Transactions on Circuits and Systems for Artificial Intelligence*, 2025, doi: 10.1109/TCASAI.2025.3617422.
[35] Arm Limited, "Arm Cortex-M7 Processor Datasheet," Arm Limited, 2021. [Online]. Available: https://www.arm.com/-/media/Arm%20Developer%20Community/PDF/Processor%20Datasheets/Arm-Cortex-M7-Processor-Datasheet.pdf [Accessed: Mar. 2026].
[36] Z. Wang, P. Yang, B. Zhang, L. Hu, W. Lv, and C. Lin, "Performance Prediction for Deep Learning Models With Pipeline Inference Strategy," *IEEE Internet of Things Journal*, vol. 11, no. 2, pp. 2964–2978, Jan. 2024, doi: 10.1109/JIOT.2023.3294253.
[37] S. Mu, S. Liao, K. Tao, and Y. Shen, "Intelligent fatigue detection based on hierarchical multi-scale ECG representations and HRV measures," *Biomedical Signal Processing and Control*, vol. 92, Art. no. 106127, 2024, doi: 10.1016/j.bspc.2024.106127.
[38] L. Yang, O. Amin, A. Faisal, and B. Shihada, "Oxygen uptake estimation during cardiopulmonary exercise testing using Temporal Fusion Networks," *ACM Transactions on Computing for Healthcare*, vol. 6, no. 3, art. 39, pp. 1–20, Jul. 2025, doi: 10.1145/3728370.
[39] W. Fischer and K. Meier-Hellstern, "The Markov-Modulated Poisson Process (MMPP) Cookbook," *Performance Evaluation*, vol. 18, no. 2, pp. 149–171, 1993, doi: 10.1016/0166-5316(93)90035-S.



Neil Joshua Limbaga. Neil Joshua Limbaga received his bachelor's degree with honors in Computer Engineering from Ateneo de Manila University, Philippines in 2023, and the master's degree in Electrical and Computer Engineering in 2025 from King Abdullah University of Science and Technology, Saudi Arabia, where he is currently working toward the PhD degree in Electrical and Computer Engineering. His research interests involve deep learning, optimization and computer architecture for sensors, edge and cyber-physical systems.



Osama Amin Dr. Osama Amin received the BSc degree in electrical and electronic engineering from Aswan University, Egypt, in 2000, the MSc degree in electrical and electronic engineering from Assiut University, Egypt, in 2004, and the PhD degree in electrical and computer engineering, University of Waterloo, Canada, in 2010. In 2012, he joined Assiut University as an assistant professor with the Electrical and Electronics Engineering Department. Currently, he is a Senior Research Scientist at King Abdullah University of Science and Technology (KAUST). His research focuses on next-generation communication networks, optical wireless communications, and bio-compatible systems. With a broad background in signal processing and optimization, he aims to improve global connectivity, enhance energy efficiency, and advance healthcare monitoring technologies.



Khaled Nabil Salama Professor Khaled Nabil Salama serves as a professor of Electrical and Computer engineering. He earned his bachelor's degree with honors from Cairo University in 1997 and later completed his master's and doctoral degrees at Stanford University in 2000 and 2005, respectively. His early academic career included a position as an assistant professor at Rensselaer Polytechnic Institute before joining KAUST in 2009, where he was the founding program chair for Electrical Engineering until 2011. He was the director of KAUST sensors initiative between 2016-2021. Professor Salama's research is highly interdisciplinary, focusing on the development of devices, circuits, systems, and algorithms to facilitate inexpensive analytical platforms for industrial, environmental, and biomedical applications. He has made significant contributions to the field of low-power mixed-signal circuits for intelligent sensors and integrated biosensors. His work also extends into VLSI architectures for bio-imaging and instrumentation. More recently, he has been involved in pioneering neuromorphic circuits aimed at emulating brain functions. His scholarly output includes over 400 papers and 48 patents.



Basem Shihada Professor Basem Shihada received the PhD degree in computer science from the University of Waterloo, Canada, in 2007. Prof. Shihada joined King Abdullah University of Science and Technology as a Founding Faculty member in 2008. His expertise lies in developing cutting-edge wireless systems, where he has made groundbreaking contributions across various domains, including intelligent wireless systems, wireless underwater systems, molecular communication systems, and non-terrestrial systems. His notable achievements are the creation and successful demonstration of Aqua-Fi, the world's first underwater Wi-Fi, Sun-Fi, the world first communication via building glass, and communication via breath. His work has been recognized with several best paper awards at renowned conferences within his field. His invaluable contributions have also been published in prestigious scientific journals such as Nature Electronics and many IEEE Transactions. In 2023, he was an area editor. He was the recipient of exemplary editor award from the IEEE Communications Letter journal.