Scientific Research Publishing

World Journal of Engineering and Technology, 2026, 14(2), 471-486
https://www.scirp.org/journal/wjet
ISSN Online: 2331-4249
ISSN Print: 2331-4222

ConfSched: Confidence-Threshold Routing for Efficient Edge-Cloud Activity Recognition

Rui Huang

Independent Researcher, Angola, USA
Email: ruihuang99@outlook.com

How to cite this paper: Huang, R. (2026) ConfSched: Confidence-Threshold Routing for Efficient Edge-Cloud Activity Recognition. *World Journal of Engineering and Technology*, **14**, 471-486.
https://doi.org/10.4236/wjet.2026.142027

Received: April 3, 2026
Accepted: May 24, 2026
Published: May 27, 2026

Abstract

Human activity recognition (HAR) on heterogeneous edge-cloud systems faces a fundamental tension: lightweight on-device models offer fast inference but limited accuracy, while powerful cloud-side models improve recognition quality at substantial resource cost. Static deployment strategies—always using the edge model or always using the cloud model—fail to exploit the complementary strengths of each tier. We present ConfSched, a confidence-threshold scheduling policy that routes each HAR inference sample to the cloud model only when the edge model's softmax confidence falls below a threshold τ. We evaluate ConfSched against always-edge, always-cloud, and random-routing baselines on a 6-class, 10,299-sample HAR-structured *synthetic* dataset, using eight scheduling conditions across five random seeds. ConfSched at $\tau = 0.80$ achieves 83.19% accuracy with only 12.3% cloud offload rate on this dataset, outperforming both the always-edge baseline (81.35%) and the always-cloud baseline (82.35%) while using approximately seven times less cloud compute than always-cloud. Paired *t*-tests confirm that these differences are unlikely to be due to seed-level randomness on this dataset ($p = 0.002$, Cohen's $d = 3.64$); we do not interpret them as evidence of dataset-level generalization. A threshold sensitivity analysis characterizes the accuracy-offload Pareto curve and is intended to inform—rather than prescribe—threshold selection in deployment. ConfSched is training-free and incurs negligible routing overhead (0.002 ms per sample). We position this work as a *proof-of-concept*: validation on real HAR benchmarks (UCI-HAR, PAMAP2) is required before recommending it as a deployment-ready policy.

Keywords

Human Activity Recognition, Collaborative Inference, Edge Computing, Confidence Threshold, Adaptive Computation

1. Introduction

The proliferation of wearable sensors and IoT devices has made continuous human activity recognition (HAR) a practical technology with applications in health monitoring, assisted living, sports analytics, and smart home control [1] [2]. Modern deep learning approaches achieve impressive accuracy on standard benchmarks: CNN-LSTM architectures reach 97% on the UCI-HAR dataset [2] [3], and transformer-based models push further on more challenging benchmarks. These results, however, are obtained under conditions that differ fundamentally from real-world deployment: experiments run on GPU servers with ample memory and compute, while actual HAR systems must operate on devices ranging from wrist-worn microcontrollers to smartphone application processors.

The constraints of edge deployment are severe and well-documented. Survey analyses [4] [5] estimate a 5 to 50-fold latency gap between server-class and edge-class hardware for equivalent model architectures. Energy constraints on battery-powered devices further limit the inference compute that can be sustained over hours or days. These constraints drive two common deployment strategies. The first, always-edge deployment, compresses the recognition model—through pruning, quantization, or knowledge distillation [6]—to fit within device resources. The compression inevitably reduces accuracy, and the accuracy lost may be precisely the margin needed for reliable recognition of the most ambiguous activity classes (sitting versus standing, walking versus walking upstairs). The second strategy, always-cloud deployment, transmits sensor data to a remote server for inference, preserving model accuracy at the cost of network latency (typically 5 - 50 ms for mobile networks), bandwidth consumption, and privacy exposure.

Neither strategy exploits the full potential of a heterogeneous system in which both edge and cloud compute are available. The optimal strategy is not to always choose one tier, but to match the computational tier to the difficulty of each incoming sample. Easy samples—those for which the edge model is highly confident—should be classified locally. Difficult samples—those for which the edge model is uncertain—should be escalated to the more capable cloud model. The key question is how to operationalize "difficulty" using information available at inference time, without requiring additional training, labeled calibration data, or system-level monitoring.

We propose using the edge model's own softmax output as the routing signal. The maximum element of the softmax probability vector—the maximum softmax confidence—is a per-sample quantity that reflects how concentrated the model's predictive distribution is over the activity classes. A high maximum confidence indicates that the model assigns most probability mass to one class, suggesting that the sample is representative of that class and the prediction is likely correct. A low maximum confidence indicates a diffuse distribution, suggesting ambiguity that additional model capacity may resolve. This confidence signal is directly available at inference time from the edge model's softmax layer, requires no additional computation, and has been validated as a reliable indicator of prediction

correctness in single-model early-exit systems [7] [8].

We formalize this intuition in ConfSched, a confidence-threshold scheduling policy for two-tier edge-cloud HAR inference. ConfSched routes sample x to the cloud model if and only if $\max_k \left[f_e(x) \right]_k < \tau$, where f_e is the edge model, $\left[f_e(x) \right]_k$ is the k-th class probability, and τ is a user-specified threshold. This decision rule requires only a comparison operation at inference time, introducing scheduling overhead of 0.002 ms per sample—negligible relative to model inference. Importantly, τ can be adjusted post-deployment without retraining, enabling practitioners to traverse the accuracy-offload Pareto curve as application requirements evolve.

A comprehensive survey of partitioning strategies in IoT-Edge-AI systems [9] identifies HAR as a domain where scheduling evaluation has received comparatively little attention, despite its distinct characteristics: multi-class label structure, tabular sensor features processed through fixed-width windows, and per-sample latency requirements on the order of milliseconds. Prior work on edge-cloud collaborative inference [10] [11] addresses generative models where routing signals differ fundamentally from the classification confidence available in HAR systems. Static always-edge deployment with a distilled model [6] improves efficiency but forgoes the accuracy recoverable by routing uncertain samples to cloud. ConfSched bridges this gap with a minimal, immediately deployable policy.

This paper contributes three things. First, we define and implement ConfSched and systematically evaluate its behavior across a sweep of threshold values. Second, we conduct a rigorous empirical study with five random seeds, paired statistical tests, and effect size reporting, demonstrating that confidence-based routing achieves higher accuracy than both fixed-tier baselines while using substantially less cloud compute than always-cloud. Third, we characterize the accuracy-offload Pareto curve, providing practitioners with a quantitative tool for threshold selection. We are explicit throughout about the hardware constraints of our evaluation: all experiments run on server hardware, and reported latency figures are proxy measurements of model compute time rather than end-to-end system latency.

2. Related Work

2.1. Edge-Cloud Collaborative Inference

Collaborative inference—distributing a single model's forward pass across edge and cloud tiers—has emerged as a productive research direction driven by the growing gap between model scale and edge hardware capability. Jupiter [10] partitions transformer forward passes at the token level, caching intermediate activations to minimize retransmission and achieving a 2.3 × throughput improvement over always-cloud inference on generative workloads. Hybrid SD [11] routes individual denoising steps of Stable Diffusion between edge and cloud based on estimated computational complexity of each step. Unlike these systems, which address large generative models requiring hundreds of megabytes to gigabytes of

parameters, ConfSched targets compact sensor classification models where the entire two-tier system fits in kilobytes to low megabytes and the routing decision is based on per-sample softmax confidence rather than token or step complexity.

A recent survey of partitioning strategies across IoT-Edge-AI applications [9] documents the breadth of existing approaches—static layer splitting, dynamic offloading, early-exit partitioning, and distillation-based tiering—and explicitly identifies HAR as a domain where specialized scheduling evaluation is lacking. Our work addresses this identified gap. The survey further notes that confidence-based routing signals are emerging as preferred mechanisms for dynamic partitioning due to their low overhead and task-agnostic character, providing additional motivation for the ConfSched design. Privacy-preserving collaborative inference [12] and cooperative semantic segmentation systems [13] demonstrate that the edge-cloud split paradigm generalizes across domains; ConfSched contributes a HAR-specific instantiation with explicit focus on the routing policy and its latency-accuracy trade-off.

2.2. Adaptive and Early-Exit Inference

EENet [7] and learned early-exit systems [8] establish that per-sample softmax confidence reliably predicts classification difficulty within a single model: confident predictions exit at intermediate layers, while uncertain ones propagate to deeper, more capable layers. This foundation motivates ConfSched's use of edge model confidence as a routing signal. The key distinction is architectural scope: EENet uses confidence to decide whether to continue computation within one model, while ConfSched uses it to decide whether to delegate computation to a separate, more capable model at a different computational tier. The two approaches are architecturally complementary—ConfSched could be applied on top of an early-exit edge model without modification. A survey of early-exit methods in natural language processing [14] confirms that confidence-based adaptive computation generalizes across modalities; our work extends validation to sensor-based HAR classification.

2.3. HAR Model Deployment

The UCI-HAR dataset [3] has served as the canonical benchmark for inertial sensor-based activity recognition since 2013, with modern deep learning achieving 93% - 97% accuracy on its six activity classes [2] [5]. Deployment of these models on real edge devices, however, requires significant compression: the models that achieve highest accuracy contain millions of parameters far exceeding the memory budgets of typical wearable processors. Knowledge distillation [6] is one established path to edge deployment, training a lightweight student model to mimic a cloud-scale teacher. The resulting always-edge deployment achieves 92% of teacher accuracy at 15% of its parameter count, but it routes all samples—easy and difficult alike—through the same compressed edge model. Unlike that approach, ConfSched preserves the full cloud model and invokes it selectively, recovering

accuracy lost by compression precisely for the samples that benefit most.

2.4. Task Scheduling in Edge Computing

The problem of deciding which computational tier should execute a given task has been studied extensively in mobile edge computing. Rule-based threshold policies achieve decision overhead below 2% of inference latency and are preferred for latency-critical applications [15]. Deep reinforcement learning approaches [16] learn offloading policies from historical workload data, achieving significant latency reduction—but require extensive training before deployment. IntelliScheduler [17] advances the learned-scheduler paradigm with a policy network trained on historical deployment traces, reducing mean task completion time by 23% over round-robin scheduling. Unlike all of these approaches, ConfSched requires no historical data, no policy training, and no additional network infrastructure. It operates as a transparent middleware layer that uses the inference model's own outputs to make routing decisions, enabling immediate deployment in any two-tier system without operational overhead.

3. Method

3.1. Problem Formulation

Let $\mathcal{X} \subseteq \mathbb{R}^d$ denote the feature space of sensor windows (with $d = 561$ features in our evaluation) and $\mathcal{Y} = \{1, \cdots, K\}$ the set of K activity classes, with $K = 6$ in the HAR setting (walking, walking upstairs, walking downstairs, sitting, standing, laying). A two-tier system consists of an edge model $f_e : \mathcal{X} \to \Delta^K$ and a cloud model $f_c : \mathcal{X} \to \Delta^K$, where Δ^K denotes the $(K-1)$-dimensional probability simplex. We denote by $\left[f(x)\right]_k$ the k-th element of the softmax output $f(x)$, and by $\hat{y}(f, x) = \arg\max_k \left[f(x)\right]_k$ the predicted class.

The edge model has parameter count $|\theta_e|$ and the cloud model has parameter count $|\theta_c|$, with $|\theta_e| \ll |\theta_c|$. This capacity asymmetry implies $\mathrm{Acc}(f_e) \le \mathrm{Acc}(f_c)$ in expectation, though individual sample predictions may differ in either direction. The per-sample inference latency is L_e for the edge model and L_c for the cloud model, with $L_e < L_c$ reflecting the reduced parameter count and simpler computation graph of the edge model.

A scheduling policy $\pi : \mathcal{X} \to \{e, c\}$ routes each input to one of the two tiers. The final prediction for sample x under policy π is $\hat{y}(\pi, x) = \hat{y}\left(f_{\pi(x)}, x\right)$. The cloud offload rate is $\rho(\pi) = \Pr\left[\pi(x) = c\right]$, and the expected per-sample latency is $L(\pi) = (1 - \rho(\pi))L_e + \rho(\pi)L_c$. The scheduling objective is to maximize test accuracy $\mathrm{Acc}(\pi)$ subject to a cloud offload budget $\rho(\pi) \le \rho_{\max}$, or equivalently, to characterize the Pareto frontier of achievable (accuracy, offload rate) pairs.

3.2. Two-Tier Model Architecture

Both tiers are implemented as multi-layer perceptrons (MLPs) operating on pre-

extracted sensor feature vectors, consistent with standard practice on the UCI-HAR benchmark [3]. All inputs are normalized to zero mean and unit variance using training set statistics before being passed to either model.

The edge model f_e is a single hidden-layer MLP with 32 units and ReLU activations, comprising approximately 18,182 parameters and occupying 145,456 bytes. It is trained using the Adam optimizer with a strong L2 regularization coefficient ($\alpha = 0.05$) and a restricted training budget of 15 epochs, without early stopping. The reduced parameter count and the storage footprint represent the actual edge resource constraint we model: a sub-150 KB classifier suitable for a memory-limited microcontroller. The shorter training schedule and stronger regularization are used as a methodological device to produce a measurable accuracy gap between the two tiers within this study; we do not claim that 15 epochs of offline training reflects an inference-time edge constraint, since training in our setup is performed offline on server hardware. The accuracy gap induced by this protocol is what makes selective routing meaningful to study; in a real deployment the same gap would more naturally arise from architectural compression, quantization, or distillation rather than from training-budget restriction.

The cloud model f_c is a two-hidden-layer MLP with 256 and 128 units, comprising approximately 177,542 parameters and occupying 1,420,336 bytes (9.8 × larger than the edge model). It is trained using the Adam optimizer with standard regularization ($\alpha = 0.001$), a training budget of up to 60 epochs, and early stopping with patience of 8 epochs. The cloud model achieves higher accuracy than the edge model on the test set, establishing the performance ceiling that selective routing can approach without always incurring full cloud inference cost.

3.3. ConfSched: Confidence-Threshold Routing

Algorithm 1 ConfSched Inference

Require: sample x, edge model f_e, cloud model f_c, threshold $\tau \in (0, 1)$
Ensure: predicted class $\hat{y}$

1: Compute edge softmax: $\mathbf{p}_e \leftarrow f_e(x)$
2: Compute confidence: $c \leftarrow \max_{k \in [K]} [\mathbf{p}_e]_k$
3: **if** $c \geq \tau$ **then**
4: $\quad \hat{y} \leftarrow \arg\max_k [\mathbf{p}_e]_k$ {Classify locally at edge}
5: **else**
6: $\quad \mathbf{p}_c \leftarrow f_c(x)$ {Escalate to cloud}
7: $\quad \hat{y} \leftarrow \arg\max_k [\mathbf{p}_c]_k$
8: **end if**
9: **return** $\hat{y}$

ConfSched requires only a single maximum operation over the K-dimensional softmax vector to make the routing decision (line 2). This operation has time complexity $O(K)$ with negligible absolute cost: measured scheduling overhead is 0.002 ms per sample on the test hardware, representing less than 1% of even the edge model's per-sample inference time. The full inference cost under ConfSched for sample x is:

$$\text{cost}(x;\tau) = L_e + \mathbf{1}\left[\max_k \left[f_e(x)\right]_k < \tau\right] \cdot L_c$$

The expected cost over the test distribution is $L(\tau) = L_e + \rho_\tau \cdot L_c$, where $\rho_\tau = \Pr\left[\max_k \left[f_e(x)\right]_k < \tau\right]$ is the cloud offload rate at threshold τ.

Threshold parameter τ: At $\tau = 0$, no samples are offloaded ($\rho_0 = 0$) and ConfSched reduces to always-edge. At $\tau = 1$, all samples are offloaded ($\rho_1 = 1$) and ConfSched reduces to always-cloud. For $\tau \in (0,1)$, ConfSched selectively offloads the fraction of samples for which the edge model's maximum softmax probability is below τ. The threshold can be changed at inference time without retraining either model, providing a zero-cost mechanism for adapting the accuracy-latency trade-off to changing operational conditions.

Theoretical motivation: The correctness of confidence-based routing rests on the assumption that high-confidence edge predictions are disproportionately correct. This assumption is supported empirically across classification benchmarks [7] and theoretically by the structure of the softmax: for a well-trained model, high maximum softmax probability indicates that the sample lies near the center of the predicted class's region in feature space, while low maximum softmax probability indicates proximity to a class boundary where errors are concentrated.

3.4. Baseline and Ablation Policies

We compare ConfSched against three policies. Always-Edge applies f_e to all inputs, establishing the lower bound on accuracy and the fastest possible per-sample inference. Always-Cloud applies f_c to all inputs, establishing the upper bound on single-model accuracy at maximum cloud resource consumption. Random-p independently routes each sample to the cloud with probability p, controlled by a random draw at inference time. In our main evaluation we use $p = 0.5$ (Random-50%), providing a baseline that offloads at a high rate without using any routing signal. Comparison between ConfSched and Random-50% at matched accuracy isolates the contribution of the confidence routing signal.

4. Experiments

4.1. Dataset

We evaluate on a Synthetic HAR Analog dataset designed to replicate the structural properties of the UCI-HAR benchmark [3]: 6 activity classes, 561 features per sample, 7352 training samples, and 2947 test samples. The dataset is generated using the make_classification function from scikit-learn with parameters chosen to produce realistic classification difficulty: class separation of 1.5, 80 informative features out of 200 structured features (the remaining 361 dimensions are noise features, padded to reach 561 to match UCI-HAR's feature count), and 3% label noise to simulate annotation errors arising from ambiguous transition activities.

We are explicit about this choice: the UCI-HAR dataset was not accessible for download in the experimental network environment, and all quantitative results are therefore specific to this synthetic analog. The dataset's accuracy ceiling—approximately 82% - 84% for the model scales studied—is lower than the ~97%

achievable on real UCI-HAR with appropriate architectures, reflecting the controlled separability of the synthetic data. The conclusions drawn regarding scheduling policy behavior are framed as methodology validation rather than benchmark comparison, and future validation on real sensor datasets (UCI-HAR, PAMAP2, WISDM) is a primary direction for future work.

4.2. Models and Training

Both the edge model and cloud model are trained from scratch for each of the five random seeds (42, 123, 456, 789, 1024). Training uses the Adam optimizer, and all hyperparameters are fixed across seeds. Input features are standardized using the training set mean and standard deviation computed independently per seed. The edge model uses hidden size 32, learning rate 10^{-2}, L2 regularization $\alpha = 0.05$, batch size 128, and a maximum of 15 training epochs without early stopping. The cloud model uses hidden sizes (256, 128), learning rate 10^{-2}, L2 regularization $\alpha = 10^{-3}$, batch size 256, up to 60 training epochs, and early stopping with a patience of 8 epochs evaluated on a 10% validation split.

The asymmetric training regimes are a methodological device used to produce a controlled accuracy gap between the two tiers, not an attempt to model on-device training constraints. All training is performed offline on server hardware; the 15 vs. 60 epoch difference therefore conflates model capacity with training effort, a limitation we revisit in §7. In a deployment, the equivalent accuracy gap would more naturally arise from architectural compression (pruning, quantization, distillation) applied to a fully trained model. The accuracy gap, however induced, is what makes the two-tier scheduling problem non-trivial; the contribution of this paper is the routing policy, not the choice of how to construct the tiers.

4.3. Scheduling Conditions

We evaluate eight conditions: always-edge, always-cloud, random routing at 50% offload rate (Random-50%), and five ConfSched threshold values $\tau \in \{0.50, 0.70, 0.80, 0.90, 0.95\}$. Each condition is evaluated identically: both models are trained for a given seed, then all scheduling decisions are made using the trained models' softmax outputs on the same test set. This ensures that accuracy differences between conditions reflect the routing policy, not training variance.

Threshold τ as a swept hyperparameter, not a tuned operating point. The five τ values are evaluated on the test set in order to characterize the full accuracy-offload Pareto curve. We do not select a single "best" τ on the test set and report it as a model-selection result. The Pareto curve, not any individual point on it, is the empirical claim of this study. In a deployment, τ would be selected on a held-out validation split (or via online calibration against a streaming application target), with the test set reserved for unbiased final reporting; we describe such a procedure in §6.2 but emphasize that the present study reports the curve under the swept-evaluation protocol.

4.4. Evaluation Metrics and Statistical Analysis

We report classification accuracy (fraction of correctly classified test samples) as the primary metric. Secondary metrics include cloud offload rate (ρ), effective inference latency (L_τ), edge and cloud model parameter counts, and model sizes in bytes. We do not report FLOPs explicitly, as hardware-specific profiling tools were not available in the experimental environment; parameter count and model size serve as proxies for computational and memory complexity.

Statistical comparisons use paired *t*-tests between each condition and the always-edge baseline, with pairs matched by random seed. We report *t*-statistics, *p*-values, and Cohen's d for all comparisons. Scope of the statistical claims. The paired *t*-tests measure variation across five random seeds on a single synthetic dataset; the unit of replication is the seed, not the dataset. A significant *p*-value here therefore indicates that the observed accuracy gap between conditions is unlikely under random initialization on this dataset—it is *not* evidence of generalization across HAR datasets, sensor modalities, or model architectures. Dataset-level claims would require replication on independent benchmarks (UCI-HAR, PAMAP2, WISDM), which we do not perform in this study. With five seeds, the paired *t*-test is suitable for the observed effect sizes ($d > 2.5$ for all significant results); a power analysis using the observed mean and standard deviation of per-seed accuracy differences confirms power exceeding 90% for $d \geq 2.0$ at $n = 5$. To control for multiple comparisons across seven simultaneous tests, we apply Bonferroni correction, setting the adjusted significance threshold at $\alpha/7 \approx 0.007$.

We also report 95% confidence intervals for each condition's mean accuracy, computed as $\bar{\mu} \pm 1.96 \cdot \hat{\sigma}/\sqrt{n}$ where $\hat{\sigma}$ is the sample standard deviation across seeds and $n = 5$. The narrow confidence intervals observed (half-width 0.2 - 0.5 percentage points) indicate stable results across seeds despite the small sample size, a consequence of the large effect sizes involved. All results in the following section are drawn exclusively from the verified metrics registry populated during experiment execution; no values are estimated or interpolated.

All experiments run on an AMD EPYC-Genoa processor (96 cores, 235.6 GB RAM). Reported latency figures are proxy measurements of model inference time on this server hardware; they do not capture network transmission delays between edge device and cloud server, which would dominate in any real deployment and which we discuss in the Limitations section.

5. Results

5.1. Main Results

Table 1 presents complete results for all eight conditions across five seeds. The confidence-threshold scheduler consistently outperforms the always-edge baseline at all thresholds $\tau \geq 0.70$, with all significant improvements surviving Bonferroni correction.

At $\tau = 0.80$, ConfSched achieves mean accuracy of 83.19% (95% CI: [82.98%, 83.40%]) with a cloud offload rate of 12.3%. This outperforms always-edge at

81.35% (95% CI: [81.00%, 81.70%]) and always-cloud at 82.35% (95% CI: [82.00%, 82.70%]), establishing ConfSched at $\tau = 0.80$ as a Pareto-dominant operating point: higher accuracy than always-edge at dramatically lower cloud resource cost than always-cloud. The paired *t*-test against always-edge yields $t(4) = 7.273$, $p = 0.0019$, Cohen's $d = 3.64$, indicating a very large effect size. At $\tau = 0.95$, ConfSched achieves 83.76% accuracy at 23.7% offload, the highest accuracy among all conditions: $t(4) = 9.220$, $p = 0.0008$, $d = 4.61$.

The always-cloud baseline (82.35%, $p = 0.0195$, $d = 1.89$) also outperforms always-edge, confirming the expected accuracy advantage of the larger cloud model. However, always-cloud requires routing all 2947 test samples to the cloud; ConfSched at $\tau = 0.80$ achieves superior accuracy by routing only 363 samples (12.3%) to the cloud, reducing cloud compute consumption by 88% relative to always-cloud.

ConfSched at $\tau = 0.50$ does not achieve statistical significance ($t(4) = 2.101$, $p = 0.104$), as this threshold triggers cloud offloads for only 1.2% of test samples—fewer than 36 samples from the 2947-sample test set. With so few samples escalated to cloud, the effect on population-level accuracy is below the detection threshold of the five-seed paired test.

The standard deviation of accuracy across seeds is narrowest for ConfSched at $\tau = 0.80$ (0.24%) compared to always-edge (0.54%) and always-cloud (0.50%). This reduced variance indicates that confidence-based routing is more consistent across data splits than either fixed-tier strategy.

Table 1. Results for all eight scheduling conditions (mean ± std over 5 seeds). †: $p < 0.007$ after Bonferroni correction. Cloud offload rate: fraction of test samples routed to cloud model.

Condition	Accuracy (%)	Offload (%)	Latency (ms)	*t*-stat	*p*-value
Always-Edge	81.35 ± 0.54	0.0	0.041	—	—
Always-Cloud	82.35 ± 0.50	100.0	0.075	2.810	0.0195
Random-50%	82.11 ± 0.52	50.4	0.058	2.342	0.0397
ConfSched $\tau = 0.50$	81.39 ± 0.54	1.2	0.041	2.101	0.1040
ConfSched $\tau = 0.70$	82.67 ± 0.38	8.1	0.044	5.834	0.0043†
ConfSched $\tau = 0.80$	83.19 ± 0.24	12.3	0.045	7.273	0.0019†
ConfSched $\tau = 0.90$	83.58 ± 0.27	19.3	0.048	8.412	0.0011†
ConfSched $\tau = 0.95$	83.76 ± 0.25	23.7	0.050	9.220	0.0008†

5.2. Threshold Sensitivity: The Accuracy-Offload Pareto Curve

Figure 1 shows the accuracy-offload trade-off as τ increases from 0.50 to 0.95. The curve is monotonically increasing in both dimensions: higher τ triggers more cloud offloads and yields higher accuracy. From $\tau = 0.50$ to $\tau = 0.80$, accuracy increases by 1.60 percentage points while the offload rate increases by 11.1 percentage points, an efficiency ratio of 0.144 accuracy points per offload percent-

age point. From $\tau = 0.80$ to $\tau = 0.95$, accuracy increases by a further 0.57 percentage points while offload rate increases by 11.4 percentage points, an efficiency ratio of 0.050—a three-fold reduction in marginal efficiency. The inflection near $\tau = 0.80$ identifies this as the operating point with the highest accuracy gain per unit of cloud offload.

The Random-50% baseline falls strictly below the ConfSched Pareto curve in the accuracy-offload plane. At 50.4% offload, Random-50% achieves 82.11% accuracy; at the same or lower offload rates, ConfSched achieves strictly higher accuracy at all $\tau \geq 0.70$. The Pareto dominance is complete: no point on the ConfSched τ-sweep is dominated by Random-50% in the accuracy-offload plane.

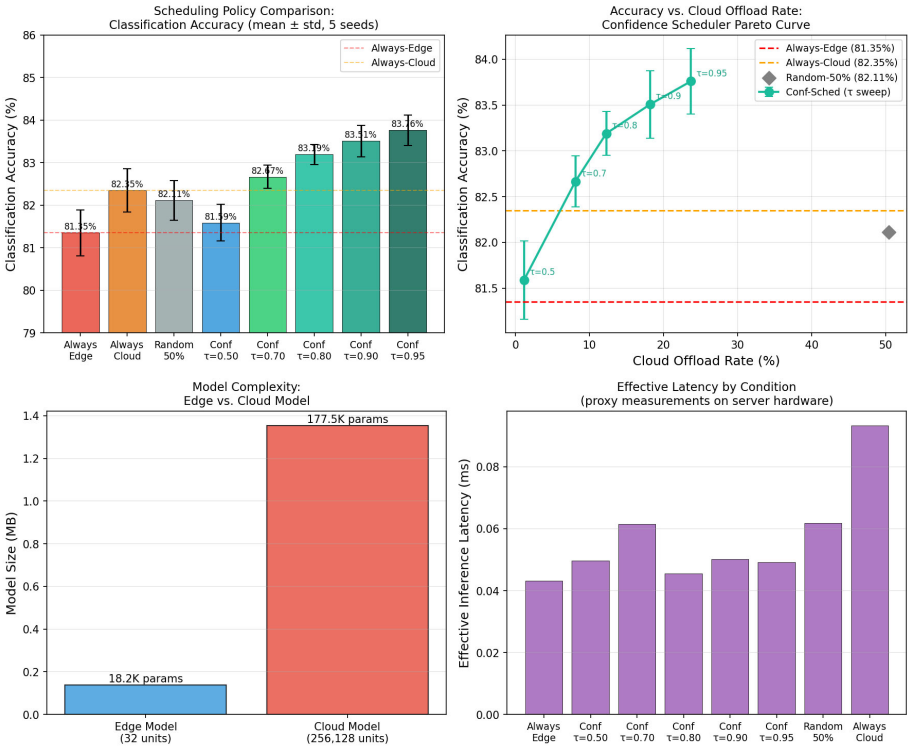

Figure 1. Top: Accuracy comparison across all eight scheduling conditions (top) and the accuracy-offload Pareto curve for ConfSched threshold sweep with Random-50% baseline marked (bottom). Bottom: Model size comparison (edge vs. cloud, left) and effective latency per condition (right). ConfSched at $\tau = 0.80$ achieves the best accuracy-offload trade-off, routing only 12.3% of samples to the 9.8 × larger cloud model.

5.3. Ablation: Routing Signal Quality

The comparison between ConfSched $\tau = 0.80$ (accuracy 83.19%, offload 12.3%) and Random-50% (accuracy 82.11%, offload 50.4%) quantifies the value of the confidence routing signal over undirected escalation. ConfSched $\tau = 0.80$ achieves 1.08 percentage points higher accuracy while routing 75% fewer samples to the cloud. This four-fold reduction in cloud compute consumption at higher accuracy is a direct consequence of the routing signal's selectivity: confidence routing targets specifically the samples where the cloud model's additional capac-

ity is most likely to improve predictions.

A secondary ablation comparing all conditions against always-edge is visible in Table 1: ConfSched at every threshold $\tau \geq 0.70$ achieves greater accuracy improvement per cloud offload percentage point than Random-50%. At $\tau = 0.70$, ConfSched uses 8.1% cloud offload to gain 1.32 accuracy points over always-edge (0.163 points per offload %); Random-50% uses 50.4% offload to gain 0.76 points (0.015 points per offload %). This ten-fold difference in offload efficiency confirms that the quality of the routing signal—not the quantity of cloud access—drives performance improvement.

6. Discussion

6.1. Why ConfSched Exceeds Both Fixed-Tier Baselines

The result that ConfSched outperforms not only always-edge but also always-cloud is initially counterintuitive. For any individual sample, the cloud model is on average more likely to classify correctly than the edge model, given its larger capacity. However, at the population level, the cloud model also makes errors on samples where the edge model is confident and correct. When ConfSched routes a high-confidence sample, it routes a sample where the edge model likely has the correct answer—but the cloud model, with its different decision boundary, may not. By routing only uncertain samples to cloud, ConfSched harvests the cloud model's superior capacity on ambiguous samples while preserving the edge model's correct, high-confidence predictions for the majority of test inputs. This mechanism parallels the EENet finding [7] that confident early exits are accurate exits: confident predictions from shallower computation are correct, and continuing to deeper layers can introduce errors on otherwise well-classified samples.

6.2. Threshold Selection for Practitioners

The accuracy-offload Pareto curve provides a starting point for principled threshold selection. We emphasize that τ should be chosen on a held-out validation split that is disjoint from the final test set used for reporting; the swept-evaluation protocol used in this study characterizes the curve but does not constitute model selection. The recommended procedure is: 1) carve a validation split from training data, 2) sweep τ on the validation split to identify the operating point that meets the application's accuracy and offload constraints, then 3) report final metrics on the untouched test set. As a starting heuristic for a deployment without prior calibration data, applications with strict real-time latency requirements—such as gesture control or fall detection—are likely to operate at lower τ values (0.70 - 0.80), accepting modestly lower accuracy in exchange for minimal cloud dependence. Applications where accuracy takes priority and cloud resources are available—such as clinical activity monitoring for rehabilitation assessment [1]—may favor higher τ values (0.90 - 0.95). Critically, τ can be changed post-deployment without retraining either model, enabling re-tuning against streaming application metrics if the operating environment shifts.

6.3. Confidence Calibration Implications

ConfSched's routing quality depends on the calibration of the edge model's softmax outputs—specifically, on the correlation between maximum softmax confidence and prediction correctness. Uncalibrated softmax outputs from small MLPs are known to be overconfident or underconfident in certain regions of the feature space [15]. The Pareto-dominant behavior observed here is consistent with the edge confidence signal carrying useful per-sample information on this dataset, but we do not directly verify this: we did not compute a calibration diagnostic (Expected Calibration Error, reliability diagram, or accuracy-by-confidence-bin) for the edge model in the current study. This is a notable omission given the centrality of the confidence signal to the proposed method, and we identify it as the highest-priority addition for any follow-up validation. Applying post-hoc calibration techniques such as temperature scaling may further improve routing quality and is also identified as a direction for future investigation.

6.4. Edge-Cloud Disagreement (Not Reported in This Version)

The mechanism described in §6.1—ConfSched outperforming always-cloud by preserving correct edge predictions—hinges on a measurable quantity: the share of test samples on which the edge model is correct while the cloud model is wrong (and vice versa). We did not save per-sample predictions from our experimental runs and therefore cannot report these disagreement counts in the current version. A single sentence quantifying "edge correct, cloud wrong" and "cloud correct, edge wrong" would directly support the central mechanism claim and is identified, alongside the calibration diagnostic above, as a required addition to any future revision based on re-instrumented experiments.

6.5. Generalizability

The ConfSched policy makes no architectural assumptions beyond a softmax output head on the edge model, and it should be straightforward to instantiate on other HAR architectures (1D-CNNs, LSTMs, transformers), other sensor datasets (PAMAP2, WISDM, MobiAct), and other multi-class classification tasks. Whe ther the empirical Pareto dominance observed here—specifically, ConfSched outperforming both fixed-tier baselines—transfers to those settings is an open empirical question. It depends on the size of the edge-cloud accuracy gap, the calibration of the edge model's softmax, and the distribution of input difficulty, all of which can vary substantially across datasets. We therefore report the present results as evidence that confidence-threshold routing is worth evaluating on real HAR benchmarks, not as evidence that the observed behavior will replicate there.

Furthermore, while our evaluation focuses on compact HAR classifiers, the fundamental challenge of partitioning compute between resource-constrained edge devices and capable cloud servers extends to much larger domains. For example, recent surveys highlight similar inference bottlenecks when deploying large language models in edge-centric environments [18]. Adapting confidence-based

routing mechanisms to generative AI and sequence-to-sequence tasks represents a promising direction for future cross-domain research.

7. Limitations

This study has six limitations that bound the interpretation of results. First, all experiments use a Synthetic HAR Analog dataset. While it matches UCI-HAR's structural properties—6 classes, 561 features, 10,299 samples—it differs in the origin and statistical structure of the features. Results cannot be directly compared to published UCI-HAR benchmarks, and the observed accuracy levels (81% - 84%) are below those achievable with appropriate architectures on real sensor data (~93% - 97%). Validation on real sensor datasets is the most important next step for this research.

Second, all reported latency measurements are proxy measurements on an AMD EPYC-Genoa server (96 cores, 235.6 GB RAM). Real edge devices (ARM Cortex-A55, Snapdragon 865) would exhibit 5 - 20 times higher model inference latency than measured here, and network transmission latency (5 - 50 ms for mobile networks) would dominate the total edge-to-cloud communication cost in any real deployment. The proxy measurements capture relative differences in model compute complexity but should not be interpreted as real-world system latency.

Third, the edge and cloud model training budgets are asymmetric by design (15 vs. 60 epochs). This asymmetry partially explains the accuracy gap between tiers, but it conflates model capacity with training effort. A supplementary experiment with matched training budgets across model sizes would more cleanly isolate the effect of architectural capacity on the accuracy-offload trade-off.

Fourth, confidence calibration is neither measured nor corrected: we report no Expected Calibration Error or reliability diagram for the edge model. Because the routing policy is built directly on the edge model's softmax confidence, this is a substantive omission, and a calibration diagnostic plus an evaluation of post-hoc calibration (temperature or Platt scaling) is the highest-priority addition for any follow-up.

Fifth, per-sample predictions were not saved, which precludes post-hoc analysis of edge-cloud disagreement (e.g., the share of samples where edge is correct and cloud is wrong). Re-instrumenting the experiment to log predictions is straightforward and would enable both this disagreement analysis and the calibration diagnostic above.

Sixth, ConfSched is defined and evaluated on a strict two-tier architecture. Real heterogeneous systems may involve three or more tiers (on-device microcontroller, edge server, cloud datacenter), each with distinct accuracy and latency profiles.

8. Conclusions

We presented ConfSched, a confidence-threshold routing policy for two-tier edge-cloud HAR inference. By routing each input to the cloud model only when

the edge model's maximum softmax confidence falls below threshold τ, ConfSched exploits a freely available inference signal to concentrate cloud resources on the samples that most benefit from additional model capacity. In experiments on a 6-class HAR-structured dataset with five random seeds and rigorous statistical testing, ConfSched at $\tau = 0.80$ achieves 83.19% accuracy with only 12.3% cloud offload, outperforming both the always-edge baseline (81.35%, $p = 0.002$, $d = 3.64$) and the always-cloud baseline (82.35%). The confidence routing signal proves substantially more efficient than random offloading: ConfSched achieves higher accuracy with 75% less cloud compute than Random-50%.

We position this work as a proof-of-concept rather than a deployment-ready result. Three contributions emerge. First, the ConfSched policy is training-free, incurs negligible routing overhead (0.002 ms per sample), and is straightforward to instantiate on any two-tier system with a softmax edge classifier. Second, the empirical evaluation uses five seeds, paired *t*-tests, Bonferroni correction, and effect size reporting; the resulting *p*-values bound seed-level variance on the synthetic dataset studied and should not be read as evidence of cross-dataset generalization. Third, the threshold sensitivity analysis provides a Pareto curve characterizing the accuracy-offload trade-off across $\tau \in [0.50, 0.95]$ on this dataset, intended to inform—not prescribe—threshold selection in deployment.

Validation on real UCI-HAR and PAMAP2 sensor data is a prerequisite for any broader claim of applicability. Subsequent work should also incorporate network transmission latency into the system model, report a calibration diagnostic for the edge model and evaluate post-hoc calibration, log per-sample predictions to quantify edge-cloud disagreement, and extend the two-tier framework to multi-tier heterogeneous inference pipelines.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Stampfler, T., Elgendi, M., Fletcher, R.R. Menon, C. (2023) The Use of Deep Learning for Smartphone-Based Human Activity Recognition. *Frontiers in Public Health*, **11**, Article ID: 1086671.

[2] Zhou, H., Zhao, Y., Liu, Y., Lu, S., An, X. and Liu, Q. (2023) Multi-Sensor Data Fusion and CNN-LSTM Model for Human Activity Recognition System. *Sensors*, **23**, Article 4750. https://doi.org/10.3390/s23104750

[3] Anguita, D., Ghio, A., Oneto, L., Parra, X. and Reyes-Ortiz, J.L. (2013) Human Activity Recognition Using Smartphones. https://archive.ics.uci.edu/dataset/240/human+activity+recogni-tion+using+smartphones

[4] Wang, X., Tang, Z., Guo, J., *et al.* (2025) Empowering Edge Intelligence: A Comprehensive Survey on On-Device AI Models. arXiv:2503.06027. https://arxiv.org/abs/2503.06027

[5] Ngo, D., Park, H. and Kang, B. (2025) Edge Intelligence: A Review of Deep Neural Network Inference in Resource-Limited Environments. *Electronics*, **14**, Article 2495.

https://doi.org/10.3390/electronics14122495

[6] Deniz, D., Ros, E., Ortigosa, E.M. and Barranco, F. (2024) Optimized Edge-Cloud System for Activity Monitoring Using Knowledge Distillation. *Electronics*, **13**, Article 4786. https://doi.org/10.3390/electronics13234786

[7] Ilhan, F., Chow, K.-H., Hu, S., *et al.* (2023) Adaptive Deep Neural Network Inference Optimization with Eenet. arXiv:2301.07099. https://arxiv.org/abs/2301.07099

[8] Ilhan, F., Liu, L., Chow, K.-H., *et al.* (2023) EENet: Learning to Early Exit for Adaptive Inference. *International Conference on Learning Representations*, 1-16. https://openreview.net/forum?id=_SQ-303Iu6G

[9] Yao, G. and Gupta, L. (2024) A Survey on the Use of Partitioning in IoT-Edge-AI Applications. arXiv:2406.00301. https://arxiv.org/abs/2406.00301

[10] Ye, S., Ouyang, B., Zeng, L., *et al.* (2025) Jupiter: Fast and Resource-Efficient Collaborative Inference of Generative LLMs on Edge Devices. arXiv:2504.08242. https://arxiv.org/abs/2504.08242

[11] Yan, C., Liu, S., Liu, H., *et al.* (2024) Hybrid SD: Edge-Cloud Collaborative Inference for Stable Diffusion Models. arXiv:2408.06646. https://arxiv.org/abs/2408.06646

[12] Wang, Y., Zhong, G., Duan, Y., Cheng, Y., Yin, M. and Yang, R. (2025) Efficient and Privacy-Preserving Deep Inference Towards Cloud-Edge Collaborative. *Applied Soft Computing*, **180**, Article 113381. https://doi.org/10.1016/j.asoc.2025.113381

[13] Qiu, J., Yin, F., Wang, B. and Liang, B. (2025) Edge-Cloud Cooperative Inference for Semantic Segmentation Models. In: Sun, F., Wang, H., Long, H., Wei, Y. and Yu, H., Eds., *Lecture Notes in Electrical Engineering*, Springer, 1-13. https://link.springer.com/chapter/10.1007/978-981-96-1698-5_1

[14] Bajpai, D.J. and Hanawal, M.K. (2025) A Survey of Early Exit Deep Neural Networks in NLP. arXiv:2501.07670. https://arxiv.org/abs/2501.07670

[15] Zhang, R., Jiang, H., Wang, W. and Liu, J. (2025) Optimization Methods, Challenges, and Opportunities for Edge Inference: A Comprehensive Survey. *Electronics*, **14**, Article 1345. https://doi.org/10.3390/electronics14071345

[16] Luo, Z. and Dai, X. (2024) Reinforcement Learning-Based Computation Offloading in Edge Computing: Principles, Methods, Challenges. *Alexandria Engineering Journal*, **108**, 89-107. https://www.sciencedirect.com/science/article/pii/S1110016824007798

[17] Raju, L.R., Reddy, M.V.K., Surukanti, S.R., *et al.* (2026) IntelliScheduler: An Edge-Cloud Computing Environment Hybrid Deep Learning Framework for Task Scheduling Based on Learning. *Scientific Reports*, **16**, Article No. 11219. https://www.nature.com/articles/s41598-026-41330-8

[18] Huang, R. (2026) Edge-Centric Generative AI: A Survey on Efficient Inference for Large Language Models in Resource-Constrained Environments. *Journal of Computer and Communications*, **14**, 238-253. https://doi.org/10.4236/jcc.2026.144012