Edge Intelligence Resource Consumption by UAV-based IR Object Detection

Andrii Polukhin
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine
polukhin@comsys.kpi.ua

Yuri Gordienko
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine

Mairo Leier
Tallinn University of Technology
Tallinn, Estonia
mairo.leier@taltech.ee

Gert Jervan
Tallinn University of Technology
Tallinn, Estonia
gert.jervan@taltech.ee

Oleksandr Rokovyi
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine

Oleg Alienin
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine

Sergii Stirenko
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine

ABSTRACT

The investigation of the feasibility of using the YOLO (You Only Look Once) architecture for object detection in infrared images from unmanned aerial vehicles (UAVs) on low-power devices, specifically the Raspberry Pi, and Orange Pi, is conducted. The study measures the consumption of computing resources for each device, such as inference time (ms), peak power consumption (W), memory consumption (MB), inference energy (J), memory consumption (MB), and storage consumption (MB). It also investigates the correlation between number of model parameters and resource consumption of the different YOLO model sizes. Finally, the study draws conclusions about the expediency and realism of using YOLO on low-power devices for Edge Intelligence and proposes methods of speeding up work. The results show that YOLO can be used effectively on low-power devices with some optimizations to increase performance and energy efficiency.

CCS CONCEPTS

• **Computing methodologies → Object detection**; • **Hardware → Energy metering**.

UAVM '23, November 2, 2023, Ottawa, ON, Canada

ACM ISBN 979-8-4007-0286-0/23/11…$15.00
https://doi.org/10.1145/3607834.3616566

KEYWORDS

Object Detection, Low-Power Device, You Only Look Once, Unmanned Aerial Vehicles, Infrared Thermal Imaging

ACM Reference Format:
Andrii Polukhin, Yuri Gordienko, Mairo Leier, Gert Jervan, Oleksandr Rokovyi, Oleg Alienin, and Sergii Stirenko. 2023. Edge Intelligence Resource Consumption by UAV-based IR Object Detection. In *Proceedings of the 2023 Workshop on UAVs in Multimedia: Capturing the World from a New Perspective (UAVM '23), November 2, 2023, Ottawa, ON, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3607834.3616566

1 INTRODUCTION

The sciences of artificial intelligence (AI) and machine learning (ML) have increased in value due to their vast range of applications [1]. Recently, a lot of focus has been placed on the fundamental computer vision problem of object detection [3, 13]. However, due to the variations in object appearance, lighting, and viewpoint adjustments, it is a challenging task. Convolutional neural networks (CNNs), such as You Only Look Once (YOLO) [9], have demonstrated astounding object detection accuracy and speed. Deploying CNNs on low-power devices, such as small embedded computers or unmanned aerial vehicles (UAVs), is still a challenging task because of the limited computational power available. These devices make it challenging to deploy CNNs rapidly and reliably due to their often low-power central processing units (CPUs), graphics processing units (GPUs), limited memory, and low energy capacity.

Our research aims to investigate this issue by providing a full understanding of YOLO's performance on popular low-power platforms such as Raspberry Pi and Orange Pi. Particularly when it comes to UAV-based object detection, our findings can be helpful for academics and professionals creating computer vision software for low-power devices.

2 RELATED WORKS AND MOTIVATION

In computer vision, object detection is a critical task with numerous real-world applications, including robotics, autonomous vehicles, surveillance, and medical imaging [5]. CNNs have become the most popular approach for object detection because they can capture complicated features from unprocessed images. YOLO is a well-known CNN architecture for object detection and is helpful for real-time performance due to its great accuracy and speed. Implementing CNNs on low-power devices, such as embedded systems, mobile devices, and UAVs, is challenging due to their limited processing capabilities. These devices struggle to carry out object detection tasks with high accuracy and speed due to their poor processor power and memory capacity.

2.1 Infrared Object Detection from UAVs

Infrared images play a crucial role in object detection in UAVs, but low contrast between objects and their background remains a challenge. Various approaches, including deep learning-based methods, aim to overcome this issue, but most studies focus on high-end computing resources, overlooking low-power devices essential for UAV applications [2].

2.2 Low-Power Devices Computing Resources

Low-power devices, such as Raspberry Pi and Orange Pi, are increasingly used in IoT, embedded systems, and UAVs for object detection [4]. However, research on their usage and effectiveness, particularly in infrared image capture, requires further investigation. Consumption and performance are crucial for evaluating their effectiveness in various applications.

2.3 YOLO for Low-Power Devices

YOLO architecture excels in object detection with high accuracy and real-time performance on UAVs [9]. However, its potential for low-power devices remains unexplored. To address this gap, YOLO should be explored for object detection in infrared images from UAVs using low-power devices.

3 MATERIALS AND METHODS

In this work, we aim to measure the performance of low-power devices in executing the YOLO architecture for object detection in infrared images from UAVs. We trained the model on the HIT-UAV dataset, which is comprehensively described in our previous work [7]. To achieve this, we use a set of metrics related to the consumption of computing resources, such as inference time (s), peak power consumption (W), inference energy (J), memory consumption (MB), and storage consumption (MB).

3.1 UAV Infrared Thermal Dataset

We trained and evaluated object detection algorithms on the HIT-UAV dataset [10], presented in the Kaggle Platform in YOLO format [6]. It comprises thousands of infrared thermal images, captured by a UAV in various scenes, including information such as flight altitude, camera perspective, and daylight intensity. The dataset is split into three sets: 2008 photos and 17,628 instances for training, 287 images and 2,460 instances for validation, 571 images and 4,811 instances for testing. See the model performance with this dataset in Figure 4.

3.2 Devices and Hardware

Two primary low-power devices were used for the study:

Raspberry Pi. A low-cost, credit-card-sized computer that was developed for educational purposes. It has a Cortex-A72 processor inside. For our studies, we utilize a Raspberry Pi 4 Model B Rev. 1.1 with 2GB of RAM. For a small fraction of the price and power consumption of entry-level x86 computers, this device provides performance levels equivalent to them.

Orange Pi. Another cheap single-board computer that is comparable to Raspberry Pi in terms of size and capabilities. It is built on a Cortex-A7 quad-core processor. For our investigations, we utilize an Orange Pi One with 512 MB of RAM.

3.3 Evaluation Methods

We used several measures to evaluate the consumption of computing resources for each device, including inference time (s), peak power consumption (W), memory consumption (MB), inference energy (J), and storage consumption (MB). These metrics allowed us to evaluate the overall efficiency and effectiveness of each device when running different versions of the YOLO models.

Inference Time (s). Measures the model's speed in processing input data and generating predictions. Faster inference time is vital for time-sensitive applications like real-time object detection.

Peak Power Consumption (W). Refers to the maximum electrical power used during the inference process. Crucial for assessing device suitability in battery-powered or energy-constrained scenarios.

Memory Consumption (MB). Indicates the RAM used during model inference. Lower memory consumption benefits devices with limited RAM and ensures efficient multitasking.

Inference Energy (J). Represents the total energy consumed by the device for a single inference, providing a comprehensive measure of energy efficiency.

Storage Consumption (MB). Accounts for the model's storage space on the device, essential for devices with limited storage capacity. Lower storage consumption allows for more space for other necessary software or data.

These methods collectively help determine the feasibility and effectiveness of deploying YOLO v5 models on low-power devices, aiding in the selection of suitable models, devices, and configurations for specific use-case scenarios.

3.4 Experimental Design and Procedures

The experimental design and procedures used a rigorous approach to measure resource consumption for each YOLO v5 model [12] on Raspberry Pi and Orange Pi devices, see Figure 3. The **inference time** was measured using 10 separate inference runs, accounting for variations and anomalies. **Peak power consumption** was measured using a wattmeter connected to the power supply of each device, capturing the maximum power draw during the high computational load. **Memory consumption** was measured using

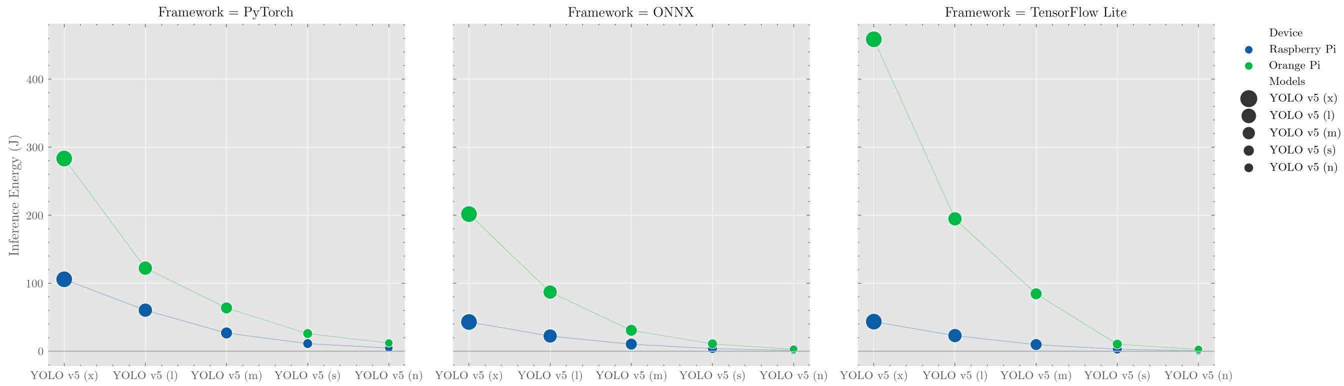

Figure 1: Comparison of Inference Energy (J) vs different model sizes using fp32 data type on Orange Pi and Raspberry Pi.

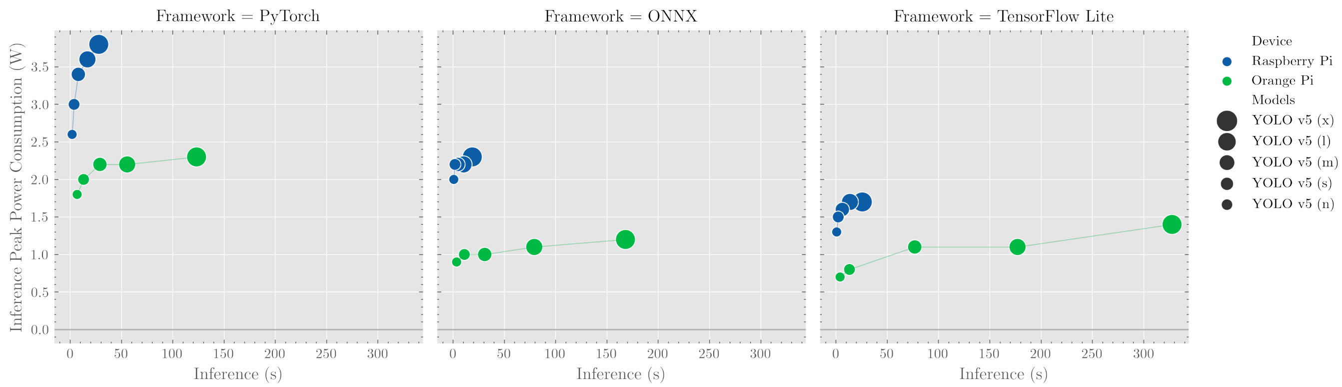

Figure 2: Comparison of Inference Peak Power Consumption (W) vs different Machine Learning Frameworks on Orange Pi and Raspberry Pi.

a Python-based memory profiling package, memory-profiler [8], which monitored and recorded Random Access Memory (RAM) utilization during the model inference line execution. **Inference energy** was calculated by the product of peak power consumption and the corresponding inference time. **Storage consumption** was determined by measuring the amount of storage space required to store the file containing the weights of each model on the device's storage. This meticulous and exhaustive approach offers a comprehensive understanding of the performance and efficiency of different YOLO v5 models on Raspberry Pi and Orange Pi devices.

4 RESULTS AND DISCUSSION

In this section, we explore and discuss the implications of the experimental observations conducted on YOLO v5 models executed on low-power devices such as Raspberry Pi and Orange Pi, using different machine learning frameworks and data types.

We emphasize that we are dealing with IR images that are generally of poorer quality, so the degradation of prediction quality drops at a lower rate when the model is reduced for such coarse data compared to fine data [7]. These results are also consistent with our previous findings of better segmentation on coarse data [11].

Different baseline models. Our experiments indicate a clear advantage of the Raspberry Pi over the Orange Pi in terms of both inference time and memory consumption, see Table 1 and Table 2. This superiority suggests that Raspberry Pi offers a more optimized environment for executing object detection tasks, especially when using infrared imaging from UAVs for tasks such as object detection.

Impact of Model Size and Data Types. As the model size increases, there is a corresponding increase in both inference time and power consumption, regardless of the platform and framework in use. However, intriguingly, the YOLO v5 (s) model exhibits power consumption compared to the more compact (n) variant, presenting it as a viable option for use cases that demand higher accuracy but are constrained by power considerations, see Figure 2 for the details.

Influence of Machine Learning Frameworks. The choice of the machine learning framework significantly affects both inference time and power consumption. While ONNX appears most efficient in terms of memory usage, TensorFlow Lite is the most energy efficient, especially with smaller models. However, as the model size increases, ONNX becomes a preferable choice for energy efficiency.

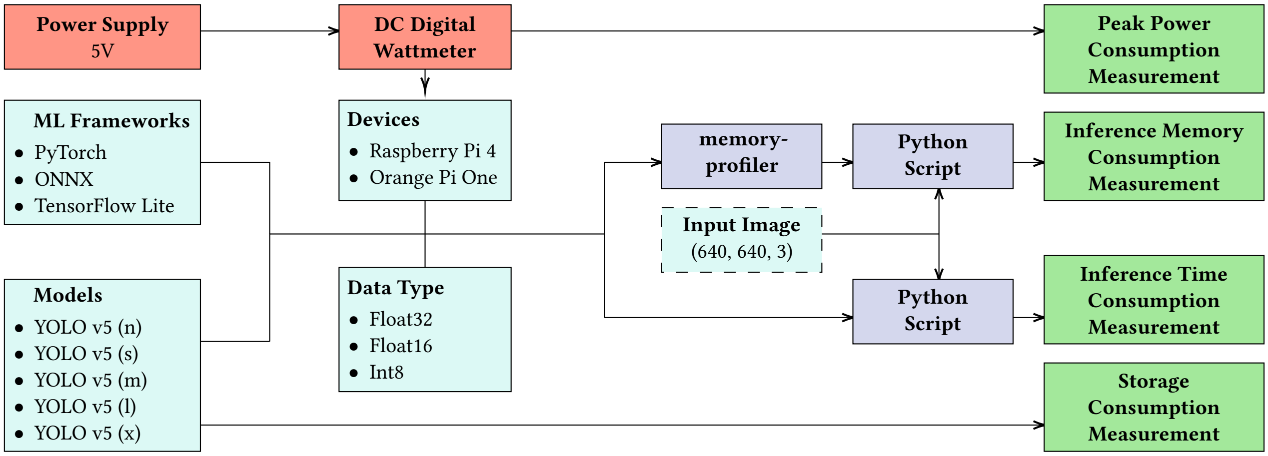

Figure 3: Resource Consumption Measurement Process Visualization.

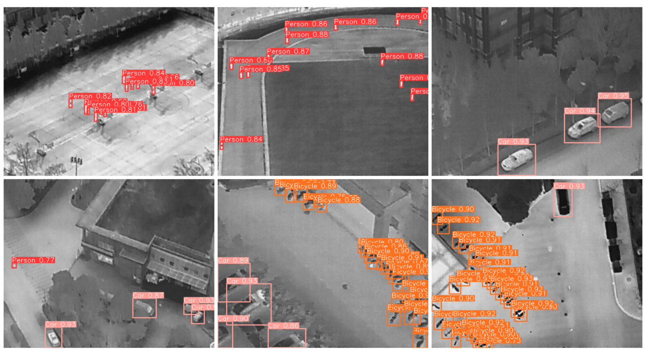

Figure 4: Sample predictions of the YOLO v5 (x).

Meanwhile, PyTorch, demonstrates a consistent performance, offering a viable choice when both memory and power consumption need to be balanced, see Table 1.

Implications for Larger Model Deployment. The deployment of larger YOLO v5 models, particularly under the TensorFlow Lite framework, imposes significant demands on computational resources, leading to increased inference times and power consumption. This observation suggests caution in deploying these models in scenarios requiring real-time object detection, particularly on low-power devices, see Figure 2.

Inference Energy Trade-off. Looking at the variability within each device for each model, it is clear that the Orange Pi One has a wider range of inference energy. This might imply a higher unpredictability or inconsistency in its performance compared to the Raspberry Pi 4. TensorFlow Lite shows the lowest inference energy consumption across the small models (n) and (s) on both the Orange Pi and Raspberry Pi devices, confirming its efficiency for smaller model deployments on low-power devices, see Figure 1.

Memory Consumption During Initialization and Inference. Memory consumed during model initialization and inference forms a significant portion of total resource consumption, with the ONNX

Table 1: Inference Time (s) of YOLO v5 model sizes with float32 data type and different ML frameworks and devices.

		YOLO v5 Model				
	Framework	**(n)**	**(s)**	**(m)**	**(l)**	**(x)**
Orange Pi	PyTorch	6.8	13.0	**29.0**	**55.6**	**123.2**
	ONNX	**3.4**	**11.0**	30.8	79.1	168.1
	TF Lite	4.0	13.1	76.8	177.0	327.7
Raspberry Pi	PyTorch	1.9	3.8	7.9	16.8	27.9
	ONNX	**0.7**	**1.8**	**4.8**	**10.2**	**18.7**
	TF Lite	**0.7**	2.2	6.2	13.6	25.6

framework and lower-precision data types appearing more efficient. Optimizing memory consumption across these stages is crucial in resource-constrained environments, see Table 2.

Future Work. Given the variance in performance depending on the specific use case, hardware configuration, and software optimization, future research could focus on optimizing these factors further for specific applications and environments.

5 CONCLUSION

In conclusion, we demonstrate the feasibility of deploying the YOLO architecture for object detection in infrared images from UAVs on Edge Intelligence low-power devices, with the Raspberry Pi emerging as a more energy-efficient option compared to the Orange Pi. We found that the choice of framework significantly impacts power consumption and performance, with TensorFlow Lite exhibiting low power consumption but high memory usage, while ONNX provided superior runtime performance, especially for larger models. Further, the choice of model size and precision level significantly influenced inference time and power consumption, with smaller models and lower precision levels (especially int8) proving more efficient. The results highlight a clear potential for optimization through careful selection and balancing of hardware, model size, framework, and precision level to fit specific use case requirements.

Table 2: Storage and Memory Consumption (MB) of YOLO v5 model sizes with float32 data type and different ML frameworks.

	Framework	YOLO v5 Model (n)	(s)	(m)	(l)	(x)
Storage Consumption	PyTorch	7.2	**26.9**	80.4	177.0	330.0
	ONNX	7.1	27.2	80.1	**176.0**	**329.0**
	TF Lite	**6.8**	**26.9**	**79.8**	**176.0**	**329.0**
Initialization Memory Consumption	PyTorch	**10.8**	**30.9**	**84.3**	**181.5**	**335.0**
	ONNX	27.2	81.9	205.7	431.0	711.1
	TF Lite	17.1	57.1	163.2	365.5	664.5
Inference Memory Consumption	PyTorch	35.3	72.9	91.2	101.3	116.1
	ONNX	**25.3**	**47.0**	**58.3**	**87.5**	**93.8**
	TF Lite	75.9	106.4	140.2	183.2	209.1

Our observations on memory and storage management emphasize the need for efficient resource utilization, particularly during model inference. Future studies should explore these trade-offs and optimization strategies in greater depth.

ACKNOWLEDGMENTS

This research was in part sponsored by the NATO Science for Peace and Security Programme under grant id. G6032.

REFERENCES

[1] Shaveta Dargan, Munish Kumar, Maruthi Rohit Ayyagari, and Gulshan Kumar. 2020. A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning. *Archives of Computational Methods in Engineering* 27, 4 (Sept. 2020), 1071–1092.

[2] Mate Kristo, Marina Ivasic-Kos, and Miran Pobar. 2020. Thermal Object Detection in Difficult Weather Conditions Using YOLO. *IEEE Access* 8 (2020), 125459–125476.

[3] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. 2020. Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision* 128, 2 (Feb. 2020), 261–318.

[4] Usman Masud, Tareq Saeed, Hunida M. Malaikah, Fezan Ul Islam, and Ghulam Abbas. 2022. Smart Assistive System for Visually Impaired People Obstruction Avoidance Through Object Detection and Classification. *IEEE Access* 10 (2022), 13428–13441.

[5] Ajeet Ram Pathak, Manjusha Pandey, and Siddharth Rautaray. 2018. Application of Deep Learning for Object Detection. *Procedia Computer Science* 132 (Jan. 2018), 1706–1717.

[6] Andrii Polukhin. 2023. HITUAV: A High-Altitude Infrared Thermal Dataset. https://www.kaggle.com/datasets/pandrii000/hituav-a-highaltitude-infrared-thermal-dataset.

[7] Andrii Polukhin, Yuri Gordienko, Gert Jervan, and Sergii Stirenko. 2023. Object Detection for Rescue Operations by High-Altitude Infrared Thermal Imaging Collected by Unmanned Aerial Vehicles. In *Pattern Recognition and Image Analysis*, Antonio Pertusa, Antonio Javier Gallego, Joan Andreu Sánchez, and Inês Domingues (Eds.). Springer Nature Switzerland, Cham, 490–504.

[8] Python Profilers. [n. d.]. memory_profiler. https://github.com/pythonprofilers/memory_profiler. Accessed: June 23, 2023.

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

[10] Jiashun Suo, Tianyi Wang, Xingzhou Zhang, Haiyang Chen, Wei Zhou, and Weisong Shi. 2022. HIT-UAV: A High-altitude Infrared Thermal Dataset for Unmanned Aerial Vehicles.

[11] Vlad Taran, Yuri Gordienko, Alexandr Rokovyi, Oleg Alienin, and Sergii Stirenko. 2020. Impact of ground truth annotation quality on performance of semantic image segmentation of traffic conditions. In *Advances in Computer Science for Engineering and Education II*. Springer, 183–193.

[12] ultralytics. [n. d.]. yolov5. https://github.com/ultralytics/yolov5. Accessed: June 23, 2023.

[13] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. 2023. Object Detection in 20 Years: A Survey. *Proc. IEEE* 111, 3 (March 2023), 257–276.