

A Hierarchical Framework for Fair and Efficient Distributed Inference at the Edge

Michele Simeone^{a,b}, Ghina Al-Atat^{a,b}, Vincenzo Mancuso^{a,c}, Marco Ajmone Marsan^a

^a IMDEA Networks, Madrid, Spain; ^b Universidad Carlos III de Madrid, Spain; ^c Università degli Studi di Palermo, Palermo, Italy.

INTRODUCTION

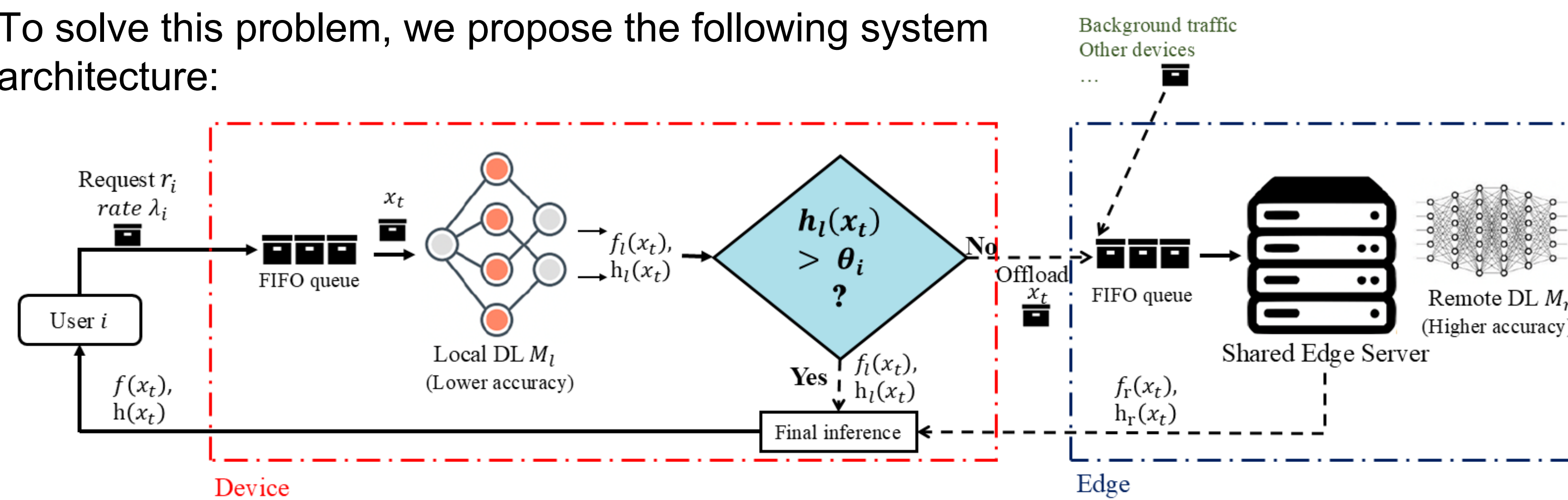
To maximize AI potential in distributed environments, managing shared resources is vital for ensuring equitable user access. This poster presents the **Hierarchical Inference Framework (HIF)**, a decentralized mechanism that optimizes collective performance across edge devices without explicit inter-communication.

Key Challenge: Preventing network saturation and the dominance of high-power nodes to ensure **fairness across heterogeneous devices**

SYSTEM ARCHITECTURE & EXPERIMENT

User's Loop

To solve this problem, we propose the following system architecture:



Experimental Setup

All experiments were conducted using a network of Raspberry Pi devices connected to a server through a shared Wi-Fi network, which was also used by other devices to introduce realistic interference and contention.

Each RaspberryPi received on average 7 image queries per second. This input rate was selected so that the remote server would experience congestion if all devices attempted to offload their entire workload simultaneously.

Threshold Update's Algorithm

```
def update_theta(confs, theta, prev_score, alpha, max_delta, s_prev1, s_prev2):

    # Compute raw score
    score = sum(confs)

    # Initialization
    if prev_score is None:
        s_prev2 = theta
        return score, None, s_prev2, None
    if s_prev1 is None:
        score = (1 - alpha) * prev_score + alpha * score
        prev_score = score
        s_prev1 = theta
        return score, s_prev1, None, None

    # Smoothing score values
    score = (1 - alpha) * prev_score + alpha * score

    # Numerical gradient
    denom = s_prev1 - s_prev2
    if abs(denom) < 1e-6:
        denom = np.copysign(1e-6, denom if denom != 0 else 1.0)
    grad = (score - prev_score) / denom

    # Compression using arctan
    compr_grad = (2 * np.arctan(grad) / np.pi)

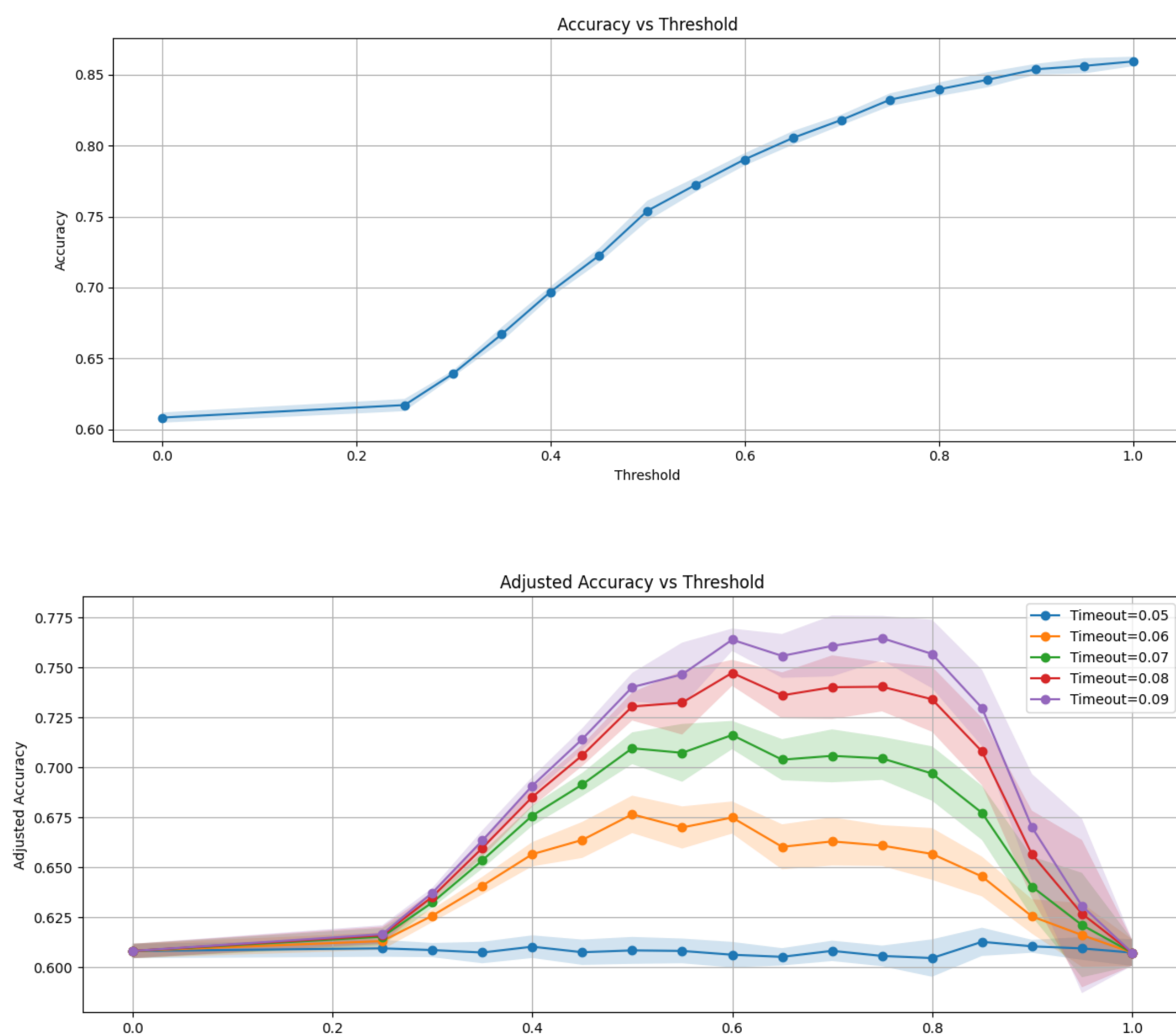
    # Asymmetric adjustment and clip
    delta = compr_grad * (1 - theta) if grad > 0 else compr_grad * theta
    delta = np.clip(delta, -max_delta, max_delta)

    # Threshold update
    s_prev2 = s_prev1
    s_prev1 = theta
    prev_score = score
    theta = theta + delta

    return score, s_prev1, s_prev2, theta
```

RESULTS

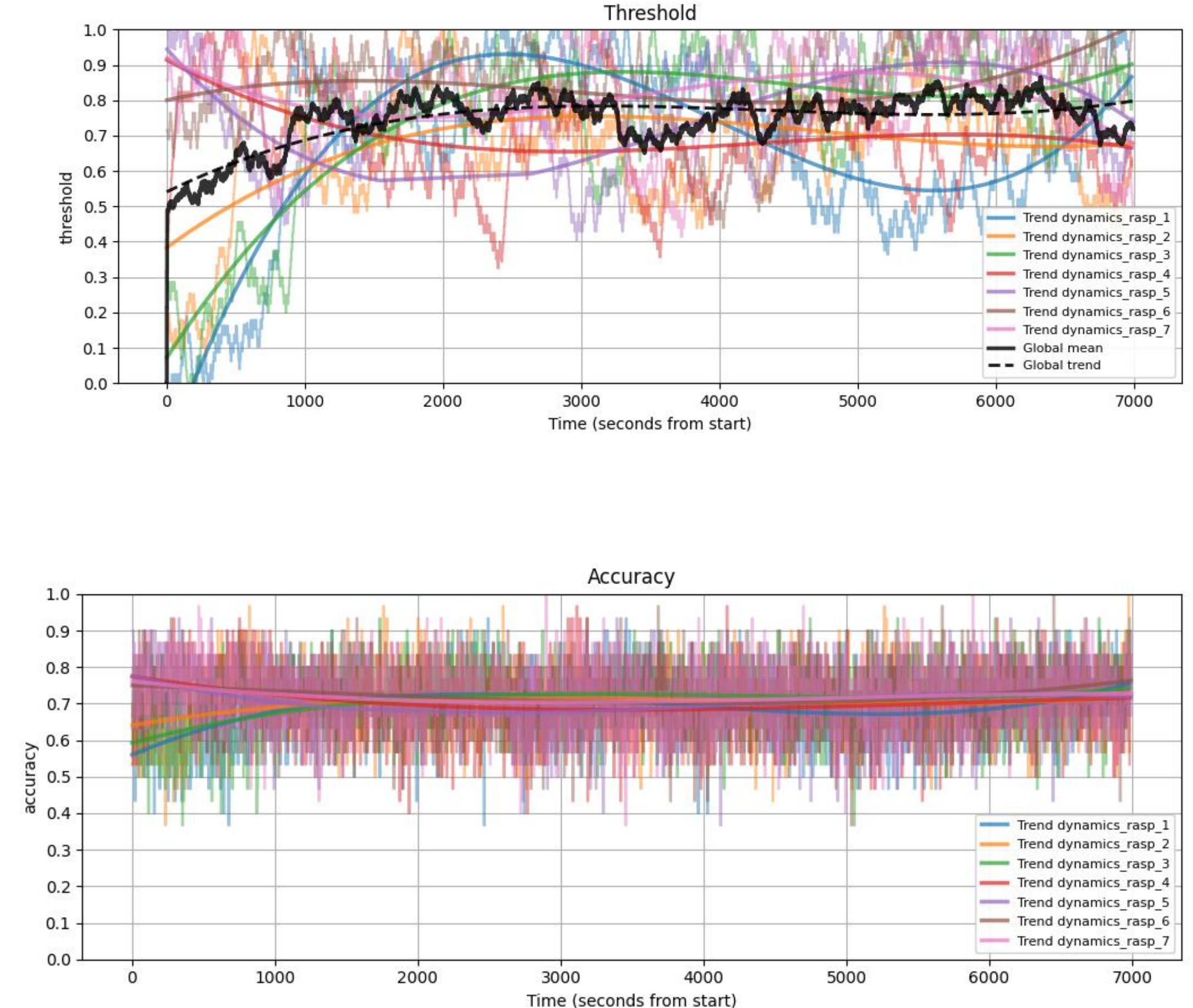
Static Threshold



Samples Distribution



Dynamic Threshold



CONCLUSIONS

By utilizing adaptive thresholding, the system balances server utilization and individual accuracy to ensure fair resource distribution.

Future directions include:

- **Hyperparameter tuning**
- **Scalability analysis**
- **Heterogeneous user scenarios**

REFERENCES

- [1] G. Al-Atat, et al., The Case for Hierarchical Deep Learning Inference at the Network Edge, 2023, 10.1145/3597062.3597278.
- [2] I. Nikoloska et al., Data Selection Scheme for Energy Efficient Supervised Learning, 2021, 10.1109/LCOMM.2020.3034992.
- [3] A. P. Behera, et al., Improved Decision Module Selection for Hierarchical Inference in Resource Constrained Edge Devices, 2024, 2406.09424.
- [4] V. Mancuso, et al., Stateful Versus Stateless Selection of Edge or Cloud Servers Under Latency Constraints, 2022, 10.1109/WoWMoM54355.2022.00022.
- [5] A. Krizhevsky, Learning Multiple Layers of Features from Tiny Images, 2009.
- [6] A. G. Howard, et al., MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, 2017, 1704.04861.
- [7] K. Simonyan, et al., Very Deep Convolutional Networks for Large-Scale Image Recognition, 2015, 1409.1556.

ACKNOWLEDGMENTS

Funded by the European Union through the **FINALITY** grant (101168816). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Commission. Neither the European Union nor the granting authority can be held responsible for them.

INES

Project TSI-064100-2023-34 funded by MINECO/SETeleco and EU-NextGenerationEU/PRTR

DRONAC

This work has been funded by project PID2022-140560OB-I00 (DRONAC) funded by MICIU/AEI /10.13039/501100011033 and ERDF, EU.



Funded by
the European Union

