Real-Time Optimization and Lightweight Architecture of Face Detection Models for Embedded Systems

Min Ki Son
Convergence Security for Automobile
Soonchunhyang University
Asan, Republic of Korea
minki133@sch.ac.kr

Seungkeun Park
Smart Automotive Engineering Department
Soonchunhyang University
Asan, Republic of Korea
keiny@sch.ac.kr

***Abstract*—This study presents optimization and lightweighting techniques to improve the performance of real-time face detection models in resource-constrained environments such as embedded systems. The selected model, RetinaFace, based on the MobileNetV1 0.25 backbone, was streamlined into SlimLite and RFBLite networks, and the components and layers of the model were efficiently restructured to effectively reduce model size and inference time. These optimizations achieved a balanced trade-off between performance and real-time processing capabilities in a resource-constrained environment. The ONNX optimization further improved the inference speed of the RetinaFace model by approximately 37.7%, reducing the average inference time from 524.89 ms to 327.24 ms on an embedded platform (Raspberry Pi 4). In addition, resource efficiency was maximized by reducing the input resolution to 320x240, resulting in an additional 67.5% reduction in inference time to 106.16 ms while maintaining face detection accuracy. This demonstrates that the model retains its detection capabilities even at reduced input resolution, ensuring reliable performance. Accuracy evaluations were performed using the WIDER FACE dataset, highlighting the importance of tailoring optimization strategies to achieve optimal performance in constrained environments. This research is expected to broaden the applicability of lightweight face detection models, enabling potential integration in areas such as software-defined vehicles (SDVs), multi-object tracking, gesture recognition, and advanced driver assistance systems (ADAS).**

Keywords—face detection, embedded systems, real-time processing, model optimization, lightweight network

I. Introduction

Computer vision technology plays an important role in the perception system of Autonomous Vehicles (AVs), focusing on the recognition of the external environment. This includes tasks such as detecting road signs, identifying pedestrians, and predicting vehicle trajectories. However, interest in in-vehicle perception technology has recently increased with the emergence of the concept of Software-Defined Vehicles (SDVs) [1], which integrate hardware and software to allow flexible deployment of functionality and software-based updates. In-vehicle perception technology can be used in a variety of areas, including monitoring driver status, detecting objects in the vehicle, and analyzing passenger behavior. Driver monitoring systems are an important example of this technology, as they are essential for detecting and responding to potentially dangerous situations such as driver inattention or drowsiness. This requires technology that can accurately recognize faces in real time and accurately estimate landmarks such as the eyes, nose, and mouth.

In the confined environment of a vehicle, achieving high-speed processing and accuracy is essential to effectively perform face detection and landmark estimation. However, the space and power-constrained nature of the in-vehicle environment limits the feasibility of using high-performance GPUs or large servers, necessitating lightweight and optimized models capable of real-time processing on compact embedded devices.

To meet these requirements, this research focuses on optimizing real-time face detection performance while maintaining accuracy in resource-constrained environments. The RetinaFace model [2], known for its high accuracy in face detection, was chosen as the basis for exploring performance optimization and lightweighting techniques for embedded vehicle applications. While the original RetinaFace model uses MobileNetV1 0.25 [3] as a backbone for partial lightweighting, further structural simplifications and layer adjustments are needed to achieve reduced model size and inference time without sacrificing detection accuracy.

The paper is organized as follows: Section II reviews related work, Section III outlines the proposed lightweighting and optimization methods, Section IV describes the experimental setup and results, and Section V discusses the conclusions and future research directions.

II. Related Work

A. Face Detection

Face detection is the task of automatically identifying the location of a face in an image or video frame, while landmark estimation predicts the positions of key facial features on the detected face. Recent advances in deep learning, particularly Convolutional Neural Networks (CNN) [4] and other neural network-based models, have achieved impressive results in these tasks. A significant area of research has focused on developing lightweight and optimized models, especially for real-time performance in resource-constrained environments.

BlazeFace, introduced by Bazarevsky et al. (2019) [5], is a lightweight face detection model developed by Google that is specifically optimized for real-time processing on resource-constrained devices such as mobile platforms. By using Depthwise Separable Convolutions [6], BlazeFace reduces the computational burden; however, its simplified structure may degrade performance when detecting small faces or faces at different angles in complex environments.

SCRFD (Sample and Computation Redistribution Face Detector), introduced by Guo et al. (20-21) [7], improves face detection performance by employing sample and computation redistribution strategies to achieve high performance in various environments. However, the complex structure of SCRFD, which requires multiple optimization techniques, can lead to increased computational cost, potentially limiting its application in highly resource-constrained environments.

While current face detection models are designed for lightweight, real-time processing, each has its own limitations. To address these limitations, RetinaFace was developed by

combining multi-task learning [8] with Feature Pyramid Network (FPN) [9] and Single Stage Headless (SSH) [10] modules to achieve accurate detection of faces of different sizes and angles. This configuration enables real-time performance even in resource-constrained environments.

B. Landmark Estimation

Landmark estimation, which involves accurately predicting the positions of key facial features (e.g., eyes, nose, mouth), plays a critical role in applications such as face detection, expression analysis, and emotion recognition. Various methods have been proposed to improve the efficiency and accuracy in this area.

Multi-Task Cascaded Convolutional Networks (MTCNN), proposed by Zhang et al. (2016) [11], performs face detection and landmark estimation simultaneously. MTCNN uses a three-stage neural network (proposal network, refinement network, output network), which progressively refines the locations of the face and its landmarks. Although it offers high accuracy and fast processing speed, MTCNN's large number of parameters and high computational complexity can hinder performance in resource-constrained environments.

3D Dense Face Alignment model proposed by Guo et al. (2020) [12] addresses the limitations of 2D landmark estimation by exploiting 3D facial information, which enables accurate landmark positioning despite rotations or tilts. However, incorporating 3D information adds computational cost, making it less suitable for resource-constrained environments such as embedded systems.

III. Proposed Method

To improve the real-time processing efficiency of the RetinaFace model in resource-constrained environments, this study presents three key approaches: (1) Modifying the MobileNetV1 backbone to incorporate the SlimLite and RFBLite networks, (2) Reconfiguring model layers and components for greater efficiency, and (3) Applying advanced lightweighting techniques, including ONNX conversion, to maximize model efficiency. These methods aim to significantly reduce model size and computational load while maintaining detection accuracy, thereby facilitating real-time performance on constrained hardware.

A. Modification of the MobileNetV1 Backbone

Current RetinaFace model integrates Feature Pyramid Network (FPN) and Single Stage Headless (SSH) modules to support multi-resolution feature maps. Building on the lightweight design of MobileNetV1 with Depthwise Separable Convolutions, we introduce two streamlined network structures: SlimLite and RFBLite networks.

SlimLite Network

SlimLite is optimized for real-time processing in constrained environments using MobileNetV1-based lightweighting techniques. To simplify the model, the FPN and SSH modules are removed and replaced by three streamlined convolutional blocks:

- Conv-BN Activation Block (conv_bn): This block, consisting of a 3x3 convolution, batch normalization (BN) [13], and ReLU activation [14], stabilizes and improves the initial feature extraction.
- Depthwise Convolution Block (conv_dw): Uses Depthwise Separable Convolution, applying a 3x3 filter to each input channel independently, followed by a 1x1 convolution for channel merging, increasing speed and reducing computational load.
- Depthwise Separable 2D Convolution (depth_conv2d): Efficient grouped convolutions optimize feature information while maintaining a lightweight structure.

Unlike the original RetinaFace model, which combines multi-resolution feature maps, the SlimLite Network relies on a single-scale feature map for prediction. Sequential conv1 through conv4 blocks extract basic features, while conv_dw blocks from conv5 through conv11 generate deeper features for a single-scale map. The conv12 block refines features for bounding box regression, classification, and landmark regression, with a multi-box configuration defining layers for each task, enabling efficient prediction with minimal resources. By eliminating complex feature combination processes, SlimLite Network reduces model complexity, increasing processing speed and resource efficiency. Fig. 1 illustrates this optimized structure, effectively demonstrating the achievement of model lightweighting under constrained settings.

RFBLite Network

RFBLite Network is designed to enhance the real-time processing performance of the RetinaFace model in resource-constrained environments. Like SlimLite Network, RFBLite is optimized for efficient operation on limited hardware, but it uniquely incorporates the Receptive Field Block (RFB) module [15] to enable accurate face detection over a range of sizes without complex multi-scale feature combinations. This network consists of the following core components:

- Receptive Field Block (RFB) Module: Replacing the SSH module, the RFB module contains three branches that capture features with distinct receptive fields, enabling effective face detection across multiple sizes without multi-scale processing.
- Single module for multi-scale feature extraction: As a single multi-scale feature extraction module, the BasicRFB module captures features of different sizes in parallel, streamlining the network, reducing memory usage, and maximizing computational efficiency. Each branch (branch0, branch1, branch2) uses different filter sizes and dilation rates to optimize feature extraction.

Structurally, RFBLite is similar to SlimLite in that the initial convolution layers (conv1 to conv4) extract basic features, while the blocks from conv5 to conv7 and the BasicRFB module (conv8) focus on extracting intermediate and high-level features, resulting in a single-scale feature map. The final prediction layers (conv9 to conv14) refine features for bounding box regression, classification, and landmark regression, with a multi-box configuration for efficient task-specific outputs. As shown in Fig. 2, this architecture provides a lightweight model that maximizes real-time processing performance in resource-constrained environments.

B. RetinaFace Base Model Modification

To further improve efficiency and real-time performance in resource-constrained environments, the RetinaFace model has been redesigned to focus on lightweighting key

components. The original RetinaFace model uses FPN and SSH modules to combine multi-resolution feature maps for face detection of different sizes. Specific lightweighting strategies were applied to optimize these modules for embedded system performance.

Lightweight FPN Module

To reduce the FPN module's computation cost and memory consumption, the dimensions of the input and output channels have been reduced. Specifically, the original input channels were reduced from [64, 128, 256] to [32, 64, 128], and the output channel was reduced from 64 to 32, thereby reducing the computational requirements for each layer. This optimization streamlined the input-output configurations for both output layers (output1, output2, output3) and merge layers (merge1, merge2) to [32 → 32, 64 → 32, 128 → 32] and [32 → 32], respectively.

SSH Module Simplification

Further reductions in computation cost and memory usage have been achieved in the SSH module by reducing the channel dimensions. The input channel size has been reduced from 32 to 16, and the output channel size has been reduced from 64 to 32, effectively reducing the computational load while maintaining accuracy. These adjustments significantly improve real-time processing capability without compromising model performance.

Together, these lightweighting modifications increase the efficiency of the FPN and SSH modules, enabling the redesigned RetinaFace model to achieve robust real-time face detection and analysis in highly resource-constrained environments, highlighting the benefits of a streamlined, lightweight network structure.

C. *Advanced Optimization Techniques for Model Efficiency*

To further improve model efficiency and maintain real-time performance in resource-constrained environments, ONNX conversion was applied. This conversion strategy streamlines the model structure, reduces computational load and memory consumption, and extends its applicability to a variety of practical environments.

ONNX Conversion [16]

ONNX is an open standard that enhances model portability and enables compatibility between different deep learning frameworks and hardware platforms. In this study, the RetinaFace model was converted to the ONNX format to streamline the model structure, remove unnecessary computations during inference, and thereby increase the processing speed. This conversion significantly improves the applicability and usability of the model in a variety of practical environments.

IV. EXPERIMENTS

A. *Dataset*

Model performance was evaluated using the WIDER FACE dataset [17], a large-scale dataset designed to reflect different conditions in face detection tasks. The dataset contains 32,203 images and 393,703 face bounding boxes, and is characterized by high variability in face size, pose, expression, occlusion, and illumination. It is divided into three difficulty levels—Easy, Medium, and Hard—based on Edge Box detection rates, which categorize samples from clearly visible faces (Easy) to complex, occluded faces

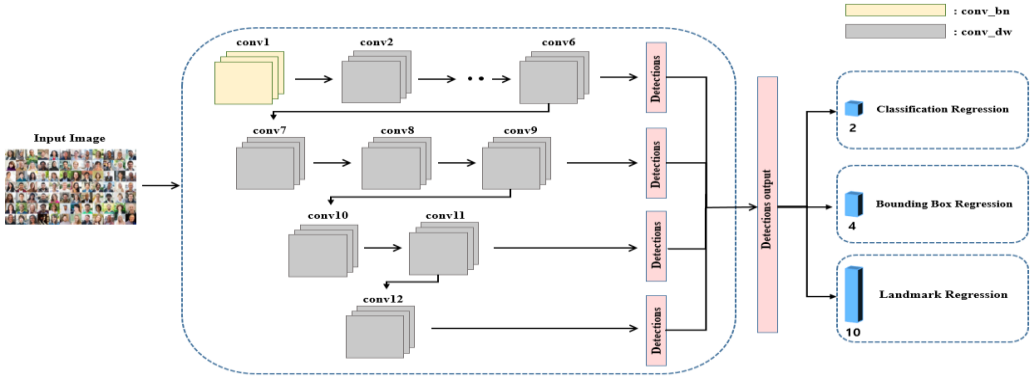


Figure 1. SlimLite Network

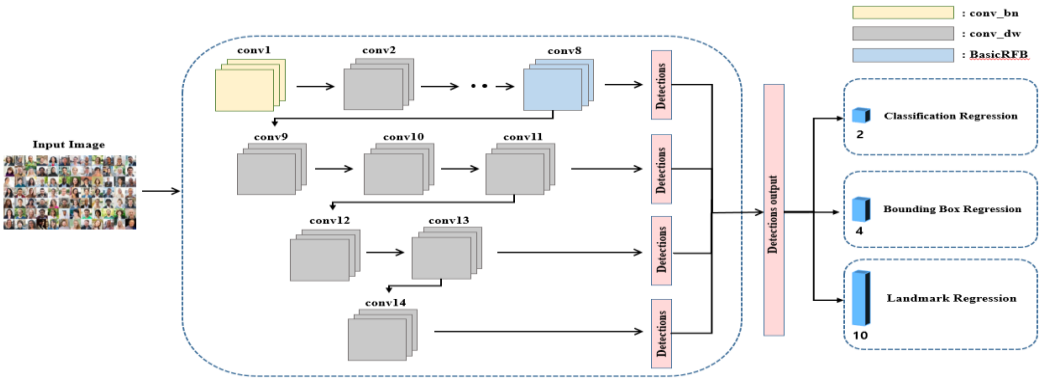


Figure 2. RFBLite Network

(Hard), allowing for a progressively stringent evaluation of model performance.

The evaluation used 3,287 validation images from the WIDER FACE dataset. Model performance was measured using Average Precision (AP) at each difficulty level, a metric that balances precision and recall, providing a critical measure of the proposed lightweighting and optimization strategies under varying conditions.

B. Model Parameters

To evaluate the effectiveness of the lightweighting approaches described in Section III, the number of parameters and the computational load (FLOPs) of each model were analyzed. Compared to the base MobileNetV1 0.25 model, the modified MobileNetV1 0.25 model achieved a reduction of approximately 73%, while the SlimLite and RFBLite networks showed reductions of approximately 77% and 58%, respectively. These results highlight the impact of the lightweighting strategies in reducing model complexity and resource consumption. Detailed figures are shown in Table I.

TABLE I. PARAMETERS AND FLOPS OF DIFFERENT MODELS

Methods	Parameters (K)	FLOPs (M)
MobileNetV1 0.25 (base)	426.608	193.870
MobileNetV1 0.25 (modified)	115.216	68.606
RFBLite	136.54	81.18
SlimLite	**110.58**	**49.09**

C. Face Detection Accuracy

Face detection accuracy was evaluated using the WIDER FACE validation dataset, with performance summarized in Table II and Fig. 3. The base MobileNetV1 0.25 model achieved the highest AP score of 0.811 on the Easy level, and also performed well on the Medium and Hard levels, with scores of 0.697 and 0.376, respectively.

SlimLite Network, optimized for reduced computational load, had lower AP scores of 0.749, 0.611, and 0.291 for Easy, Medium, and Hard levels, respectively, reflecting a trade-off between computational efficiency and detection performance. These results indicate that SlimLite Network maintains reasonable detection accuracy while supporting real-time processing in resource-constrained environments.

TABLE II. PERFORMANCE COMPARISON OF DIFFERENT MODELS ON THE WIDER FACE VALIDATION DATASET (AP SCORES)

Methods	EASY	Medium	Hard
MobileNetV1 0.25 (base)	**0.811**	**0.697**	**0.376**
MobileNetV1 0.25 (modified)	0.769	0.658	0.331
RFBLite	0.793	0.677	0.337
SlimLite	0.749	0.611	0.291

D. Real-Time Inference

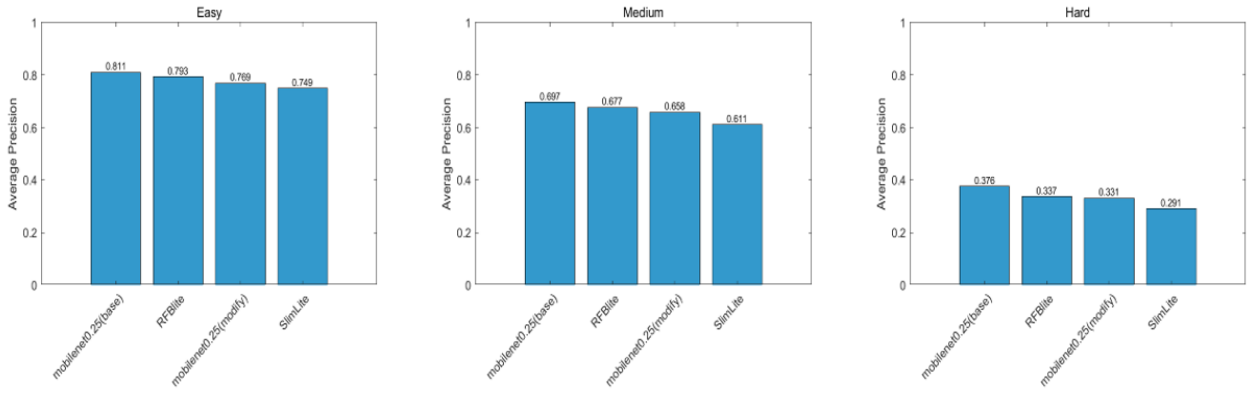
Real-time inference was tested in two environments: a standard PC with an Intel(R) Core(TM) i7-14700F CPU (2.10 GHz), 32GB RAM, and a 64-bit operating system, and a Raspberry Pi 4 Model B with 4GB RAM. In the PC environment, the models were tested in a CPU-only setting to evaluate processing efficiency. The Raspberry Pi 4, which represents a resource-constrained environment similar to automotive embedded systems, was chosen to evaluate the models' ability to meet real-time requirements in a constrained hardware setup. This setup is consistent with the requirements of Software-Defined Vehicles (SDVs), and highlights the need for efficient face detection models optimized for real-time performance in resource-constrained environments.

i7-14700F CPU Environment

In the i7-14700F CPU environment, the MobileNetV1 0.25 modified model demonstrated approximately 21.9% faster inference performance than the MobileNetV1 0.25 base model (Table III), with an average inference time of 22.69 ms compared to 29.06 ms for the base model. Although lightweighting effectively reduced parameters, the SlimLite and RFBLite networks had slightly longer inference times of 28.35 ms and 24.39 ms, respectively, due to architectural adjustments or unoptimized operations that affected performance.

Raspberry Pi 4 Environment

The results of the real-time inference test on Raspberry Pi 4 are shown in Table IV. SlimLite Network achieved the fastest performance with an average inference time of 2472.33 ms, while the modified MobileNetV1 0.25 model took 26.9% longer at 3137.07 ms. This suggests that the SlimLite Network may be more suitable for environments with limited resources.



(a) Easy Level (b) Medium Level (c) Hard Level

Figure 3. Precision-Recall (PR) curves of different models on the WIDER FACE Validation Dataset for Easy, Medium, and Hard Levels

TABLE III. INFERENCE TIMES (MS) OF DIFFERENT MODELS ON I7-14700F CPU ENVIRONMENT

Methods	Min Time(ms)	Max Time(ms)	Avg Time(ms)
bileNetV1 0.25 (base)	20.49	63.08	29.06
MobileNetV1 0.25 (modified)	**18.02**	40.83	**22.68**
RFBLite	20.69	**37.69**	24.39
SlimLite	26.76	49.24	28.36

TABLE IV. INFERENCE TIMES (MS) OF DIFFERENT MODELS ON RASPBERRY PI 4 ENVIRONMENT

Methods	Min Time(ms)	Max Time(ms)	Avg Time(ms)
MobileNetV1 0.25 (base)	3366.90	4260.90	3876.98
MobileNetV1 0.25 (modified)	**2346.38**	3716.65	**3137.07**
RFBLite	2367.59	**2939.28**	2725.15
SlimLite	2299.42	2882.68	2472.33

In summary, while the MobileNetV1 0.25 (modified) model achieved approximately 21.9% improvement in the i7-14700F CPU environment, the SlimLite Network showed approximately 36.2% faster inference time on Raspberry Pi 4. These results suggest that each model performs optimally depending on specific resource conditions and hardware capabilities, highlighting the importance of selecting a model tailored to application environments.

ONNX Real-Time Inference

Previous performance evaluations suggested that additional optimization was needed to maintain real-time processing performance for the proposed models under various conditions. To address this need, Open Neural Network Exchange (ONNX) optimization techniques were applied to improve model inference speed and reduce latency.

Specifically, the ONNX optimization was validated in the Raspberry Pi 4 environment, and the performance was further evaluated at a resolution of 320 x 240, taking into account the system memory and computational capacity (Table V and Fig. 5).

ONNX-optimized MobileNetV1 0.25 modified model achieved an average inference time of 327.24 ms, which represents a significant performance improvement over the pre-optimization state. The ONNX-optimized MobileNetV1 0.25 base and RFBLite models achieved average inference times of 524.89 ms and 432.05 ms, respectively. Notably, reducing the resolution of the MobileNetV1 0.25 modified to 320x240, as shown in Figure 4, achieved the fastest average inference time of 106.16 ms.

This demonstrates that ONNX conversion and resolution tuning can significantly improve real-time performance in resource-constrained environments.

While the SlimLite Network previously showed the fastest performance in the Raspberry Pi 4 environment, the modified MobileNetV1 0.25 model showed superior performance after ONNX optimization. The limited performance improvement for the SlimLite Network after ONNX optimization may be due to its already lightweight structure, which reduces potential optimization gains, or possible interactions with existing techniques that affect parallel processing or memory utilization efficiency.

TABLE V. INFERENCE TIMES WITH ONNX OPTIMIZATION ON RASPBERRY PI 4

Methods	Min Time(ms)	Max Time(ms)	Avg Time(ms)
MobileNetV1 0.25 (base)	514.10	535.39	524.89
MobileNetV1 0.25 (modified)	314.48	404.26	327.24
RFBLite	417.20	493.08	432.05
SlimLite	515.57	569.19	548.80
MobileNetV1 0.25 (modified, 320x240)	**91.12**	**120.07**	**106.16**

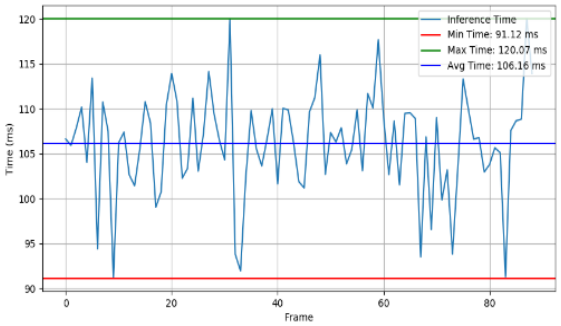
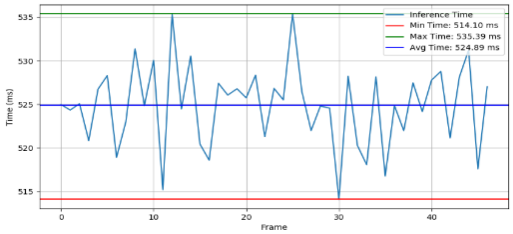
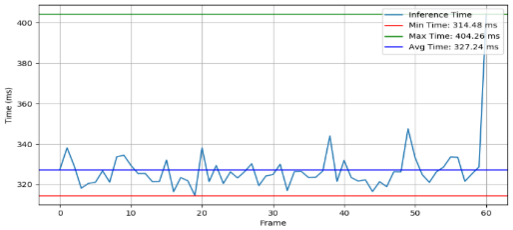


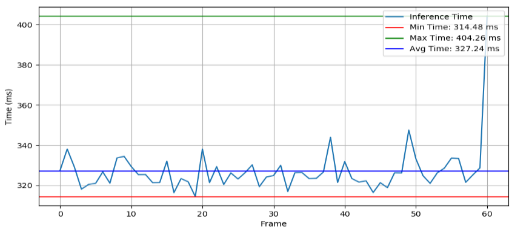
Figure 4. Real-Time Inference Times for MobileNetV1 0.25 (modified) with ONNX Optimization at a 320x240 Resolution



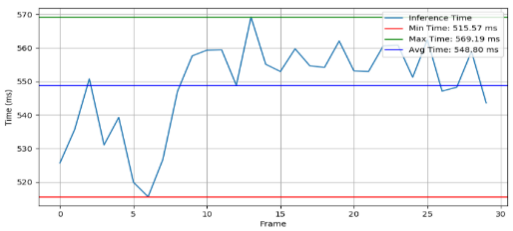
(a) MobileNetV1 0.25 (base)



(b) MobileNetV1 0.25 (modified)



(c) RFBLite



(d) SlimLite

Figure 5. Real-time Inference Times of Different Models with ONNX Optimization (Raspberry Pi 4 Environment)

These results emphasize that additional optimization does not always guarantee performance improvement, and highlight the importance of selecting an appropriate optimization strategy tailored to the specific application environment.

V. Conclusion

This study introduced and evaluated various optimization and lightweighting techniques to improve the performance of real-time face detection models in resource-constrained environments. Using the WIDER FACE dataset, we evaluated the effectiveness of the SlimLite Network and the RFBLite Network - both based on a lightweight RetinaFace model with a MobileNetV1 0.25 backbone - and applied techniques such as ONNX optimization to improve model efficiency.

The performance evaluation showed that the SlimLite and RFBLite networks, designed with lightweight architectures, significantly reduced the number of parameters and computational load while maintaining real-time performance in resource-constrained environments. In the Raspberry Pi 4 environment, SlimLite initially showed the fastest performance, but after applying ONNX optimization, the modified MobileNetV1 0.25 model outperformed it. This finding suggests that ONNX optimization may have a limited impact on models that are already lightweight, such as SlimLite, but can significantly improve the performance of models such as the modified MobileNetV1 0.25.

In particular, when the resolution of the modified MobileNetV1 0.25 model was reduced to 320x240, it achieved the fastest average inference time of 106.16 ms, demonstrating that real-time performance can be significantly improved by adjusting image resolution, lightweighting, and optimizing the model in a resource-constrained environment. These results highlight the fact that optimal performance can vary greatly depending on resource conditions and hardware performance, and emphasize the need to choose an appropriate optimization strategy for the application environment.

This study presents effective approaches for optimizing real-time face detection performance in resource-constrained environments, such as Software-Defined Vehicles (SDVs), and provides insights into the effectiveness and limitations of each technique. To maximize real-time performance, it is essential to tailor optimization strategies to the specific characteristics and resource constraints of the application environment. Future research will explore more advanced optimization methods and investigate the potential for integrating these lightweight face detection models with multi-object tracking, gesture recognition, and Advanced Driver Assistance Systems (ADAS), thereby broadening their practical applicability.

VI. Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (RS-2023-00245084), Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-01197, Convergence security core talent training business (Soonchunhyang University))

VII. References

[1] Fengjunjie Pan, Jianjie Lin, Markus Rickert. 2024. Automatic Platform Configuration and Software Integration for Software-Defined Vehicles. arXiv:2408.02127.

[2] Jiankang Deng, Jia Guo, Yuxiang Zhou, Jinke Yu, Irene Kotsia, Stefanos Zafeiriou. 2019. RetinaFace: Single-stage Dense Face Localisation in the Wild. arXiv:1905.00641.

[3] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv:1704.04861.

[4] O'Shea, K., & Nash, R. 2015. An Introduction to Convolutional Neural Networks.arXiv:1511.08458.

[5] Valentin Bazarevsky, Yury Kartynnik, Andrey Vakunov, Karthik Raveendran, and Matthias Grundmann. 2019. BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs. In Proceedings of the CVPR Workshop on Computer Vision for Augmented and Virtual Reality, Long Beach, CA, USA. arXiv:1907.05047.

[6] Chollet, F. 2017. Xception: Deep Learning with Depthwise Separable Convolutions.arXiv:1610.02357.

[7] Jia Guo, Jiankang Deng, Alexandros Lattas, and Stefanos Zafeiriou. 2021. Sample and Computation Redistribution for Efficient Face Detection. arXiv:2105.04714.

[8] Crawshaw, M. 2020. Multi-Task Learning with Deep Neural Networks: A Survey. arXiv:2009.09796.

[9] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. 2017. Feature Pyramid Networks for Object Detection. arXiv:1612.03144.

[10] Najibi, M., Samangouei, P., Chellappa, R., & Davis, L. 2017. SSH: Single Stage Headless Face Detector. arXiv:1708.03979.

[11] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. 2016. Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks. Submitted to IEEE Signal Processing Letters. arXiv:1604.02878.

[12] Jianzhu Guo, Xiangyu Zhu, Yang Yang, Fan Yang, Zhen Lei, and Stan Z. Li. 2020. Towards Fast, Accurate and Stable 3D Dense Face Alignment. In Proceedings of the European Conference on Computer Vision (ECCV 2020). arXiv:2009.09960.

[13] Ioffe, S., & Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv:1502.03167.

[14] Agarap, A. F. 2018. Deep Learning using Rectified Linear Units (ReLU). arXiv:1803.08375.

[15] Liu, S., Huang, D., & Wang, Y. 2018. Receptive Field Block Net for Accurate and Fast Object Detection. arXiv:1711.07767.

[16] Lin, W.-F., Tsai, D.-Y., Tang, L., Hsieh, C.-T., Chou, C.-Y., Chang, P.-H., & Hsu, L. 2019. ONNC: A Compilation Framework Connecting ONNX to Proprietary Deep Learning Accelerators. In Proceedings of the 2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), Hsinchu, Taiwan, March 18-20, 2019. IEEE.

[17] Yang, S., Luo, P., Loy, C. C., & Tang, X. 2015. WIDER FACE: A Face Detection Benchmark. arXiv:1511.06523.