Hybrid Iris Recognition for Authentication Systems in AI Edge Device (NPU) Environments

Hui-Jae Bae *, Eun-Su Yun *, Seon-Kyun Park *, Jong-Weon Kim †, Dae-Sik Jung ‡
* *Dept. of Computer Science, Sangmyung University, Seoul, South Korea*
† Dept. of *Intelligent IoT Convergence, Sangmyung Universigy, Seoul, South Korea*
‡ *Dept. of Faculty of SW Convergence, College of Convergence Engineering, Sangmyung University, South Korea*
Emails: *baehj100@gmail.com, hkhk0331@gmail.com, qkstjsrbs315@gmail.com, jwkim@smu.ac.kr, jungsoft97@smu.ac.kr*

***Abstract*— In this study, we developed a deep learning-based hybrid iris recognition system that operates on AI edge devices (NPU) and analyzed its performance in various hardware environments. The goal is to extract the unique patterns of the iris for fast and accurate recognition. We applied the YOLOv8 model for the segmentation of the iris, pupil, and eyelid regions. Furthermore, eyelash removal and iris code extraction were conducted to enhance the matching process. Performance evaluation results showed that the average IoU (Intersection over Union) value across all classes was above 0.94, and the inference speed in the NPU environment was 42.7 ms. This research presents an iris recognition system that demonstrates high performance even in edge device environments. Future research aims to improve generalization performance by utilizing diverse datasets and implement a real-time system.**

Keywords—Deep Learning, Iris Recognition, Biometrics, NPU, Inference Speed

I. INTRODUCTION

Recently, with the growing demand for contactless biometric authentication [1], iris recognition technology has become an essential component in various fields, including airport security, financial authentication, and medical data protection. The iris exhibits a unique pattern inherent to each individual, which remains largely unchanged throughout adulthood. [2] This distinctiveness is evident even among identical twins, as their iris patterns differ.[3] Due to this stability, the iris offers high reliability as a biometric authentication method.

Over the past decade, significant technological advancements have been achieved through deep learning. While deep learning has primarily been applied to facial recognition, recent research has also proposed the use of these technologies for iris recognition. However, deep learning typically requires substantial computational resources, which can result in long response times on edge devices, depending on the network architecture. To address this challenge, we propose a hybrid iris recognition method that utilizes deep learning for iris region extraction and incorporates traditional iris recognition techniques using a 1D Gabor wavelet filter.

Figure 1 illustrates the overall process of the proposed method.

1) Acquire an eye image through a camera.

2) The acquired image is segmented into the iris, pupil, and eyelid regions using the YOLOv8s segmentation model.

3) The iris region is separated before extracting the iris feature code. Based on the segmentation result, the area other than the iris region is masked, and the center and radius of the inner and outer circles of the iris region are calculated.

4) Detect eyelashes and exclude the eyelash region when extracting the iris feature code.

5) Compare the extracted iris feature code with the information stored in the DB to verify the identity. If the identity is not verified, check whether to register the iris code. code extraction. Finally, the extracted iris feature code is compared with the database to verify the identity. If the identity cannot be confirmed, a check is performed to determine whether to register the iris code.

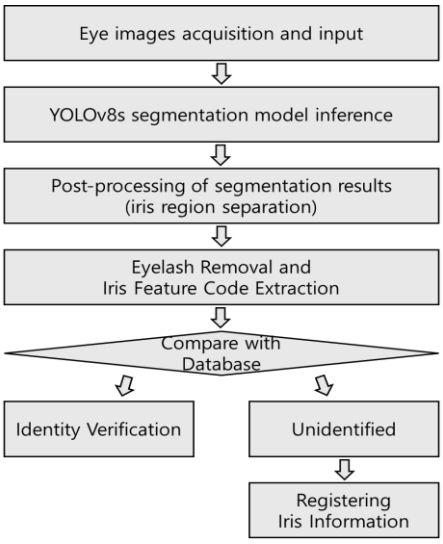


Fig 1. Overview of Hybrid Iris Recognition in AI Edge Device Environments.

II. RELATED WORK

A. Polar Coordinate Transformation-Based Iris Recognition

The study "Iris Recognition Based on Feature Detection in Polar Coordinate Circular Iris Images" (2007, Dae-sik Jeong, Gang-ryeong Park)[4] proposed a polar coordinate transformation method that facilitates distortion-free feature extraction while preserving the circular structure of the iris. This method converts the circular structures of the iris and pupil into polar coordinates based on their respective centers. By addressing the distortion issues inherent in traditional rectangular methods, this approach enhances the accuracy of feature extraction.

The polar coordinate transformation process involves the following steps. First, the boundaries of the pupil and iris are precisely detected in the input iris image. Subsequently, the image is transformed into polar coordinates using the radius

(r) and angle (θ) relative to the center point. The resulting polar coordinate image preserves the structural integrity of the circular shape while being stored in a distortion-free format. During this process, a Gabor wavelet filter is employed to extract features that capture the unique texture information of the iris, including its frequency and directional characteristics. The polar coordinate transformation method has been shown to significantly improve recognition performance compared to the traditional rectangular approach. Specifically, the Equal Error Rate (EER) was reduced by 25%, and the d' value increased from 5.8 to 6.4, indicating improved discriminatory power.

B. *Impact of Activation Functions on YOLO Model Performance*

The In deep learning models, activation functions play a critical role in determining the performance and efficiency of the network. In object detection models like YOLO, the choice of activation function has a significant impact on balancing inference speed and accuracy. Sudhakar Sah et al. [5] proposed a method to optimize YOLO models for specific hardware platforms (CPU, NPU, and GPU edge devices) by tailoring the combination of activation functions. Their research demonstrated that optimizing activation functions such as SiLU, ReLU, and Hardswish across YOLO model layers effectively minimizes latency and memory usage while mitigating accuracy loss.

Replacing SiLU with ReLU in the YOLOv8m model showed a significant improvement in speed with minimal accuracy (mAP) reduction compared to the SiLU-based model. this study replaces the activation function in YOLOv8s-seg from SiLU to ReLU to enhance the processing speed of iris segmentation in edge board environments.

III. EXPERIMENTS

The experimental setup in this study consists of three stages. The first stage involves segmentation tasks using the YOLOv8 model to separate the iris, pupil, and eyelid regions. Through post-processing, the iris region is extracted, and the center coordinates and radii of the iris and pupil are calculated. The second stage includes eyelash removal and the generation of iris codes, which are utilized for matching tasks based on Hamming distance. The third stage focuses on model optimization and measuring inference time in CPU, GPU, Jetson Nano, and NPU environments to compare performance. This assessment evaluates whether the NPU can operate efficiently and quickly in the iris recognition system.

A. *Data Set*

1) *CASIA-IrisV4-Lamp*

In this study, the CASIA-IrisV4-Lamp database [6] was utilized for the development of the iris recognition algorithm. The CASIA-IrisV4-Lamp consists of images collected using portable iris sensors, where lighting conditions were varied by turning a lamp on or off near the subject. This makes the database particularly useful for studying iris texture variations under different illumination conditions. The CASIA-IrisV4-Lamp database comprises a total of 16,212 high-resolution images (640x480 pixels), including a wide range of lighting conditions and texture variations, making it well-suited for evaluating the performance of iris recognition algorithms.

2) *Data Preprocessing*

From the CASIA-IrisV4-Lamp database, 1,060 images were selected for data labeling to train the segmentation model. The labeling process involved defining three classes: iris, pupil, and eyelid, and annotating the boundaries of each class region in each image. The labeling results consisted of boundary coordinates for each class region, which were used as input data for training the segmentation model. Figure 2 illustrates an example of the labeling process, showing the boundaries of the iris, pupil, and eyelid regions.

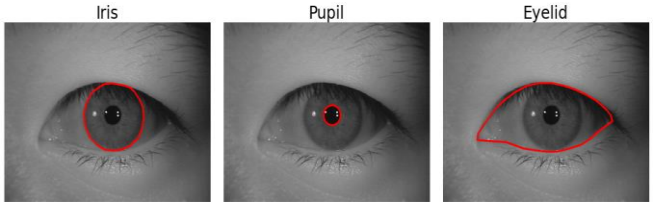


Fig 2. Examples of Labeling Results for Iris, Pupil, and Eyelid Regions

B. *Iris, Pupil, and Eyelid Regions Segmentation*

1) *YOLOv8 Segmentation Model*

In this study, the YOLOv8 model [7] was applied to extract the iris, pupil, and eyelid regions through segmentation tasks. YOLOv8 is a state-of-the-art deep learning model that can perform object detection and segmentation simultaneously, providing fast inference speed and high accuracy. For this research, the lightweight version YOLOv8s-seg was adopted.

During the model training process, the activation function was changed from the existing SiLU [8] to ReLU [9]. SiLU is defined as the product of the input (x) and the Sigmoid function (sigma(x)), as shown in Equation (1). While SiLU provides smooth nonlinear output depending on the input value, it has the drawback of high computational cost due to the inclusion of the Sigmoid operation. ReLU is defined as a simple conditional function that outputs (x) when the input (x) is greater than 0 and outputs 0 otherwise, as shown in Equation (2). ReLU has a simple and efficient computational structure, making it suitable for lightweight environments. It has also been demonstrated in [5] that using ReLU significantly reduces latency while maintaining minimal accuracy loss. Based on this, the YOLOv8s-seg model was optimized in this study by changing the activation function to ReLU to enhance computational efficiency and ensure smooth operation even in constrained hardware environments, such as edge devices.

$$f(x) = \frac{x}{1+e^{-x}} = x \cdot \sigma(x) \quad (1)$$

$$f(x) = \max(0, x) \quad (2)$$

The training dataset was divided into Train, Validation, and Test sets in a ratio of 8:1:1, consisting of 832, 104, and 104 images, respectively. During the training process, a batch size of 16 was used, and the model was trained for a total of 200 epochs. The optimizer was set to AdamW, and the learning rate was configured to 0.05. The training was conducted on a server equipped with an NVIDIA GeForce RTX 3060 Lite Hash Rate GPU.

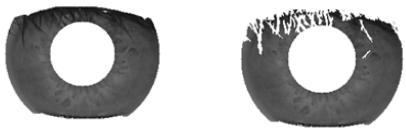
2) Detection of Iris Region Center and Radius

Additional processing was conducted to utilize the iris-pupil mask generated during the segmentation task in subsequent algorithms. In this step, the iris region was isolated using the label information obtained from the segmentation results. Next, the center coordinates and radii of the iris and pupil were calculated and prepared as data for feature extraction and classification tasks. The detailed processing steps are as follows: (1) Iris region extraction: The pupil mask is removed from the iris mask to separate the pure iris region. (2) Calculation of center coordinates and radius: The center coordinates and radius of the pupil are obtained by extracting the outer line of the pupil mask, selecting the outline with the largest area, and calculating the minimum circumscribed circle. The center coordinates of the iris are assumed to be the same as the center of the pupil, and the radius is calculated in the same way as the pupil.

C. Iris Feature Code Extraction

1) Eyelash Removal

Eyelash extraction is a crucial step in the preprocessing phase, as it can significantly affect the integrity of the iris pattern. Using the preprocessed image segmented into iris and pupil regions, eyelashes are identified in areas with high contrast in pixel values. These identified regions are then set to white (pixel value: 255) to exclude them from the iris code. In Figure 3, (a) shows the image before eyelash extraction, while (b) shows the image after eyelash extraction.



(a) (b)

Fig 3. Examples of Iris image : (a) Before eyelash extraction image, (b) After eyelash extraction image

2) Iris Code Extraction

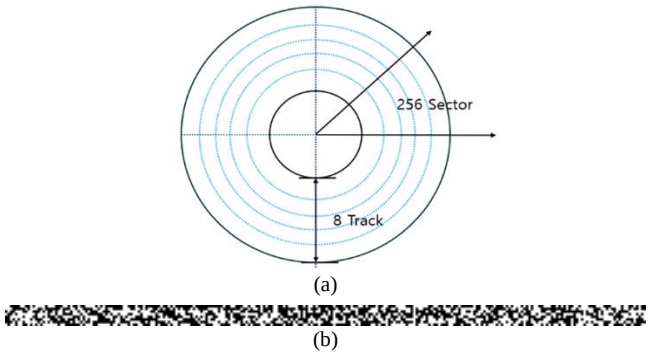
In this study, a polar coordinate-based circular iris feature extraction technique from [4] was utilized to implement the iris recognition system. Instead of the commonly used 2D Gabor Wavelet Filter [10] for generating the iris code, a 1D Gabor Wavelet Filter [11] was applied.

$$G(x) = Ae^{-\pi\left[\frac{(x-x_0)^2}{\sigma^2}\right]}(\cos(2\pi[\mu_0(x-x_0)])) \quad (3)$$

$$\sum_{X_0}^{N} G(x) = 0 \quad (\because DC = 0) \quad (4)$$

In Equation (3), the Gabor filter from the traditional Daugman method was modified to address the issue of varying image brightness affecting the extracted code values. To achieve this, the DC component of the filter values was adjusted to 0, as shown in Equation (4).

To extract the iris code from the input image with eyelashes removed, a polar coordinate-based method, as illustrated in Figure 4(a), was employed. This approach is based on 8 tracks and 256 sectors, applying the Gabor filter as in Equation (3) to represent the computed values as bits of 0 and 1. This method minimized distortion in the iris features and enabled more accurate extraction of the iris feature code. Ultimately, the iris code extracted from a single image is 4096 bits, as shown in Figure 4(b).



(a)

(b)

Fig 4. Iris Code Extraction Process: (a) 8 Tracks, 256 Sectors, (b) Extracted Iris Code

3) Iris Recognition

The extracted iris code was utilized in the matching process to differentiate between the individual and others using Hamming distance. Given iris codes A and B, the normalized Hamming distance obtained by comparing each bit of A and B is expressed in Equation (5) as follows.

$$HD = \frac{1}{N}\sum_{i=1}^{N} A_i(XOR)\,B_i \quad (5)$$

If the Hamming distance is smaller than a predefined threshold, the system recognizes the two iris codes as belonging to the same individual; otherwise, it identifies them as belonging to different individuals [12]. Using the Hamming distance, iris recognition is conducted by comparing the extracted iris codes with the reference dataset.

To ensure the reliability of the recognition results, the accuracy and performance of the iris feature extraction algorithm were evaluated using the Equal Error Rate (EER). EER is defined as the rate at which the False Acceptance Rate (FAR), the probability of incorrectly accepting a non-matching biometric as a match, equals the False Rejection Rate (FRR), the probability of incorrectly rejecting a genuine biometric [13]. EER can be conveniently obtained from the ROC curve, and generally, systems with the lowest EER are considered the most accurate.

D. Model Optimization and Performance Analysis in Various Environments

To measure the inference time of the model in various environments, experiments were conducted using the CPU (Intel(R) Core(TM) i5-12400F), GPU (NVIDIA GeForce RTX 3060 Lite Hash Rate), Jetson Nano, and NPU of the server on which the model was trained. An NPU (Neural Processing Unit) is a neural processing device that quickly handles neural operations while consuming less power. [14] Using an NPU allows for faster training and inference in artificial intelligence tasks such as deep learning by utilizing accelerators for lower power consumption. [15][16]

In addition to optimizing the activation function of the YOLOv8 Segmentation model, further optimizations were performed on the Jetson Nano and NPU. First, on the Jetson Nano, to maximize performance, the PyTorch model file (.pt) was converted into a TensorRT format file (.engine).

TensorRT[17] is NVIDIA's deep learning inference optimization library that employs acceleration techniques such as Quantization & Precision Calibration, Graph Optimization, Kernel Auto-tuning, Dynamic Tensor Memory, and Multi-stream execution. This optimization allows the model to run faster and more efficiently. For the NPU, model lightweighting was performed by converting the PyTorch (.pt) file provided by Ultralytics into an ONNX file [18]. After that, within the NPU, the model's data type was changed to floating point, and retraining was conducted. The file was then modified to match the format required by the NPU for experimentation.

In the performance measurement process, hardware-specific inference time analysis was performed using the test set. It was measured by dividing the steps into preprocessing, inference, and postprocessing. Preprocessing refers to the process of processing data from the acquired image to the input form for segmentation. Inference refers to the process of iris segmentation as shown in Figure 1, and postprocessing refers to the process of processing the label information when outputting a txt file as a result of segmentation.

IV. Results

A. Performance of the Segmentation Model

The performance of the segmentation model was evaluated using a test set. To assess performance, the metrics Intersection over Union (IoU), Precision, Recall, and mAP50 were utilized. IoU is calculated as the ratio of the overlapping area between the predicted mask from the segmentation model and the actual mask. The average IoU measured for each class in the test set is shown in Table 1. The average IoU value recorded for all classes was above 0.94, with the eyelid class achieving the highest value of 0.97.

TABLE 1. Average IoU Results by Class

Class	Average IoU
Iris	0.96
Pupil	0.94
Eyelid	0.97

TABLE 2. Performance Evaluation Results by Class

Class	Precision	Recall	mAP50
Iris	0.99	1	0.99
Pupil	1	1	0.99
Eyelid	0.99	1	0.99

The results of Precision, Recall, and mAP50 measured for each class in the test set are presented in Table 2. For all classes, Precision was above 0.99, Recall was 1, and mAP50 was 0.99.

B. Evaluation of Iris Classification Algorithms

To evaluate the iris recognition algorithm, Hamming distance and EER were used. Figure 5 shows an example image of measured Hamming distance. The proposed method measures the Hamming distance between the input iris feature code and the registered iris feature code. If the measured Hamming distance is less than a predefined threshold, the individual is authenticated as themselves. Conversely, if the Hamming distance exceeds the threshold, the individual is authenticated as someone else. We set the threshold at 0.45, and the measured EER at this threshold was 0.25.

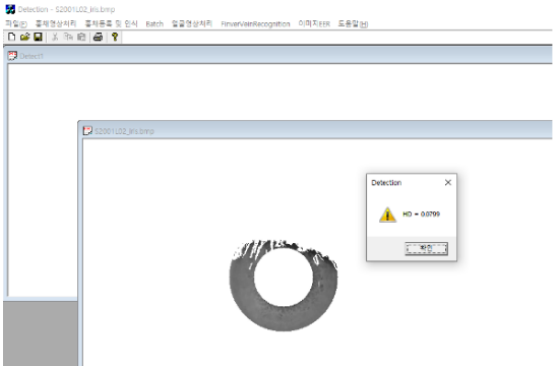


Fig 5. Image showing the measurement of Hamming distance through the iris feature extraction algorithm.

C. Inference Time Analysis Results by Hardware

TABLE 3. Inference Time Results by Hardware

Time per image (ms/images)	Preprocess	Inference	Postprocess	Total
CPU (server): AMD Ryzen 9 5900X 12-Core Processor	0.6	66.4	1.5	68.5
GPU (server): NVIDIA GeForce RTX 3060 Lite Hash Rate	0.7	7.0	4.2	11.9
Jetson Nano (Pytorch)	0.5	437.5	19.9	457.9
Jetson Nano (TensorRT)	66.2	225.0	39.5	330.7
NPU	0.4	42.7	4.3	47.4

The processing time was measured and evaluated across various cores. The NVIDIA GeForce RTX 3060 achieved the shortest processing time of 11.9 ms. The proposed method demonstrated a processing time of 47.4 ms on the NPU, indicating the capability for real-time processing at 21 frames per second (fps) in edge device environments.

V. Conclusion

In this study, we proposed a hybrid iris recognition method that operates in an AI edge device (NPU) environment and conducted experiments across various core environments. We utilized the YOLOv8 model to segment the iris, pupil, and eyelid regions, and optimized the activation function by changing it from SiLU to ReLU to enable efficient performance even in constrained hardware environments. Additionally, 1D Gabor Wavelet based polar coordinate algorithm was applied to reduce feature distortion and enhance matching accuracy. Furthermore, we performed various optimizations on the Jetson Nano and NPU to maximize hardware performance.

The model was trained and tested using the CASIA-IrisV4-Lamp dataset, and the results showed that the average IoU value across all classes was above 0.94, indicating high performance. Furthermore, in iris feature extraction and recognition, the Equal Error Rate (EER) was recorded at 0.25, confirming stable performance in recognition tasks. The inference time measurements for the model by hardware showed an average inference speed of 42.7 ms in the NPU environment. This was faster than the Jetson Nano and comparable to server CPUs. Based on these results, the potential for rapid real-time iris recognition at 21 fps in AI edge device environments was confirmed.

However, this study has limitations as it primarily focused on the CASIA-IrisV4-Lamp dataset, resulting in insufficient validation across various environments. Additionally, the actual performance in real-time systems was not evaluated, leading to a lack of validation for practical applications. Future research will leverage various datasets to enhance the model's generalization performance and the performance of the iris feature extraction algorithm. We also plan to apply model compression techniques to improve the iris recognition system's efficiency for real-time environments.

References

[1] Global information, “Contactless Biometrics Technology Market Size, Share & Trends Analysis Report By Component (Hardware, Software, Services), By Application, By End-use, By Region, And Segment Forecasts, 2025-2030”, 2024. [Online].Available: https://www.giikorea.co.kr/report/grvi1588339-contactless-biometrics-technology-market-size.html

[2] M. H. Cho, J. Y. Hur, “The Study on Searching Algorithm of the center of Pupil for the Iris Recognition”, Journal of The Korea Society of Computer and Information (JKSCI), vol. 11, no. 1, pp. 19-26, March 2006.

[3] NRF, “A Study on Biometric Personal Identification Based on Iris Analysis”, R05-2001-000-00926-0, 2002-2003

[4] D. S. Jeong and K. R. Park, "A Study on Iris Recognition by Iris Feature Extraction from Polar Coordinate Circular Iris Region," The Institute of Eletronics and Information Engineers(Signal Processing), vol. 44, no. SP-3, pp. 48–59, May 2007.

[5] S. Sah, R. Kumar, D. C. Ganji, and E. Saboori, "ActNAS: Generating Efficient YOLO Models using Activation NAS,", NeuRIPS 2024, doi: 10.48550/arXiv.2410.10887

[6] Chinese Academy of Sciences, "CASIA-IrisV4: Iris Image Database," 2020. [Online]. Available: https://hycasia.github.io/dataset/casia-irisv4/?utm_source=chatgpt.com.

[7] Ultralytics, "YOLOv8: Object Detection and Segmentation Models," 2023. [Online]. Available: https://docs.ultralytics.com/ko/models/yolov8/?utm_source=chatgpt.com.

[8] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for Activation Functions," arXiv preprint arXiv:1710.05941, 2017.

[9] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814, 2010.

[10] J. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters," J. Opt. Soc. Am. A, vol. 2, no. 7, pp. 1160–1169, July. 1985

[11] Dae Sik Jeong, Hyun-Ae Park, Kang Ryoung Park, Jaihie Kim, “Iris Recognition in Mobile Phone Based on Adaptive Gabor Filter”, Lecture Notes in Computer Science (ICB), Vol. 3832, pp.457~463, January 5-7, 20

[12] K. R. Park, “Iris Recognition Technology”, Korea Multimedia Society, vol. 7, no. 2, pp. 23-31, June. 2003

[13] T. I. Kang, I. Y. KWAK, “A two-stage training approach for voice spoofing detection”, The Korean Data Information Science Society(JKDIS), vol. 34, no. 2, pp.203-214, March. 2023.

[14] S. C. Kim, Y. Y. Kim, T. H. Kim, “Development of NPU Operating System Platform for Inference of Multiple Neural Networks on Multiple Heterogeneous NPU Devices”, vol. 26, no. 12, pp. 561-566, December 2020

[15] J. H. Park, M. Kim, Y. Kim, K. Lee, M. Yoon, W. W. Ro, "Research Trends of Hardware Accelerators for Artificial Neural Networks," Communications of the Korean Institute of Information Scientists and Engineers, Vol. 34, No. 9, pp. 21-26, Sep. 2016. (in Korean)

[16] M. Yu, Y. Ha, T. Kim, "Trends in Neuromorphic Software Platform for Deep Neural Network," Electronics and Telecommunications Trends, pp. 14-22, 2018. (in Korean)

[17] NVIDIA, "TensorRT," 2024. [Online]. Available: https://developer.nvidia.com/tensorrt.

[18] ONNX, “ONNX”, 2024. [Online]. Available: https://onnx.ai/