Volume 11 Issue 1 Year 2026 | Page 276-287 ISSN: 2527-9866
*Received:* ***13-01-2026*** | *Revised:* ***21-01-2026*** | *Accepted:* ***10-02-2026***

# Performance Analysis of YOLOv11 Integrated with Lightweight Backbones (MobileNetV2, GhostNet, ShuffleNet V2) for Cigarette Detection

**Kevin Andreas[1], Yohannes[2], Meiriyama[3]**
[1,2]Multi Data Palembang University, Palembang, South Sumatra, Indonesia, 30113
*e-mail: kevinandreas@mhs.mdp.ac.id[1], yohannesmasterous@mdp.ac.id[2], meiriyama@mdp.ac.id[3]*
*Correspondence: kevinandreas@mhs.mdp.ac.id

***Abstract:*** *Cigarette object detection in indoor environments plays a vital role for enforcing smoke-free zone regulations and protecting public health from secondhand smoke exposure. This study investigates the performance of YOLOv11n architecture integrated with three lightweight backbone modifications (MobileNetV2, GhostNet, and ShuffleNet V2) for real-time cigarette detection with the aim of achieving efficiency suitable for potential deployment on resource-constrained edge devices. Comprehensive experiments were conducted using the Cigar Detection Dataset comprising 5,333 images, augmented to 8,890 samples through horizontal flipping and brightness adjustment techniques. All models were trained for 100 epochs using SGD optimizer on NVIDIA Tesla T4 GPU. The evaluation metrics included detection accuracy (mAP@0.5, mAP@0.5:0.95, Precision, Recall, F1-score) and computational efficiency (parameters, model size, GFLOPs, FPS). Experimental results demonstrate that the pretrained YOLOv11n baseline achieves the highest detection accuracy with mAP@0.5 of 0.8072 and Precision of 0.8688. Among lightweight backbone variants, ShuffleNet V2 (0.5x) provides the most compact solution with only 2.28M parameters and 4.73 MB model size, while ShuffleNet V2 (0.75x) offers optimal balance between accuracy (mAP@0.5: 0.7430) and efficiency with only 0.95% accuracy degradation compared to 1.0x variant. These findings provide practical guidance for selecting appropriate model configurations based on deployment constraints in smoke-free area monitoring systems.*

***Keywords:*** *cigarette detection, YOLOv11, MobileNetV2, GhostNet, ShuffleNet V2*

## 1. Introduction

Tobacco consumption represents a significant global health challenge causing millions of deaths annually [1], with harmful effects extending to non-smokers through secondhand smoke exposure [2]. In Indonesia, regulatory frameworks such as Government Regulation No. 109 of 2012 have been implemented to establish smoke-free areas [3]. However, the enforcement of these regulations remains ineffective due to the labor-intensive nature of manual surveillance [4], a challenge underscored by the persistent smoking prevalence of approximately 70 million active smokers recorded by the Central Bureau of Statistics (Badan Pusat Statistik/BPS) [5]. This persistent challenge motivates the development of automated detection systems capable of identifying cigarette objects in real-time.

However, implementing such automated systems presents significant technical challenges. Unlike detecting large objects, cigarettes are visually distinct due to their extreme aspect ratios and small scale relative to the image frame. Visual detection of cigarettes is particularly difficult because the object is relatively small and frequently occluded by the smoker's hand or body [6]. Furthermore, standard detection models often struggle to balance the high resolution required for such fine-grained feature extraction against the computational constraints of real-time surveillance hardware, as small object detection remains a persistent difficulty in general object detection architectures [7].

Deep learning-based object detection has emerged as a promising solution for visual surveillance applications. The YOLO (You Only Look Once) family of detectors, introduced by Redmon et al. [8], revolutionized real-time object detection by framing detection as a single regression problem. Subsequent iterations including YOLOv5, YOLOv8, and the latest YOLOv11 have progressively improved detection accuracy and inference efficiency [9][10]. YOLOv11 introduces architectural enhancements including the C3k2 (Cross Stage Partial with 3x3 kernels) blocks and SPPF (Spatial Pyramid Pooling Fast) module for more effective feature extraction [11]. However, deploying these complex models on edge devices with limited computational resources, such as embedded systems, mobile devices, and IoT sensors, presents significant challenges related to memory constraints, power consumption, and inference latency.

Lightweight convolutional neural network (CNN) architectures address these deployment challenges by reducing model complexity while maintaining acceptable detection performance. MobileNetV2 [12] employs inverted residual blocks with linear bottlenecks and depthwise separable convolutions, achieving substantial parameter reduction compared to standard convolutions. GhostNet [13] generates feature maps through linear transformations of intrinsic features, reducing computational cost by minimizing redundant computations. ShuffleNet V2 [14] implements channel split and shuffle operations following practical design guidelines derived from empirical efficiency analysis on various hardware platforms.

Recent studies have explored the integration of lightweight backbones into YOLO-based detectors for various applications. Wang et al. [6] proposed a smoking behavior detection algorithm based on YOLOv8 with modified neck structures. Alkhammash [15] investigated multi-classification using YOLOv11 and hybrid YOLO11n-MobileNet models for fire detection. Su et al. [16] demonstrated the integration of ShuffleNet V2 with YOLOv5s for lightweight object detection in elevator monitoring systems. However, a comparative analysis of YOLOv11 integrated with lightweight backbones specifically addressing the critical trade-off between tiny object detection accuracy and inference latency has not been adequately explored. This study specifically evaluates the integration of lightweight backbones into the YOLOv11 framework to optimize efficiency. In this approach, the standard backbone (comprising C3k2 blocks) is replaced with MobileNetV2, GhostNet, or ShuffleNet V2, while the original Neck (PAN with C2PSA) and Head structures are retained to preserve YOLOv11's advanced feature fusion capabilities. This study aims to systematically evaluate the performance of YOLOv11n integrated with three lightweight backbone architectures, namely MobileNetV2, GhostNet, and ShuffleNet V2, for cigarette detection. The research objectives are: (1) to compare detection accuracy across different backbone configurations with varying width multipliers, (2) to analyze trade-offs between detection performance and computational efficiency, and (3) to provide deployment recommendations for smoke-free area monitoring systems operating under different resource constraints.

## 2.Methods

### *A. Dataset Description*

This study utilizes the Cigar Detection Dataset [16] sourced from Roboflow Universe. The dataset comprises 5,333 images annotated with bounding box coordinates for a single target class labeled "Cigarette". The dataset represents a compilation of diverse imagery collected from public sources, designed to cover a wide range of real-world appearances.

To address the need for generalization, the dataset includes various environmental contexts:

a. Environment: The images capture both indoor settings (e.g., rooms, offices) and outdoor scenarios, providing varied background complexities ranging from clean backgrounds to cluttered scenes.

b. Lighting and Quality: The dataset incorporates images with diverse lighting conditions, ranging from well-lit environments to shadow-dominated scenes. The image quality varies from high-resolution photography to lower-quality captures, simulating the varying visual quality often encountered in surveillance feeds.
c. Object Characteristics: Cigarette instances appear in multiple scales, from distinct close-ups to small objects occupying a minor fraction of the frame. Instances frequently exhibit partial occlusion by hands or facial features, presenting realistic detection challenges.

All images underwent preprocessing including auto-orientation correction and resizing to 640×640 pixels with black edge padding to maintain aspect ratio consistency. The dataset was initially partitioned into three subsets with a ratio of approximately 67:17:17. To enhance model robustness, data augmentation techniques were applied exclusively to the training subset. Given the computational constraints of the training environment, we employed a focused augmentation strategy, prioritizing only the most effective techniques for this specific domain. Horizontal flip and brightness adjustment were selected as the optimal combination to introduce necessary variability in viewpoint and illumination while maintaining training efficiency. This process expanded the training samples from 3,557 to 7,114, resulting in a final effective distribution approximating an 80:10:10 ratio across training, validation, and testing sets.

**Figure 1.** Representative samples showing detection challenges in real-world monitoring, including varying illumination, partial occlusions, and diverse background clutter.

### *B. Base Model Architecture*

The base detection framework employs YOLOv11n, the nano variant of the YOLOv11 architecture optimized for resource-constrained deployment scenarios. YOLOv11 represents the latest evolution in the YOLO family, introducing several architectural enhancements [10][11]. The architecture comprises three primary components: backbone for feature extraction, neck for multi-scale feature fusion, and head for bounding box prediction and classification.

The backbone utilizes C3k2 (Cross Stage Partial with 3×3 kernels, variant 2) blocks combining residual connections with partial channel processing for efficient gradient flow. The Spatial Pyramid Pooling Fast (SPPF) module aggregates multi-scale contextual information through pooling operations. Crucially, the backbone extracts hierarchical features at three varying scales: P3 (stride 8), P4 (stride 16), and P5 (stride 32), which serve as inputs for the fusion layers. The neck implements a Path Aggregation Network (PAN) structure with C2PSA (Cross Stage Partial with Position-Sensitive Attention) modules for enhanced feature representation. The anchor-free detection head generates predictions at three scales (80×80, 40×40, 20×20) to accommodate objects of varying sizes.

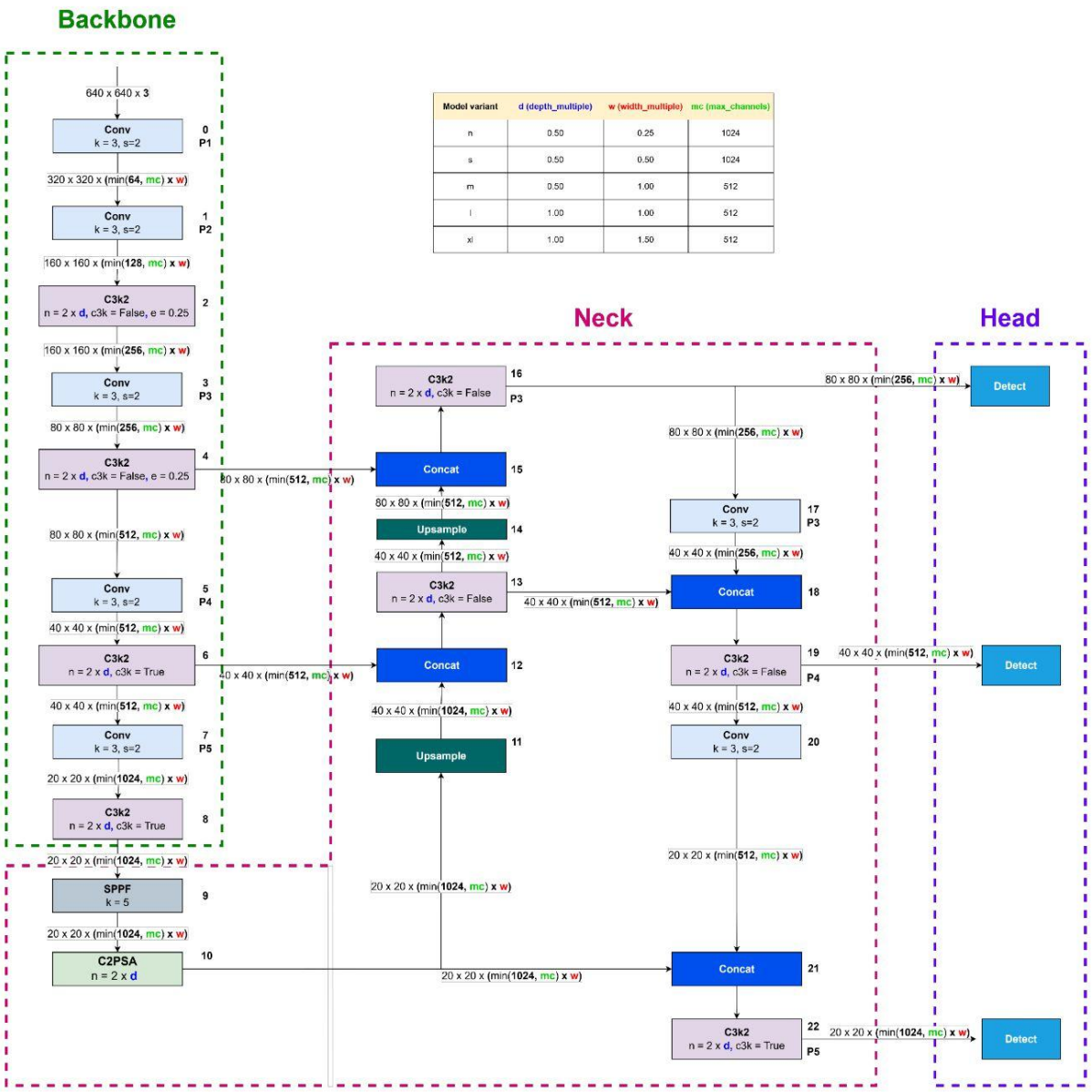


**Figure 2.** YOLOv11 architecture diagram illustrating the backbone, neck, and head components with configurable width multiplier parameters.

### *C. Lightweight Backbone Modifications*

Three lightweight backbone architectures were integrated into the YOLOv11n framework to replace the standard backbone. Each architecture was evaluated with width multipliers of 0.50, 0.75, and 1.0 to systematically control channel capacity and computational cost:

**MobileNetV2:** This architecture employs inverted residual blocks where the input is first expanded to a higher dimensional space, processed with depthwise convolutions, and then projected back to a lower dimensional representation using linear bottlenecks [12]. The expansion ratio of 6 enables rich feature learning while maintaining efficiency. Depthwise separable convolutions factorize standard convolutions into depthwise and pointwise operations, significantly reducing computational complexity compared to standard convolutions.

**GhostNet:** The Ghost module generates feature maps by applying linear transformations to a subset of intrinsic features [13]. Rather than computing all output channels through expensive convolutions, GhostNet generates a portion of features conventionally and derives the remaining "ghost" features through cheap linear operations. This approach effectively reduces

computational cost by utilizing redundancy in feature maps to generate additional features efficiently while preserving representation capability. The ratio between intrinsic and ghost features is configurable through the ghost ratio parameter.

**ShuffleNet V2:** Following practical efficiency guidelines derived from direct speed measurements on actual hardware, ShuffleNet V2 implements channel split operations that divide input features into two branches [14]. One branch maintains identity mapping while the other undergoes depthwise convolution and pointwise convolution. Channel shuffle operations after concatenation enable cross-group information exchange, addressing the limitation of grouped convolutions that restrict information flow between channel groups.

To ensure seamless integration with the YOLOv11 architecture, specific feature extraction layers from each lightweight backbone were mapped to the YOLOv11 Neck. Crucially, we extracted feature maps corresponding to strides of 8 (P3), 16 (P4), and 32 (P5) from each backbone to serve as inputs for the PAN feature fusion layers. This mapping ensures spatial dimension alignment with the original Neck components. The integration was implemented by modifying the model's YAML configuration to reference the specific output layers of the respective backbones while retaining the original SPPF module, PAN Neck, and Detection Head structure unchanged. This design isolates the backbone's impact on feature extraction efficiency while preserving the detector's multi-scale fusion capabilities.

### *D. Experimental Setup*

All experiments were conducted on Google Colab infrastructure equipped with NVIDIA Tesla T4 GPU (14.74 GB VRAM), Intel Xeon CPU @ 2.20 GHz (2 cores), and 12.67 GB RAM. The software environment comprised Python 3.12.12, PyTorch 2.9.0 with CUDA 12.6 and cuDNN 9.1.0, Ultralytics YOLO framework version 8.3.241, NumPy 2.0.2, and Pandas 2.2.2. Training hyperparameters were kept consistent across all model configurations to ensure fair comparison: 100 epochs, batch size of 15, SGD optimizer with default Ultralytics hyperparameters, initial learning rate of 0.01, and input resolution of 640×640 pixels. To ensure reproducibility and consistency of experimental results, a fixed random seed of 42 was applied, and CuDNN deterministic mode was enabled.

To strictly isolate the impact of architectural modifications from the benefits of transfer learning, the experimental design utilized two distinct comparison strategies:

1. **Architectural Impact Analysis (Scratch vs. Scratch):** This is the primary evaluation focus. All modified backbone models (MobileNetV2, GhostNet, ShuffleNet V2) were trained from scratch (using random initialization) with custom YAML configurations. To ensure a direct and unbiased comparison, a YOLOv11n Baseline (Scratch) was also trained without pre-trained weights. This methodology ensures that any observed performance difference is attributed solely to the architectural efficiency of the backbone, rather than prior knowledge transferred from external datasets.
2. **Performance Benchmark (Reference):** A YOLOv11n Baseline (Pretrained) initialized with COCO weights was trained to establish an upper-bound performance benchmark. This serves to quantify the performance gap between lightweight architectural optimization and the benefits of transfer learning.

### *E. Evaluation Metrics*

Detection accuracy was quantified using standard object detection metrics. Mean Average Precision at IoU threshold 0.5 (mAP@0.5) served as the primary metric, measuring the area under the precision-recall curve when considering predictions with IoU $\geq 0.5$ as true positives [17]. The more stringent mAP@0.5:0.95 was also calculated to average precision across IoU thresholds from 0.5 to 0.95 in 0.05 increments. Additionally, Precision represented the

proportion of correct positive predictions, while Recall indicated the proportion of actual positives correctly identified. The F1-score was computed as the harmonic mean of Precision and Recall [19] to provide a balanced assessment of the model's detection capability.

Efficiency was evaluated using both theoretical complexity and practical throughput metrics:

a. Theoretical Complexity: We utilized Parameter Count (Millions) to measure the model's spatial size and GFLOPs (Giga Floating Point Operations per second) to quantify the algorithmic computational cost independent of hardware.
b. Practical Throughput: We measured Model File Size (MB) for storage requirements and Inference Speed (FPS). It is important to note that FPS was recorded on the NVIDIA Tesla T4 GPU. While this high-performance environment differs from edge devices, the FPS metric serves as a relative benchmark to compare the throughput potential of different backbone architectures under identical hardware conditions.

## 3. Results and Discussion

### *A. Baseline Model Performance*

Table 1 presents comprehensive performance metrics for all evaluated model configurations. The pretrained YOLOv11n baseline shows the best performance, achieving mAP@0.5 of 0.8072, mAP@0.5:0.95 of 0.4635, Precision of 0.8688, and Recall of 0.7521. This configuration benefits from transfer learning, where COCO-pretrained weights provide strong feature representations that accelerate convergence and improve generalization to the cigarette detection domain. Comparison between pretrained and scratch-trained YOLOv11n reveals substantial performance differences. Training from scratch yields mAP@0.5 of 0.7320, representing a 7.52 percentage point degradation. This 9.3% relative accuracy reduction shows the importance of transfer learning when working with moderately-sized datasets. The pretrained model's superior Precision (0.8688 vs 0.8138) indicates more reliable positive predictions, reducing false alarm rates critical for practical deployment scenarios.

**Table 1**. Comprehensive Performance Comparison of YOLOv11n Backbone Variants

| Model | Params (M) | GFLOPs | *Size* (MB) | mAP@0.5 | mAP@ 0.5–95 | *Precision* | *Recall* | FPS |
|---|---|---|---|---|---|---|---|---|
| **YOLOv11n (*Pretrained*)** | 2.59 | 6.4 | 5.22 | **0.8072** | **0.4635** | **0.8688** | **0.7521** | **149.55** |
| **YOLOv11n (*Scratch*)** | 2.59 | 6.4 | 5.22 | 0.7320 | 0.3866 | 0.8138 | 0.6741 | 149.30 |
| **MobileNetV2 (1.0x)** | 3.98 | 9.4 | 8.01 | 0.7640 | 0.4014 | 0.8304 | 0.7173 | 95.71 |
| **MobileNetV2 (0.75x)** | 3.18 | 7.4 | 6.48 | 0.7305 | 0.3771 | 0.8183 | 0.6804 | 96.83 |
| **MobileNetV2 (0.5x)** | 2.59 | 5.9 | 5.35 | 0.7257 | 0.3796 | 0.8169 | 0.6614 | 95.88 |
| **GhostNet (1.0x)** | 5.65 | 8.8 | 11.31 | 0.7614 | 0.3973 | 0.8085 | 0.7131 | 58.41 |
| **GhostNet (0.75x)** | 4.12 | 7.1 | 8.38 | 0.7409 | 0.3893 | 0.8217 | 0.6878 | 57.68 |
| **GhostNet (0.5x)** | 3.00 | 5.7 | 6.24 | 0.7204 | 0.3691 | 0.8127 | 0.6688 | 58.21 |
| **ShuffleNet V2 (1.0x)** | 3.00 | 6.7 | 6.13 | 0.7501 | 0.3877 | 0.8237 | 0.6730 | 85.27 |
| **ShuffleNet V2 (0.75x)** | 2.64 | 5.9 | 5.44 | 0.7430 | 0.3956 | 0.8352 | 0.6629 | 86.41 |
| **ShuffleNet V2 (0.5x)** | **2.28** | **5.0** | **4.73** | 0.6818 | 0.3428 | 0.7872 | 0.6010 | 85.05 |

### *B. Lightweight Backbone Performance Analysis*

Among the lightweight backbone variants, MobileNetV2 (1.0x) achieves the highest detection accuracy with mAP@0.5 of 0.7640, followed closely by GhostNet (1.0x) at 0.7614 and ShuffleNet V2 (1.0x) at 0.7501. However, these accuracy rankings do not directly correspond to efficiency characteristics, necessitating multi-dimensional trade-off analysis. MobileNetV2 configurations exhibit consistent but moderate efficiency gains across width multipliers. The 1.0x variant requires 3.98M parameters (53.7% more than baseline), while the 0.5x variant matches the baseline with 2.59M parameters but at reduced accuracy. Notably, MobileNetV2 demonstrates slower inference speeds (95-97 FPS) compared to baseline (149 FPS), attributed to the architectural complexity that, while theoretically efficient in terms of parameters, results in higher inference latency on the test hardware compared to the baseline.

GhostNet configurations demonstrate the largest parameter counts among evaluated architectures, with the 1.0x variant reaching 5.65M parameters and 11.31 MB model size, more than double the baseline. This unexpected result is caused by the additional layers required to implement Ghost modules while maintaining feature map dimensions compatible with the YOLO neck and head structures. Despite theoretical computational savings, GhostNet variants achieve the slowest inference speeds (57-58 FPS), likely due to the architectural characteristics that may not be fully optimized for the specific GPU hardware used in this study. ShuffleNet V2 emerges as the most efficient backbone option across multiple dimensions. The 0.5x variant achieves the smallest footprint at 2.28M parameters and 4.73 MB, representing 12% reduction in parameters and 9.4% reduction in model size compared to baseline. Even the 1.0x variant maintains competitive parameter count (3.00M) while delivering reasonable inference speed (85 FPS). The ShuffleNet V2 architecture demonstrates higher compatibility with the GPU inference process used in this study compared to the other lightweight variants.

A notable finding across all lightweight variants is the non-linear relationship between computational complexity reduction and inference speed improvement. Despite significant GFLOPs reductions (up to 22% lower than baseline), the lightweight backbones consistently achieve slower FPS on the Tesla T4 GPU. This indicates that on high-end GPU hardware, performance bottlenecks are not solely determined by the number of operations (GFLOPs) but are significantly influenced by architectural characteristics. Specifically, operations like depthwise separable convolutions in lightweight backbones reduce FLOPs but often incur higher Memory Access Costs (MAC), which are less efficiently parallelized on high-throughput GPUs like the Tesla T4 compared to standard dense convolutions. Consequently, these lightweight models might show more significant speed benefits on resource-constrained edge devices, for which they are primarily designed, rather than on high-performance server GPUs.

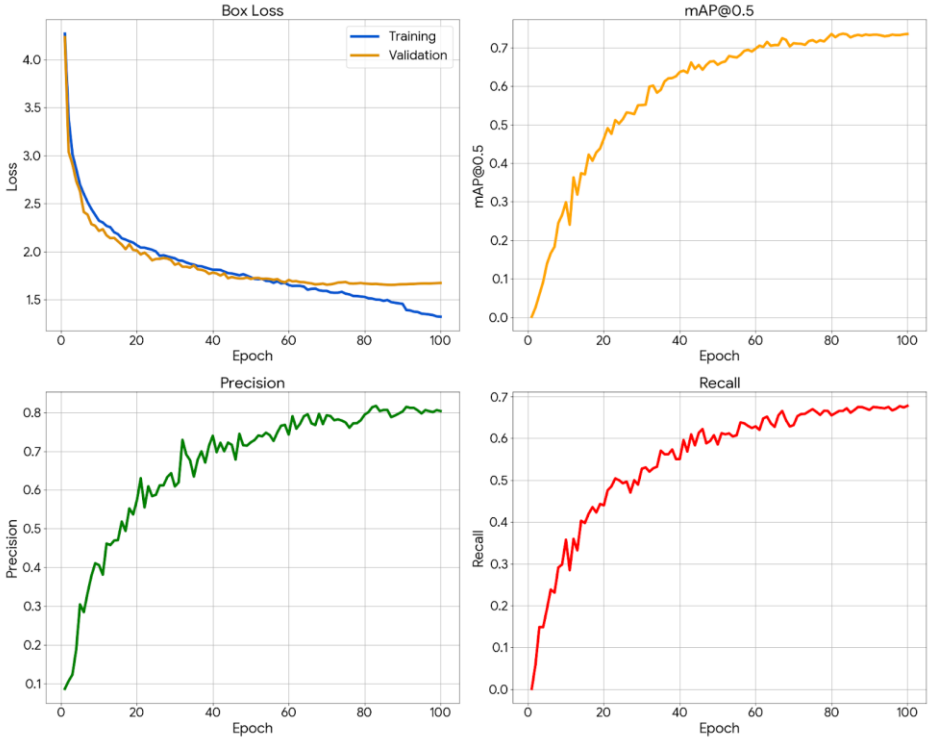


**Figure 3.** Training curves showing Box Loss convergence, mAP@0.5 progression, Precision evolution, and Recall development across 100 epochs for the baseline YOLOv11n model.

### *C. Width Multiplier Sensitivity Analysis*

Width multiplier analysis reveals different results across backbone architectures, with significant implications for deployment configuration selection. Table 2 summarizes the accuracy degradation when reducing width multipliers from 1.0x to lower values. Figure 4 visualizes the trade-off between detection accuracy and computational complexity across all configurations.

ShuffleNet V2 demonstrates good stability when reducing width multipliers, with the 0.75x variant experiencing only 0.95% mAP@0.5 degradation (from 0.7501 to 0.7430) while achieving 12% GFLOPs reduction (from 6.7 to 5.9). This favorable accuracy-efficiency trade-off suggests that the ShuffleNet V2 architecture remains effective for this specific task even with reduced channel capacity. The 0.5x variant shows more substantial degradation (9.1%, mAP@0.5: 0.6818) but remains viable for extremely resource-constrained scenarios.

MobileNetV2 exhibits steeper accuracy gradients with width reduction. The 0.75x variant shows 4.4% degradation compared to 1.0x, while 0.5x shows 5.0% degradation. However, the absolute accuracy difference between 0.75x and 0.5x is minimal (0.7305 vs 0.7257), suggesting less significant improvement from further width reduction below 0.75x. GhostNet shows intermediate sensitivity, with 2.7% degradation at 0.75x and 5.4% at 0.5x. The relatively lower baseline accuracy of GhostNet combined with its larger model sizes makes it less attractive for lightweight deployment compared to ShuffleNet V2. Figure 5 presents the inference speed comparison, demonstrating that all models exceed the 30 FPS real-time threshold despite the varying computational characteristics.

**Table 2**. Accuracy Degradation with Width Multiplier Reduction

| Backbone | 1.0x | 0.75x | Δ75% | 0.5x | Δ50% |
|---|---|---|---|---|---|
| MobileNetV2 | 0.7640 | 0.7305 | -4.4% | 0.7257 | -5.0% |
| GhostNet | 0.7614 | 0.7409 | -2.7% | 0.7204 | -5.4% |
| ShuffleNet V2 | 0.7501 | 0.7430 | **-0.9%** | 0.6818 | -9.1% |

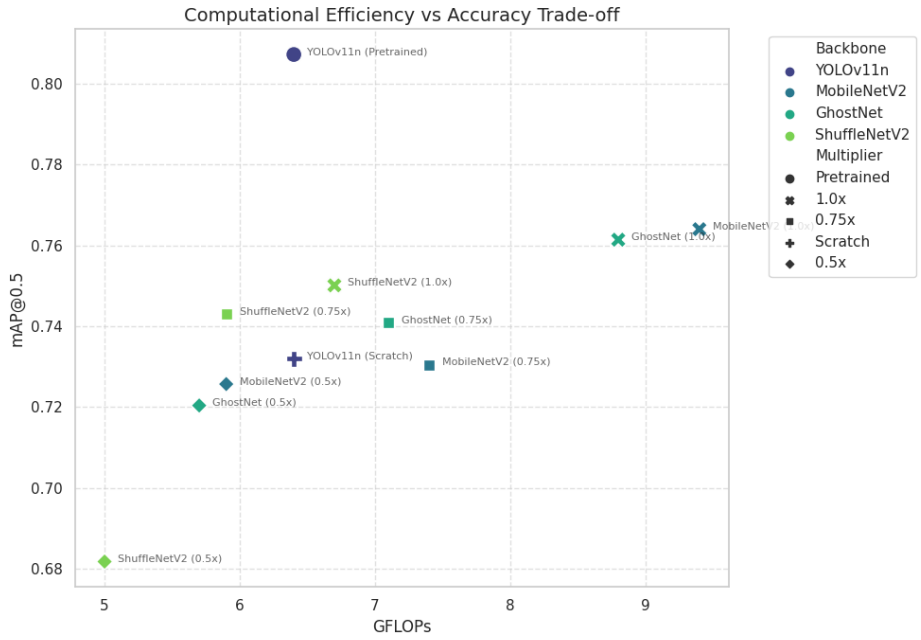


**Figure 4.** Trade-off visualization between detection accuracy (mAP@0.5) and computational complexity (GFLOPs) for all model configurations. The pretrained YOLOv11n baseline achieves optimal accuracy at moderate complexity, while ShuffleNet V2 variants offer the best efficiency-accuracy balance among lightweight backbones.

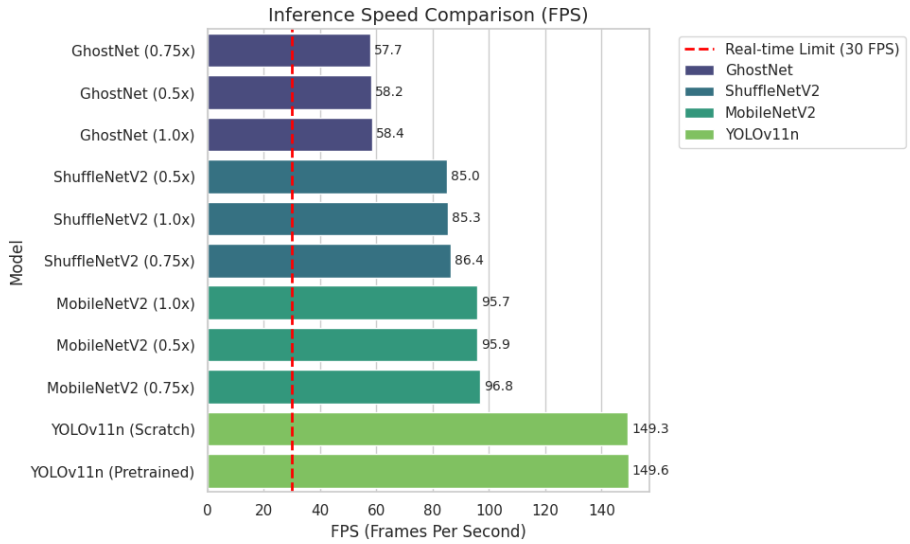


**Figure 5.** Inference speed comparison across all model configurations. The dashed red line indicates the 30 FPS real-time threshold. All models exceed real-time requirements, with the baseline YOLOv11n achieving highest throughput (149 FPS) followed by MobileNetV2 variants (95-97 FPS).

### *D. Analysis of Detection Cases*

Figure 6 shows sample detection results from the ShuffleNet V2 (0.75x) model on test images showing various smoking scenarios. The model successfully detects cigarettes held at different angles and positions in (a) and (b). However, as shown in (c), the model failed to detect the object, likely due to the background complexity or the specific pose. The confidence scores (0.35 to 0.70) correspond to the object visibility, with higher confidence for clearly visible instances and lower confidence for partially occluded or distant objects.


**Figure 6.** Detection results from ShuffleNet V2 (0.75x) model. (a) High confidence detection (0.70), (b) Medium confidence detection (0.35), (c) Missed detection case.

This visualization aligns with the quantitative metrics in Table 1 where Precision is consistently higher than Recall across all evaluated models. For instance, the ShuffleNet V2 0.75x variant achieves a Precision of 0.83 compared to a Recall of 0.66. This significant gap explains why the model rarely produces false alarms in clear scenes such as (a). However, the lower Recall indicates a systematic limitation in detecting 'hard samples'. The failure in (c) specifically highlights the model's dependency on facial context; when the cigarette is held away from the mouth in a hand-held pose against a noisy background, the lightweight feature extractor struggles to distinguish the object from the clutter.

### *E. Deployment Recommendations*

Based on the quantitative trade-off analysis of computational metrics (GFLOPs, Parameters) and detection accuracy, three deployment scenarios are proposed. While direct validation on embedded hardware remains a future objective, the following configurations are recommended based on their theoretical efficiency profiles:

**Scenario A - Maximum Accuracy Priority:** For applications where detection reliability is crucial and computational resources are not severely constrained (e.g., server-based processing, high-end surveillance systems), the pretrained YOLOv11n baseline is recommended. Its mAP@0.5 of 0.8072 and Precision of 0.8688 minimize false negatives and false positives critical for effective enforcement.

**Scenario B - Balanced Performance:** For edge devices with moderate computational capabilities (e.g., embedded AI boards), ShuffleNet V2 (0.75x) provides optimal trade-off. With mAP@0.5 of 0.7430, 2.64M parameters, 5.44 MB size, and 86 FPS, it delivers acceptable detection accuracy while fitting within the theoretical memory and processing constraints of typical embedded systems [7].

**Scenario C - Ultra-Lightweight Deployment:** For severely constrained environments or scenarios requiring minimal power consumption, ShuffleNet V2 (0.5x) offers the smallest footprint at 4.73 MB and 2.28M parameters. While mAP@0.5 of 0.6818 is lower, it provides a viable solution for standalone monitoring on extreme edge devices where efficiency is the primary constraint [18].

## 4. Conclusions

This study systematically evaluated the performance of YOLOv11n by substituting its standard backbone with three lightweight architectures, namely MobileNetV2, GhostNet, and ShuffleNet V2, to address the trade-off between accuracy and efficiency in cigarette detection. Based on the experimental results, ShuffleNet V2 (0.75x) is identified as the optimal design choice for edge deployment. It achieves a balanced performance with an mAP@0.5 of 0.7430 and a low computational cost of 5.9 GFLOPs, offering a practical solution for resource-constrained monitoring systems where model size and energy efficiency are critical.

While the pretrained YOLOv11n baseline remains the most accurate model (mAP@0.5 0.8072), its reliance on standard convolutions results in higher theoretical computational costs. Conversely, the integration of GhostNet proved ineffective for this specific architecture, resulting in increased model size and latency on the tested GPU hardware. The study also observes that theoretical reductions in GFLOPs do not automatically result in faster inference on server-grade GPUs (Tesla T4). This discrepancy highlights that lightweight architectures are specifically optimized for constrained CPUs or NPUs, and their efficiency gains are not fully realized on high-throughput parallel hardware.

Therefore, the selection of backbone presents a clear design decision: for applications prioritizing maximum detection rates regardless of hardware cost, the standard YOLOv11n is superior due to its higher Recall. However, for battery-powered or memory-constrained edge devices, ShuffleNet V2 (0.75x) is the recommended configuration, offering a critical 12% reduction in computational complexity (GFLOPs) with less than 1% loss in precision compared to the 1.0x variant.

Future Work: The primary limitation of this study is the lack of validation on physical embedded hardware, such as Raspberry Pi or Jetson Nano. Future research must prioritize benchmarking inference speeds on these specific platforms to validate the efficiency gains in real-world edge scenarios. Additionally, the dataset should be expanded to include multi-class scenarios, such as distinguishing between vapes and conventional cigarettes, to address broader monitoring requirements.

## References


[1] World Health Organization, “Tobacco.” [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/tobacco

[2] Institute for Health Metrics and Evaluation (IHME), “Global Burden of Disease (GBD) Results.” [Online]. Available: https://vizhub.healthdata.org/gbd-results/

[3] Peraturan Pemerintah RI, “Peraturan Pemerintah (PP) Nomor 109 Tahun 2012 tentang Pengamanan Bahan Yang Mengandung Zat Adiktif Berupa Produk Tembakau Bagi Kesehatan,” 2012, *Jakarta*. [Online]. Available: https://peraturan.bpk.go.id/Details/5324/pp-no-109-tahun-2012

[4] I. G. Y. E. Pramana Putra and Y. Setyowati, “Implementasi Kebijakan Kawasan Tanpa Rokok Di Lingkungan Instansi Pemerintah Kota Yogyakarta,” *TheJournalish: Social and Government*, vol. 3, no. 1, pp. 17–27, Jan. 2022, doi: 10.55314/tsg.v3i1.223.

[5] Badan Pusat Statistik Indonesia, “Persentase Penduduk Berumur 15 Tahun ke Atas yang Merokok Tembakau selama Sebulan Terakhir Menurut Provinsi.” [Online]. Available: https://www.bps.go.id/id/statistics-table/2/MTQzNSMy/persentase-penduduk-berumur-15-tahun-ke-atas-yang-merokok-tembakau-selama-sebulan-terakhir-menurut-provinsi.html

[6] Z. Wang, L. Lei, and P. Shi, “Smoking behavior detection algorithm based on YOLOv8-MNC,” *Front Comput Neurosci*, vol. 17, Aug. 2023, doi: 10.3389/fncom.2023.1243779.

[7] M. F. Tariq and M. A. Javed, “Small Object Detection with YOLO: A Performance Analysis Across Model Versions and Hardware,” Apr. 2025.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[9] S. Tasnim and W. Qi, “Progress in Object Detection: An In-Depth Analysis of Methods and Use Cases,” *European Journal of Electrical Engineering and Computer Science*, vol. 7, no. 4, pp. 39–45, Jul. 2023, doi: 10.24018/ejece.2023.7.4.537.

[10] P. Hidayatullah, N. Syakrani, M. R. Sholahuddin, T. Gelar, and R. Tubagus, “YOLOv8 to YOLO11: A Comprehensive Architecture In-depth Comparative Review,” Jan. 2025.

[11] R. Khanam and M. Hussain, “YOLOv11: An Overview of the Key Architectural Enhancements,” Oct. 2024, doi: 10.48550/arXiv.2410.17725.

[12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2018, pp. 4510–4520. doi: 10.1109/CVPR.2018.00474.

[13] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More Features From Cheap Operations," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2020, pp. 1577–1586. doi: 10.1109/CVPR42600.2020.00165.

[14] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design," Jul. 2018.

[15] E. H. Alkhammash, "Multi-Classification Using YOLOv11 and Hybrid YOLO11n-MobileNet Models: A Fire Classes Case Study," *Fire*, vol. 8, no. 1, p. 17, Jan. 2025, doi: 10.3390/fire8010017.

[16] J. Su, M. Yang, and X. Tang, "Integration of ShuffleNet V2 and YOLOv5s Networks for a Lightweight Object Detection Model of Electric Bikes within Elevators," *Electronics (Basel)*, vol. 13, no. 2, p. 394, Jan. 2024, doi: 10.3390/electronics13020394.

[17] M. A. Hameed and Z. A. Khalaf, "A survey study in Object Detection: A Comprehensive Analysis of Traditional and State-of-the-Art Approaches," *Basrah Researches Sciences*, vol. 50, no. 1, p. 16, Jun. 2024, doi: 10.56714/bjrs.50.1.5.

[18] T. Shahriar, "Comparative Analysis of Lightweight Deep Learning Models for Memory-Constrained Devices," May 2025.

[19] Tedyyana, A., Ahmad, A. A., Idrus, M. R., Mohd Shabli, A. H., Abu Seman, M. A., Ghazali, O., & Abd Razak, A. H. (2024). Enhance Telecommunication Security Through the Integration of Support Vector Machines. *International Journal of Advanced Computer Science & Applications*, *15*(3).