Efficiency Assessment of Neural Network Classifiers on Single-board Computers for Batch Image Processing

Pylyp Prystavka[1,†], Nykolay Sydorov[2,*,†] ,Olha Cholyshkina [3,†], Borys Stetsenko[4,†]

[1] *The State University "Kyiv Aviation Institute", 1 Lubomyr Huzar Avenue, 03058 Kyiv, Ukraine*
[2] *National Technical University of Ukraine Igor Sikorsky Kyiv Polytechnic Institute, Prospect Beresteiskyi (former Peremohy) 37, 03056, Kyiv, Ukraine*
[3] *Taras Shevchenko National University of Kyiv, 60 Volodymyrska Street, 01033 Kyiv, Ukraine*
[4] *The State University "Kyiv Aviation Institute", 1 Lubomyr Huzar Avenue, 03058 Kyiv, Ukraine*

Abstract

The aim of this study is to experimentally evaluate the performance of contemporary deep learning models under constrained computational resources and to establish formalized dependencies between inference speed and input batch size. The Raspberry Pi 4 Model B single-board computer was used as the computing platform, representing a typical example of low-power, resource-limited hardware employed onboard unmanned aerial vehicles (UAVs).
In contrast to existing YOLO-based approaches that remain ill-suited for low-resource UAV platforms, this study proposes an alternative method focused on modeling classification time using linear regression. The results provide a framework for the development of onboard vision subsystems for domestic UAVs, capable of near-real-time image analysis under batch-processing constraints.
This work presents, for the first time, linear regression models that describe the classification time of neural networks with varying architectures—specifically, mobile, balanced, and deep-level models—depending on the number of images in the input batch. The statistical significance of the resulting regression models has been experimentally validated, as well as their consistency with real-world measurements obtained during actual single-board computer operation. These regression equations enable inference time prediction without the need for repeated empirical testing, thus significantly improving the efficiency of neural system configuration in embedded environments.
Based on the developed models, boundary conditions were identified for the applicability of each considered architecture to ensure near-real-time data processing. The results demonstrate that some models can handle relatively large data batches without exceeding critical time thresholds, whereas others, despite offering higher classification accuracy, exhibit excessive computational complexity and require hardware acceleration or optimization.
The relevance of this research is driven by the growing demand for autonomous image analysis systems, particularly in the context of UAV deployment for military operations, reconnaissance, search and rescue missions, and monitoring applications. The proposed approach can be integrated into hardware-software systems to enable adaptive selection of neural architectures according to operational conditions and resource constraints. This creates a foundation for the further development of intelligent UAV systems with enhanced autonomy.

Keywords

unmanned aerial vehicles (UAVs), aerial imaging data, neural network classifiers, machine learning, limited computational resources.

Workshop "Intelligent information technologies" UkrProg-IIT`2025 co-located with 15th International Scientific and Practical Programming Conference UkrPROG'2025, May 13-14, 2025, Kyiv, Ukraine
[*] Corresponding author.
[†] These authors contributed equally.
NykSydorov@gmail.com (N.Sydorov); olha.cholyshkina@knu.ua (O.Cholyshkina); chindakor37@gmail.com (P.Prystavka); bs7206426@gmail.com (B. Stetsenko)
0000-0002-0360-2459 (P.Prystavka); 0000-0003-08000-1668 (N.Sydorov), 0000-0002-0681-0413 (O.Cholyshkina); 0009-0007-6588-9532 (B. Stetsenko)

1. Introduction

One of the key challenges in processing aerial imaging data is to ensure the high efficiency of visual analysis systems under the constrained computational resources typical of onboard equipment used in unmanned aerial vehicles (UAVs) [1]. Traditional methods for processing streaming video from surveillance cameras require substantial computational power and time, rendering them unsuitable for real-time operation—particularly in scenarios where the timely detection of target objects is critical [2]. In military applications, for instance, delays in object identification may lead to a loss of tactical advantage, while in search and rescue operations, they may reduce the effectiveness of victim detection [3].

This issue is further exacerbated by the limited energy supply available to UAVs, necessitating the use of energy-efficient algorithms. The application of optimized models for real-time classification and segmentation can significantly reduce the volume of data transmitted to ground stations, thereby improving the overall autonomy and functionality of the system [4].

To enable effective processing of video streams from UAV onboard cameras, an approach that combines high processing speed with an acceptable level of detection and classification accuracy is required. Given the constrained computational resources of such platforms, real-time processing of high-resolution full frames using deep models is largely impractical due to considerable hardware demands.

This study proposes a two-stage image processing procedure. It involves an initial downscaling of the frame resolution, followed by detailed processing of selected fragments using more accurate classification models that are capable of efficient operation on single-board computers. This approach reduces the volume of input data processed at each stage, thereby decreasing the overall computation time and increasing the system's efficiency.

However, it is important to note that downscaling an image is functionally equivalent to smoothing or filtering out high-frequency components. This may result in the loss of critical information, particularly when target objects are small or when images are captured from high altitudes. Consequently, the likelihood of missing objects of interest increases—an important drawback in tasks that require precise detection and localization.

Therefore, optimizing video stream processing onboard UAVs requires careful balancing of processing speed and image informativeness. This can be achieved through a combination of downscaling, pre-filtering, and the use of adapted deep models.

We begin by reviewing some of the most widely used deep learning models for object detection in UAV-acquired imagery.

YOLO (You Only Look Once) is one of the most popular deep learning architectures for real-time object detection [5]. Its main advantage lies in simultaneously determining object boundaries and performing classification in a single network pass, enabling high-speed inference with acceptable accuracy (up to 75% mAP in YOLOv8 on the COCO dataset [6]). YOLO performs well in scenarios with fixed camera positions or predictable movement, such as ground vehicle navigation systems.

However, YOLO presents several critical limitations when applied to aerial monitoring with UAVs. First, it is poorly adapted to scale variations and perspective distortions, which are inevitable due to changes in altitude and camera angles during flight. Second, in dynamic environments with highly variable backgrounds and lighting conditions, YOLO often exhibits reduced accuracy due to limited contextual adaptability. Even simplified versions like Tiny-YOLO remain too resource-intensive for deployment on single-board computers such as the Raspberry Pi or OrangePi.

Thus, despite its overall effectiveness in ground-based applications, YOLO is not an optimal choice for onboard deployment in UAVs operating under constrained computational conditions and complex aerial dynamics. To overcome these limitations, this paper proposes a different approach aimed at quantifying classification performance on single-board computers under batch processing conditions. The proposed method involves estimating classification time via linear regression models as a function of batch size for various neural architectures. This approach enables a deeper

understanding of performance boundaries and supports the development of onboard computer vision solutions tailored to the constraints of domestic UAVs.
Several other modern architectures also merit discussion for their potential applicability to various computer vision tasks involving UAV imagery.
Vision Transformers (ViT) [7] employ a self-attention mechanism that effectively captures spatial dependencies in images. This architecture demonstrates high accuracy, especially on large datasets. However, its significant computational requirements and dependence on powerful hardware limit its use in low-resource environments like single-board computers.
NASNet (Neural Architecture Search Network) [8] is the result of automated architecture optimization tailored for classification and detection tasks. While it delivers high accuracy and can be adapted to resource constraints, it is impractical for real-time deployment due to its resource demands.
Faster R-CNN is one of the most widely used architectures for object detection, offering excellent accuracy in complex scenes with multiple objects [9]. Nevertheless, its computational complexity and substantial processing delays make it unsuitable for real-time applications on platforms like the Raspberry Pi.
SqueezeNet [10], by contrast, is explicitly designed for resource-constrained environments. Its compact architecture (~1.2 million parameters) enables efficient operation on embedded platforms. However, its main drawback remains the notably lower recognition accuracy compared to more advanced models.
Each of these models exhibits specific strengths, yet all suffer from considerable limitations. Their complexity, energy consumption, and processing delays hinder real-time deployment on autonomous UAV platforms. This creates a demand for thorough analysis of models that not only deliver acceptable classification accuracy but also meet strict performance and energy-efficiency criteria.
The objective of this study is to experimentally assess the performance of contemporary deep learning models under limited computational resources and to formalize the relationship between processing speed and input batch size. The Raspberry Pi 4 Model B was used as the target device for evaluation, serving as a prototypical low-power, resource-constrained hardware platform commonly employed onboard UAVs.
For the experiments, four widely adopted convolutional neural network architectures were selected, differing in complexity, accuracy, and computational requirements: EfficientNetV2S, MobileNetV2, ResNet50, and ResNet101. This selection is based on their prevalence in applied computer vision tasks, widespread support across deep learning frameworks, and availability of open-source models with well-documented performance metrics. Furthermore, these architectures represent distinct categories in terms of the trade-off between accuracy and performance—ranging from lightweight mobile models to deep high-precision networks. Evaluating their behavior on the Raspberry Pi platform enables the formulation of informed recommendations regarding architecture selection based on task-specific requirements and computational constraints.

2. Materials and Methods

The following neural network models were considered in this study.
EfficientNetV2S (Efficient Network V2 Small) is a representative of the second generation of deep convolutional neural networks, optimized for fast inference and training. The model was developed using Neural Architecture Search (NAS) and the compound scaling strategy. A key feature of EfficientNetV2S is the combination of traditional mobile blocks (MBConv) with the newer Fused-MBConv blocks, which significantly reduces processing time without compromising accuracy. Due to its balanced architecture, EfficientNetV2S achieves high performance on the ImageNet benchmark with a relatively small number of parameters, making it suitable for both server-based and embedded applications [11].

MobileNetV2 is a lightweight neural network architecture designed for use on mobile and embedded devices. It combines depthwise separable convolutions with inverted residual blocks and linear bottlenecks, significantly reducing computational overhead. A unique characteristic of MobileNetV2 is the transformation order within each block: instead of reducing the number of output channels as in conventional networks, the number of channels is first expanded and then compressed. This helps retain informative features and minimizes information loss during convolutional processing. The architecture is especially well-suited for real-time tasks under resource-constrained conditions [12].
ResNet50 (Residual Network-50) is a classic deep convolutional neural network architecture based on shortcut (residual) connections. ResNet was introduced to address the vanishing gradient problem in very deep networks. ResNet50 employs residual blocks with three layers (bottleneck blocks), enabling effective signal propagation across deep layers. Due to its high accuracy and moderate computational requirements, ResNet50 is widely used in various computer vision tasks [13].
ResNet101 is an extended version of the ResNet architecture, consisting of 101 convolutional layers and using the same bottleneck blocks as ResNet50. The increased depth allows the model to capture more complex features, which enhances classification performance, particularly on large datasets. However, the larger number of parameters and FLOPs increases processing time and makes ResNet101 less suitable for real-time inference. This model is best suited for high-accuracy tasks where computational resources are not a limiting factor [13].

Table 1
Key Model Characteristics

Model	**Depth**	**Parameters**	**FLOPs**	**Top-1 Accuracy**	**Real-time Performance**
EfficientNetV2S	~479 blocks	~22.1M	8.4B	84.9%	High
MobileNetV2	53 layers	~3.4M	0.3B	72.0%	Very high
ResNet50	50 layers	~25.6M	4.1B	76.2%	Medium
ResNet101	101 layers	~44.5M	7.8B	77.4%	Low

For model deployment and testing, the hardware platform selected was the Raspberry Pi 4 Model B single-board computer, in order to evaluate the suitability of the networks for integration onboard UAVs.
To ease the processing load on the models by limiting the number of regions passed for classification, a study [2] explored a fast filtering method to extract informative image fragments from each frame. Specifically, it was demonstrated that for a frame of 1920×1080 pixels, the authors' custom rapid selection algorithm for informative fragments (each 64×64 pixels) processes the frame using a sliding window in less than 0.1 seconds. Thus, within this time frame, a classification model receives a batch consisting of a variable number of 64×64-pixel images. In general, the number of images in a batch may range from several dozen to several hundred, depending on the hyperparameters of the frame processing procedure. The authors of the study suggest that this number is primarily determined by two factors: the texture variability of individual image fragments and the low likelihood of encountering multiple informative fragments within a single frame [15].
The goal of this work is to determine the maximum allowable batch size for which the classification model's inference time does not exceed 0.85 seconds. This duration ensures that both

the information filtering and the classification of informative image fragments are guaranteed to be completed within one second on the given computational device.
This paper presents the results of an experimental evaluation of the performance of four selected convolutional neural network architectures: EfficientNetV2S, MobileNetV2, ResNet50, and ResNet101.
For each model, a series of inference time measurements (i.e., image processing times) was conducted across varying input batch sizes, ranging from 1 to 512 images. During testing, a fixed hardware configuration and software environment were maintained to eliminate external factors that could influence the results. All measurements were performed on Raspberry Pi 4 Model B single-board computers, emulating the resource-constrained environment typical of onboard UAV systems. This platform presents significant limitations in terms of CPU power, RAM, and energy consumption, thereby providing a realistic simulation of real-time operation conditions.
For further analysis, the data were grouped into six batch size intervals: 0–16; 17–32; 33–64; 65–128; 129–256; and 257–512. For each interval, observations were aggregated, and the average batch processing time was computed in seconds (Table 2). Outlier data were removed using the interquartile range (IQR) method to ensure statistical robustness of the results.

Table 2
Comparison of Batch Processing Time Across Deep CNN Architectures

BatchRange	**EfficientNetV2S**	**MobileNetV2**	**ResNet101**	**ResNet50**
0-16	0.669	0.324	0.95	0.633
17-32	1.259	0.51	1.976	1.204
33-64	2.078	0.74	3.684	1.992
65-128	3.683	1.248	7.142	3.56
129-256	7.088	2.291	13.979	6.638
257-512	13.625	4.369	22.539	13.06

3. Result and Discussion

Based on the results presented in Table 2, the following conclusions can be drawn.
MobileNetV2 demonstrates the lowest average processing time across all batch sizes, confirming its efficiency for mobile and real-time applications. Its architecture, characterized by a small number of parameters and low computational complexity, enables high-speed performance even under increased processing loads.
EfficientNetV2S provides the highest classification accuracy among all models considered, while also maintaining high performance. This makes it an optimal choice for systems where a balance between recognition quality and processing speed is essential.
ResNet50 exhibits moderate inference time and represents a well-balanced trade-off between accuracy and speed. As a result, it is frequently used as a baseline model in production-level computer vision systems.
ResNet101 is the slowest of all the models across all batch size intervals. Despite offering slightly higher accuracy compared to ResNet50, it demonstrates low efficiency in terms of performance, which limits its practical applicability in real-time tasks.

For all models, an approximately linear relationship between batch size and processing time is observed; however, the rate of increase (i.e., the slope of the regression line) varies by architecture. Further results are presented below.
Figure 1 illustrates the empirical relationship between input batch size and the average processing time per batch for the EfficientNetV2S model. Prior to regression modeling, the data were cleaned to remove statistical outliers.

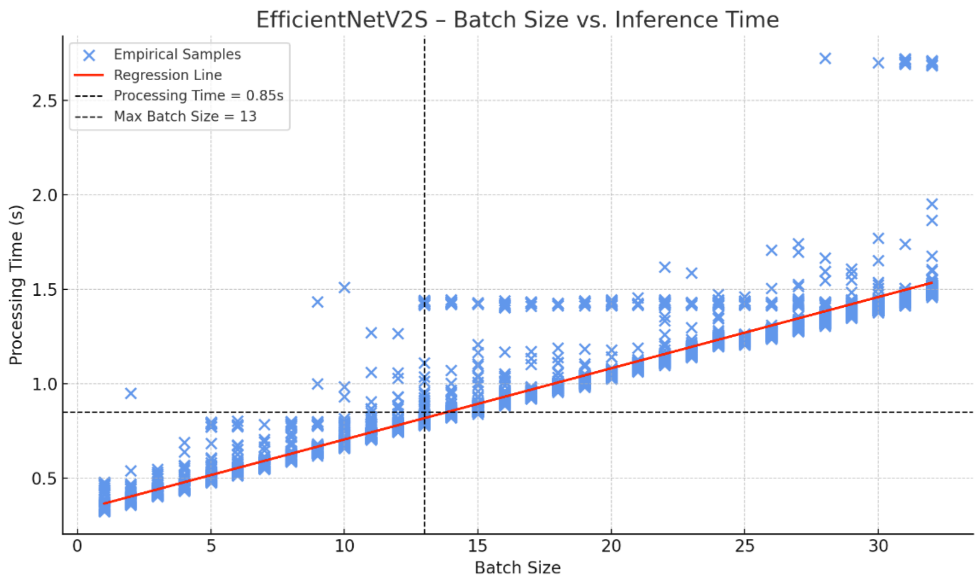

Figure 1: Inference Time Dependency on Batch Size for EfficientNetV2S With Linear Regression Fit.
Regression Results
The equation of the linear regression model is:
$\bar{y}(x) = 0.03519x + 0.36807$

where
$\bar{y}(x)$is the average processing time (in seconds), and x is the batch size.
The coefficient of determination is R^2=0.95618.
The red line in Figure 1 represents the linear regression result, which approximates the empirical relationship within the investigated range (batch size up to 32). The coefficient of determination R^2=0.95618 18 indicates a strong linear relationship between batch size and processing time.
The horizontal dashed line marks the threshold for acceptable processing time, set at 0.85 seconds—corresponding to typical real-time constraints when accounting for the frame pre-filtering step. The vertical dashed line intersects the regression curve at the point corresponding to the maximum permissible batch size that does not exceed the specified time limit. According to the regression results, this threshold for the given model is 13 images per batch.

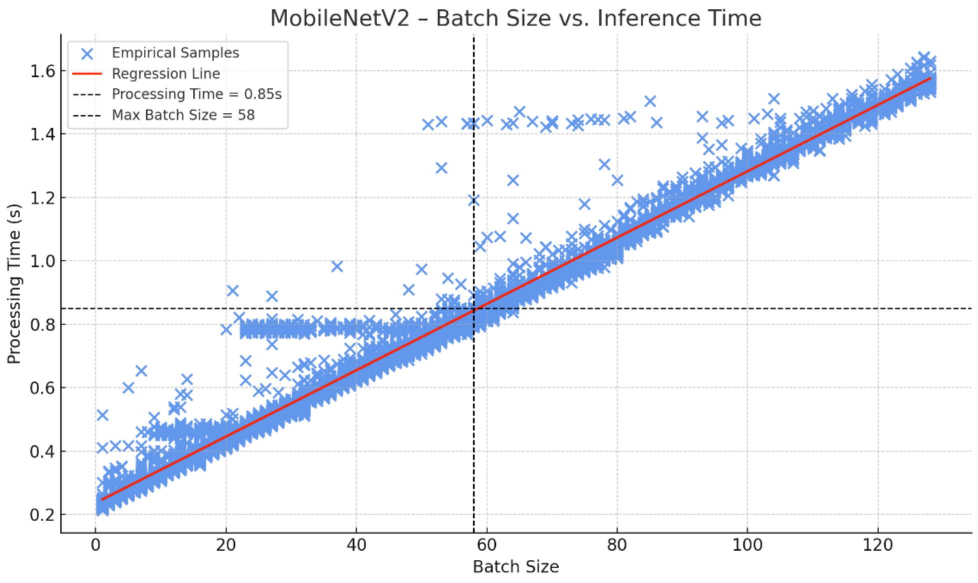

Figure 2: Inference Time Dependency on Batch Size for MobileNetV2 With Linear Regression Fit.
Regression Results
The equation of the linear regression model is:

$$\bar{y}(x) = 0.01046x + 0.23673,$$

where
$\bar{y}(x)$is the average processing time (in seconds), and x is the batch size.
The coefficient of determination is R^2=0.97811.
This indicates a very high degree of agreement between the model and the empirical data.
The maximum batch size that ensures processing within 0.85 seconds is 58 images per batch.
Figure 3 presents the results for the ResNet50 model.

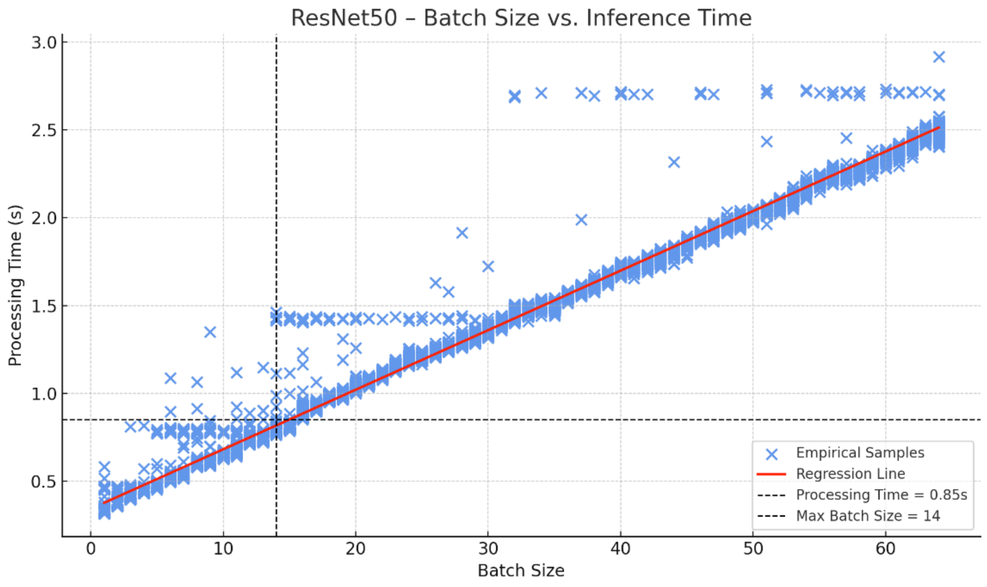

Figure 3: Inference Time Dependency on Batch Size for ResNet50 With Linear Regression Fit.
Regression Results
The equation of the linear regression model is:
$\bar{y}(x) = 0.03247x + 0.38786$
where
$\bar{y}(x)$is the average processing time (in seconds), and x is the batch size.

The coefficient of determination is R^2=0.99373. This value indicates an excellent linear fit to the empirical data. Maximum batch size processed within 0.85 seconds – 14.

Figure 4 illustrates the relationship between the processing time per image and the input batch size for the ResNet101 model, based on a subset of the experimental data (batch sizes up to 32 included for improved visualization).

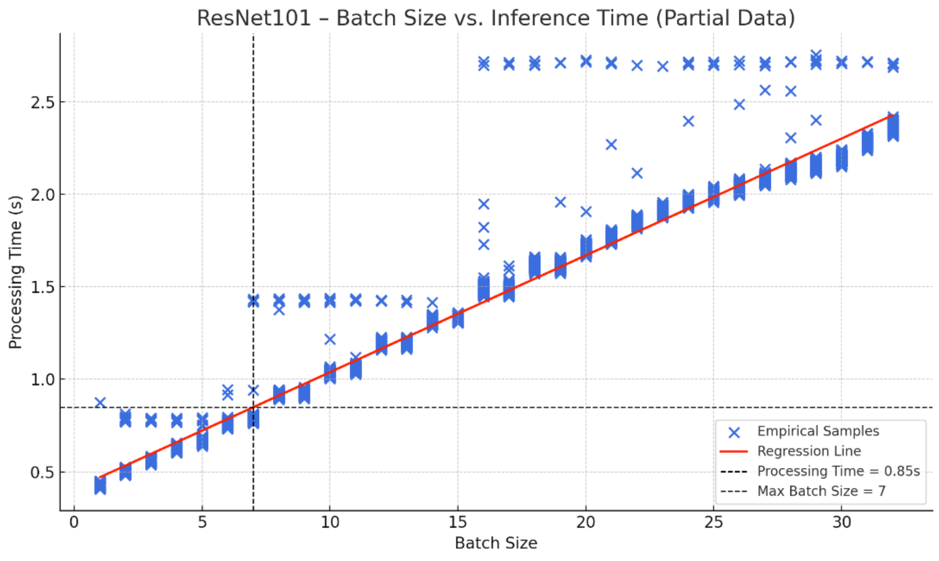

Figure 4: Inference Time Dependency on Batch Size for ResNet101 With Linear Regression Fit.
Regression Results
The equation of the linear regression model is:

$$\bar{y}(x) = 0.06316x + 0.40654$$

where
$\bar{y}(x)$is the average processing time (in seconds), and x is the batch size.
The maximum batch size that allows processing within 0.85 seconds is 7 images.

A summary table (Table 3) is provided below, presenting a comparative overview of the models at the processing time threshold of 0.85 seconds. The table includes key characteristics: regression slope, intercept, coefficient of determination, maximum allowable batch size, and Top-1 classification accuracy.

Table 3
Comparative Analysis of CNN Architectures: Inference Efficiency and Accuracy Under Real-Time Constraints

Model	**Max Batch (≤0.85s)**	**Slope (s/img)**	**Intercept (s)**	**R²**	**Top-1 Accuracy (%)**
EfficientNetV2S	13	0.0352	0.3681	0.956	84.9
MobileNetV2	58	0.0105	0.2302	0.993	72
ResNet50	14	0.0329	0.3325	0.981	76.2
ResNet101	7	0.0632	0.4065	0.973	77.4

The results presented in Table 3 support the following conclusions.
EfficientNetV2S demonstrates the highest classification accuracy (84.9%) while maintaining acceptable inference speed, with a maximum batch size of 13 under the 0.85-second threshold. This enables group image processing even in near real-time scenarios, making it one of the most suitable models for tasks where high recognition precision is critical.
MobileNetV2 offers the best performance in terms of speed, with the largest maximum batch size (58) and the lowest regression slope (0.0105 s/image). However, it yields the lowest Top-1 accuracy

(72%) among the models considered. This makes it an attractive option for applications where speed is paramount and accuracy is of secondary importance.
ResNet50 provides a balanced compromise between inference speed and accuracy: it supports batches of up to 14 images within the 0.85-second threshold and achieves a classification accuracy of 76.2%. This makes it suitable for a wide range of tasks, particularly those requiring moderate depth of analysis and computational robustness.
ResNet101 achieves relatively high accuracy (77.4%) but significantly lower throughput, supporting only 7 images per batch within the same time limit. Due to its computational complexity, it is the least suitable model for deployment on resource-constrained devices.

4. Conclusions

The conducted research focused on solving the classification problem under constrained computational conditions, aiming to provide practical solutions for the development of onboard vision systems for domestic UAV applications. Based on an analysis of existing methods, including YOLO-based models that exhibit significant limitations in aerial environments, an alternative approach was proposed. To support this, the authors constructed regression models describing classification time as a function of batch size for several neural network architectures. The obtained results may serve as a foundation for future research in the development of onboard image processing technologies for aerial surveillance systems.In this study, linear regression models were developed for the first time to describe the classification time of four neural network architectures—EfficientNetV2S, MobileNetV2, ResNet50, and ResNet101—as a function of the number of 64×64-pixel images in the input batch. The statistical significance and adequacy of the regression models were substantiated, confirming their alignment with empirical measurement results.
For each model, the maximum permissible batch size that ensures processing within 0.85 seconds (real-time mode) on a Raspberry Pi 4 Model B single-board computer was determined.
The constructed models enable accurate quantitative estimation of inference time without the need for repeated physical measurements, which is particularly valuable for rapid performance evaluation on low-power devices.
MobileNetV2 demonstrated the highest performance, enabling the processing of up to 58 images per batch within the 0.85-second threshold, with a strong regression fit (R^2=0.993). This makes it an appropriate choice for real-time systems with limited computational resources.
EfficientNetV2S, offering higher classification accuracy (84.9%), supports batches of up to 13 images. The high coefficient of determination (R^2=0.956) confirms the reliability of the regression model for this architecture.
ResNet50 achieves a batch size of 14 images, with strong approximation reliability (R^2=0.981), although it incurs higher computational costs compared to MobileNetV2.
ResNet101 showed the lowest throughput—only 7 images per batch—due to its substantial computational complexity. Despite its high classification accuracy (77.4%), this architecture requires hardware acceleration to be viable for real-time applications.
The results of this study provide a solid foundation for the informed selection of neural network architectures in autonomous image analysis tasks, including onboard deployment in UAVs for monitoring, inspection, and reconnaissance.
The application of these findings is relevant in domains requiring rapid image analysis, particularly in military operations, intelligence gathering, search and rescue missions, critical infrastructure inspection, agrotechnology, and logistics. Furthermore, the results may be integrated into modern monitoring and data analysis platforms to enhance their performance, adaptability to dynamic conditions, and overall real-time efficiency.

In conclusion, the outcomes of this work have practical value both for direct implementation in modern embedded AI systems and for guiding the design of next-generation intelligent image analysis modules with a focus on performance, autonomy, and energy efficiency.

Declaration on Generative AI

During the preparation of this work, the authors used GPT-4.o for figures 1-4 in order to generate images. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] P., Prystavka, A., Chyrkov Suspicious Object Search in Airborne Camera Video Stream // In: Z., Hu, S., Petoukhov, I., Dychka, M., He (eds) Advances in Computer Science for Engineering and Education. ICCSEEA 2018. Advances in Intelligent Systems and Computing, vol. 754. Springer, Cham, 2018. – P. 340–348. – DOI: 10.1007/978-3-319-91008-6_34

[2] P., Prystavka, A., Shevchenko, I., Rokitianska Comparative Analysis of Detector-Tracker Architecture for Object Tracking Based on SBC for UAV // Proceedings of the 2024 IEEE 7th International Conference on Actual Problems of Unmanned Aerial Vehicles Development (APUAVD 2024). – 2024. – P. 175–178. – DOI: 10.1109/APUAVD64488.2024.10765897

[3] P., Prystavka, A., Chyrkov, V., Sorokopud, V., Kovtun Automated Complex for Aerial Reconnaissance Tasks in Modern Armed Conflicts // CEUR Workshop Proceedings, vol. 2588, 2019. – P. 57–66. – [Online]. Available: https://ceur-ws.org/Vol-2588/paper6.pdf

[4] P., Prystavka, O., Cholyshkina, S., Dolgikh, D., Karpenko Automated Object Recognition System Based on Convolutional Autoencoder // 2020 10th International Conference on Advanced Computer Information Technologies (ACIT). – 2020. – P. 830–833. – DOI: 10.1109/ACIT49673.2020.9208945

[5] J., Redmon, A., Farhadi YOLO9000: Better, Faster, Stronger // Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2017. – P. 6517–6525. – DOI: 10.1109/CVPR.2017.690

[6] Y., Tian, S., Wang, E., Li, G., Yang, Z., Liang, M., Tan MD-YOLO: Multi-scale Dense YOLO for Small Target Pest Detection // Computers and Electronics in Agriculture. – 2023. – Vol. 213. – Article 108233. – DOI: 10.1016/j.compag.2023.108233

[7] A., Dosovitskiy, L., Beyer, A., Kolesnikov, et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv:2010.11929 [cs.CV], 2020. https://arxiv.org/abs/2010.11929

[8] B., Zoph, V., Vasudevan, J., Shlens, Q. V., Le, Learning Transferable Architectures for Scalable Image Recognition. arXiv:1707.07012 [cs.CV], 2017. https://arxiv.org/abs/1707.07012

[9] S., Ren, K., He, R., Girshick, J., Sun Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. arXiv:1506.01497 [cs.CV], 2015. https://arxiv.org/abs/1506.01497

[10] F. N., Iandola, S., Han, M. W., Moskewicz, et al. SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and <0.5MB Model Size. arXiv:1602.07360 [cs.CV], 2016. https://arxiv.org/abs/1602.07360

[11] M., Tan, & Q. Le, (2021). EfficientNetV2: Smaller Models and Faster Training. arXiv preprint arXiv:2104.00298. https://doi.org/10.48550/arXiv.2104.00298

[12] M., Sandler, A., Howard, M., Zhu, A., Zhmoginov, & L. C., Chen (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

[13] K., He, X., Zhang, S., Ren, & J., Sun (2016). Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778. https://doi.org/10.1109/CVPR.2016.90

[14] P., Prystavka, A., Zhultynska Spline Approaches for Anomaly Detection in UAV-Based Aerial Surveillance // Proceedings of the 2024 IEEE 7th International Conference on Actual Problems of Unmanned Aerial Vehicles Development (APUAVD 2024). – 2024. – P. 187–190. – DOI: 10.1109/APUAVD64488.2024.10765898

[15] V., Zivakin, O., Kozachuk, P., Prystavka, O., Cholyshkina Training set AERIAL SURVEY for Data Recognition Systems From Aerial Surveillance Cameras // CEUR Workshop Proceedings. – 2022. – Vol. 3347. – P. 246–255. – [Online]. Available: https://ceur-ws.org/Vol-3347/Paper_21.pdf.

A. Online Resources

The ceur-art template for Word can be downloaded at https://ceur-ws.org/Vol-XXX/.