Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Online Multimodal Compression using Pruning and Knowledge Distillation for Iris Recognition

Samihah Abdul Latif[1], Khairul Azami Sidek[1,*], Eddyzulham Abu Bakar[2], Aisha Hassan Abdalla Hashim[1]

[1] Department of Electrical and Computer Engineering, Kuliyyah of Engineering, International Islamic University Malaysia, Selangor, Malaysia
[2] Department of Electrical Engineering, Sultan Azlan Shah Polytechnic, Perak, Malaysia

ARTICLE INFO

Article history:
Received 29 June 2023
Received in revised form 18 October 2023
Accepted 8 November 2023
Available online 16 January 2024

Keywords:
Model compression; pruning; knowledge distillation; iris recognition

ABSTRACT

Deep learning models have advanced to the forefront of image recognition tasks, resulting in high-performing but enormous neural networks with millions to billions of parameters. Yet, deploying these models in production systems imposes considerable memory limits. Hence, the research community is increasingly aware of the need for compression strategies that can reduce the number of model parameters and their resource requirement. Current compression techniques for deep learning models have limitations in efficiency and effectiveness, indicating that more research is required to develop more efficient and practical techniques capable of balancing the trade-offs between compression rate, computational cost, and accuracy. This study proposed a multimodal method by combining multimodal Pruning and Knowledge Distillation techniques for compressing the iris recognition model, which is the size constraint for many image recognition models. To maintain accuracy while shrinking the model's size, the models are trained, compressed, and further retrained in the downstream job. The analysis includes both fully connected and convolutional layers. Experimentally, the findings show that the proposed technique can achieve 91% accuracy, the same as the existing or original model. Besides that, the model compression can reduce the size of the model almost six times, from 529MB to 90MB, which is a significantly reduced rate. The primary outcome of this study is developing a CNN lightweight model for iris recognition technology that can be used on mobile devices and is resource constrained.

1. Introduction

A Convolutional Neural Network (CNN) is a type of deep learning model commonly used in computer vision tasks such as image classification, object detection, and segmentation. CNN consists of multiple layers, including convolutional layers, that use filters to extract features from input images, while pooling layers down to sample the output of the convolutional layers. Reduced storage and computing costs become crucial as larger neural networks with more layers, weights, and nodes are examined, especially for some real-time applications that are implemented on the embedded

* *Corresponding author.*
E-mail address: azami@iium.edu.my

https://doi.org/10.37934/araset.37.2.6881

system [1-3]. Novel computing paradigms and emerging technologies are under investigation to make CNNs available for edge computing [4]. However, it is important to make these models more efficient in situations where resources are constrained, such as deployment on mobile devices [5]. When compared to traditional feature recognition algorithms, deep learning has the greatest advantage of enhancing the model's robustness and generalization ability to image noise based on improving accuracy [6,7]. However, large numbers of datasets are required for input during model training to achieve higher accuracy [8].

The development of iris recognition systems, especially for low-powered mobile sensors and portable devices, is currently being driven by the effective storage and transmission of iris biometric records [9,10]. Implementing iris recognition apps on mobile devices requires model compression due to the limitations of mobile hardware, such as limited memory and processing power. A program or model that successfully runs on a system might not successfully run on a device with a lower processing power. The hardware limitations of the target device may be the cause of this.

Model compression reduces the size and complexity of the iris recognition model while maintaining its accuracy. This is achieved by removing unnecessary parameters, reducing the number of layers, and using more efficient algorithms. Model compression is essential for deploying iris recognition on mobile devices as it reduces computational requirements, speeds up the inference process, and reduces power consumption [9]. By compressing the iris recognition model, the app can run smoothly on mobile devices without compromising the accuracy of the authentication process. Therefore, model compression is critical in implementing iris recognition apps on mobile devices.

Although methods for lowering the size and complexity of deep learning models using model compression can be helpful, they also have limitations and may only sometimes be effective in all situations [3]. More reliable model compression techniques are required to deal with more complex tasks [11]. Pruning methods have been widely employed in prior research to compress CNN models [12,13]. These strategies, however, can result in sparse networks that may only be efficient on some hardware platforms. Therefore, this study uses pruning and knowledge distillation to avoid sparsity in the network and can integrate into mobile platforms. It is important to carefully evaluate the trade-offs and choose the most appropriate technique for a particular application. As a result, a reasonable option is to compress and accelerate the deep convolution neural network while maintaining recognition accuracy. The trade-off between preserving accuracy while reducing the model size and inference time, however, needs to be addressed by any of the aforementioned research that integrates model compression.

This study aims to design, develop and compare, in terms of compression of model size and accuracy, iris recognition that can be performed as a lightweight version. Thus, this lightweight model can embed into the mobile platform and achieve the same result as the original model. The rest of the paper is divided into the following sections. Section 2 offers an overview of the model compression techniques and the study involved. Section 3 describes a recommended solution. The simulation results and discussions are presented in section 4 before the conclusion.

2. Related Works

The introduction of deep learning in vast amounts of data has increased the true utilization of the data, and deep learning accomplishes this by traversing those data on millions of factors. However, this has increased the demand for computer resources such as GPU, which are absent in cutting-edge devices like mobile phones. As a solution to this problem, researchers have developed a variety of compression approaches, including Parameter Pruning, Optimization, Knowledge Distillation, and Low-rank Factorization, which are the process of transforming complicated model behavior into a smaller one in terms of fewer parameters. Figure 1 depicts a summarization of four techniques used in compressing a CNN model based on research of [1,4,11].

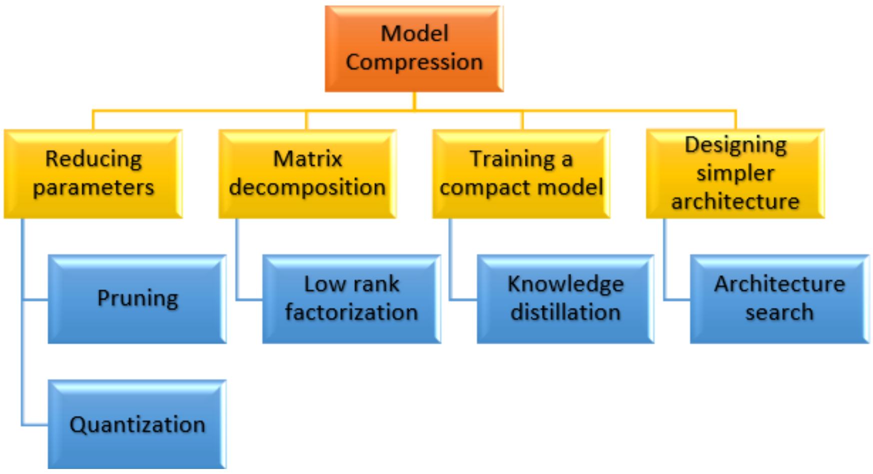

Fig. 1. Summarization of model compression methods

In deep learning, pruning aims to build a more compact and practical neural network model [14]. By eliminating the weight tensor values as in Figure 2, this method aims to optimize the model. The goal is to create a computationally efficient, less time-consuming model to train. Pruning is essential since it reduces time and resources and enables the model to function on low-end devices like mobile and other edge devices. There have been several studies in this research field; some are pruning from scratch and Adversarial Neural Pruning [14,15]. These methods have the advantage that previously removed weights can be added back in later; however, due to their great complexity, these can only be applied to CNNs of specific sizes [16]. The [15] study optimizes the pruning approach, which may considerably reduce pruning time by modifying step size according to the sensitivity of each layer, resulting in a 54% reduction in FLOPS, a 71% reduction in parameter and model sizes, and a 1.85% reduction in mean Average Precision (mAP).

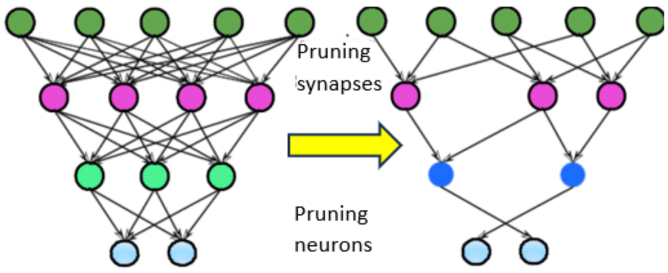

Fig. 2. Weights pruning method

Quantization is a method to reduce the number of bits required to represent data in model compression as shown in Figure 3 [14]. Additionally, it can greatly minimize the need for processing power and storage. In deep learning, quantization normally refers to converting from floating numbers to constant factor integers. There could be an accuracy loss and avoid accuracy loss in post-training model quantization by conducting quantization-aware training. The study by Ref. [17] applied the quantization method reduces the model size of a trained neural network by a quarter while retaining high accuracy in classification. In the context of edge computing, this finding can contribute to cost savings and improved performance when implementing deep learning algorithms like convolutional neural networks on limited devices like FGPAs.

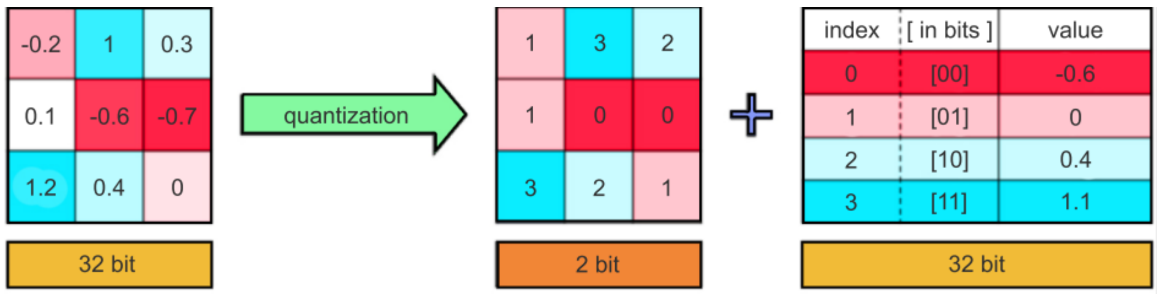

Fig. 3. Binary quantization method

Knowledge Distillation is a model compression technique that involves training a small model known as a student to match or imitate the behavior of a larger model which is known as a teacher model. Figure 4 shows this technique aims on reducing a loss function focused on matching softening teacher logits as well as ground-truth labels, knowledge is transferred from the teacher model to the student. Deep neural networks are one of the numerous types of models that can benefit from knowledge distillation in terms of performance. The study's findings revealed that by implementing the knowledge distillation method outperformed traditional CNN models and other lightweight methods [18]. The proposed model has fewer parameters and reduces computational complexity, making it better suited for real-time recognition tasks.

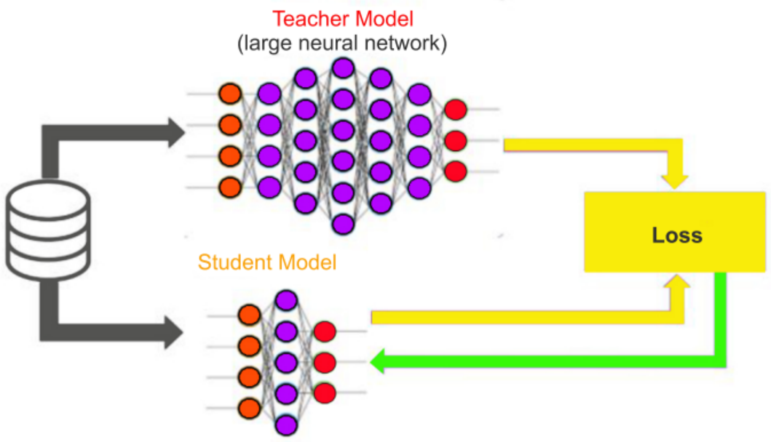

Fig. 4. Knowledge distillation method in model compression

The method of model compression known as low-rank factorization has grown in popularity recently. A lower-rank matrix is used to approximate a matrix in this method to keep them more effective while retaining as much information as possible [19]. This can be done by using singular value decomposition (SVD) or by using other methods such as eigenvalue decomposition. This method can be used to compress any sort of matrix, including neural network matrices. It has been demonstrated that low-rank factorization is a reliable method for downsizing neural networks without sacrificing performance. In the study by Ref. [19], the cost function of compression and the loss function are combined to form a joint function that is optimized as an optimization framework and combined with the CUR decomposition technique. The factored matrices can also be stored in a way that is more effective than the original matrix as illustrated in Figure 5. Consequently, this method can be utilized to lower the memory needs of models that use machine learning.

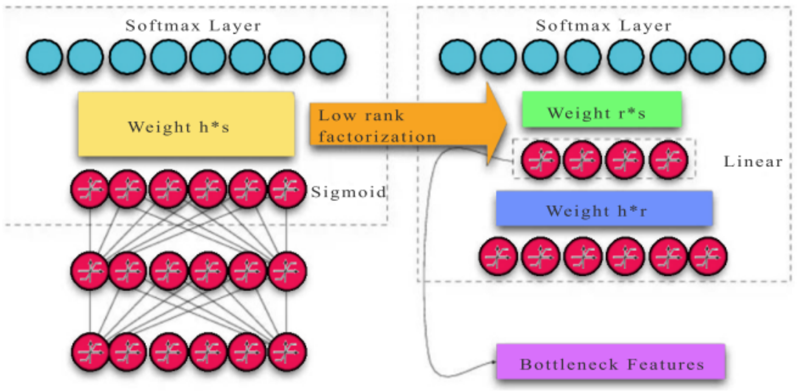

Fig. 5. Low-rank factorization method in model compression

In conclusion, pruning involves removing the least important connections or weights in a model, which can lead to a significant reduction in the number of parameters. However, this technique may not work well for all types of models, and it can be difficult to determine which connections to prune. To reduce the size of the model, quantization involves using fewer bits to represent each weight parameter. However, this can also result in a loss of accuracy due to rounding errors. Knowledge distillation involves training a smaller model to mimic the behavior of a larger model. While this can be an effective way to compress a model, it requires access to the larger model during training, which may not always be possible. Therefore, this study is combining pruning and knowledge distillation methods to achieve better results.

3. Methodology

This section will go over the CNN model, performance parameters, and datasets used in this model compression, as well as the proposed algorithm.

3.1 CNN Model

Visual Geometry Group (VGG) is a deep neural network model that has experienced widespread application in computer vision. Figure 6 shows the VGG architecture that consists of multiple layers of blocks, where each block is composed of 2D Convolution and Max Pooling layers. It has a high number of connections, which can contribute to greater computational complexity and slower inference times. VGG uses very small (3x3) convolution filters and supports up to 19 layers. All VGG's hidden layers use Rectified Linear Unit (ReLU) which is a huge innovation from AlexNet that cut training time and a pre-trained version of the network trained on more than a million images from ImageNet is available. VGG16 can be compressed using model compression techniques to minimize its size while retaining its accuracy. These techniques include network quantization, low-rank factorization, pruning, and knowledge distillation. Compression methods have been shown to reduce the size of VGG16 by up to 35 times with minimal loss in accuracy.

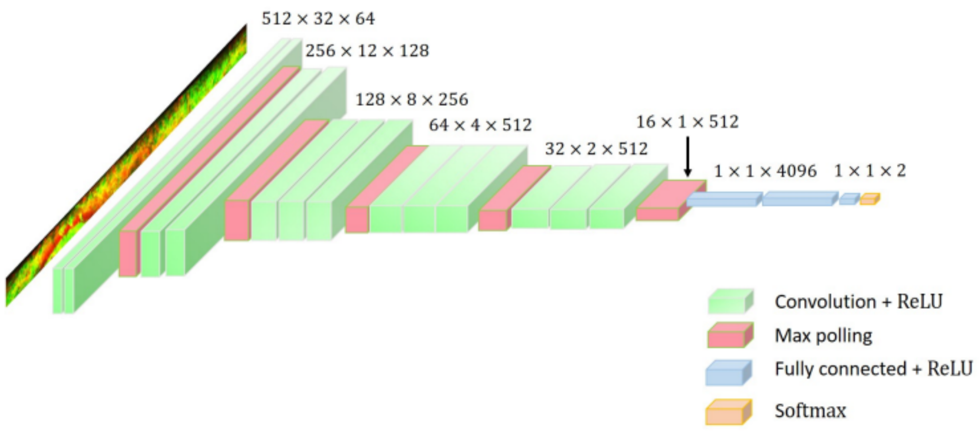

Fig. 6. Architecture of VGG16 [20]

The VGG16 model had the highest recognition accuracy, according to a study that assessed the effectiveness of three deep learning models for iris recognition, including AlexNet, VGG16, and VGG19 [21]. In this study, the VGG16 model was used due to the dataset obtained is in small quantity, so it is sufficient to use VGG16. Therefore, it is adequate to utilize a CNN model with 16 layers to assess the efficacy of the compression model approach because there are only 1000 iris images, in contrast to ImageNet or CIFAR100, which contain tens of thousands to millions of images. Additionally, it has been demonstrated that the VGG16 algorithm performs well on a variety of image classification tasks, including iris recognition [22,23]. Although VGG19 has more layers and parameters than VGG16, this does not ensure that it will perform better on the iris recognition task. A model that has more layers will not provide an advantage in terms of experimental results but only increases processing time and this could be a reasonable option.

3.2 Performance Parameters

The performance parameters that are evaluated in this study include accuracy, model size or compression rate, and inference time.

3.2.1 Accuracy

Accuracy is a measure of how well a model can correctly predict the correct output of a given input. It is typically expressed as a percentage, with a higher percentage indicating better performance. When compressing a model, it is important to maintain a high level of accuracy. However, the process of compression can sometimes lead to a reduction in accuracy, as some of the model's complexity is removed. This reduction in accuracy can be acceptable if the compressed model is still able to perform well enough for its intended application. In summary, accuracy is a critical metric for evaluating the performance of compressed deep learning models, and maintaining a high level of accuracy is important when compressing models for real-world applications. Eq. (1) shows the formula to calculate the accuracy of the model.

Accuracy = Number of correct predictions/ Total number of predictions (1)

3.2.2 Model size

The model size is an important parameter in model compression that is used to assess the effectiveness of a compressed model. The amount of memory required to hold the model's parameters is referred to as model size. It is usually measured in terms of the number of parameters or bytes needed to store those parameters. The ability to deploy the model on systems with constrained memory or bandwidth makes a reduced model size desirable. Model compression methods, in general, strive to lower the size of the model while maintaining or enhancing its accuracy. Yet, model size and accuracy are frequently traded off. Although smaller models may be less accurate than larger models, it may be more efficient and easier to deploy. Eq. (2) was used to measure the size of the model before and after compression.

$$\text{Size of model} = \text{Number of parameters x Size of each parameter} \quad (2)$$

3.2.3 Inference time

Inference time is an important metric in model compression that measures the amount of time required for the model to make predictions on new data. Inference time is extremely crucial in applications that require real-time predictions, such as autonomous vehicles, video processing, or image recognition. The model is more effective and better suited for real-time applications since a shorter inference time enables it to process more data in a given period. Pruning, quantization, and network architecture adjustment are common approaches used during model compression to lower the model's computational complexity.

By lowering the number of computations required to make a prediction, these methods can contribute to faster inference times. To measure the duration, it takes for the model to process the input data and return the predictions, Python’s built-in module called ‘time’ was used in this experiment. In reality, model compression approaches are frequently assessed in terms of their impact on model size, inference time, and correctness. This enables developers to select the compression method that best balances these considerations for a specific application.

3.3 Datasets

In this paper, the performance of iris recognition algorithms was examined using three of the most widely used iris datasets: CASIA, UBIRIS, and MMU iris dataset.

3.3.1 CASIA dataset

The (CASIA) assembled the iris dataset as a collection of iris images for iris identification testing and study in this experiment. The dataset comprises 108 subjects who collectively contributed 1,000 distinct iris images, with each person providing 9 to 10 images. The images were captured with the CASIA Iris Capture system, which takes photos of the iris using a near-infrared (NIR) sensor [24]. The images in the dataset have a resolution of 320 by 280 pixels and are in JPEG format. These are separated into two subsets: the CASIA-IrisV1-Lamp subset, which includes images captured under various lighting environments, and the CASIA-IrisV1-Interval subset, which includes images taken at intervals ranging from a few days to several months.

3.3.2 UBIRIS.v1

This visible light iris dataset was created by the Portugal University of Beira Interior. The dataset contains 1,872 iris images from 624 subjects, each of whom contributed three images [25]. The images were captured with a visible light camera at a frame rate of 25 frames per second with a resolution of 640 × 480 pixels. The images are in JPEG format and range in size from 50 to 60 KB and were taken in two settings: indoors and outdoors. Images captured in controlled lighting conditions are included in the indoor scenario, while images captured in natural lighting conditions are included in the outdoor scenario. The dataset also includes segmentation masks indicating the iris and pupil region boundaries, as well as metadata such as subject ID, gender, and age.

3.3.3 MMU iris dataset

The Multimedia University in Malaysia created this visible light iris dataset. It is made up of 1,000 iris images that a visible light camera recorded from 100 different people with each person providing 10 images [24]. The photographs were taken with a visible light camera with a resolution of 640 x 480 pixels and 8-bit vibrant colors. The images are in BMP format and are around 1.6 MB in size. The images were taken in two separate sessions, separated by a time gap of 1-2 weeks. The dataset comprises segmentation masks indicating the iris and pupil region boundaries, as well as metadata such as subject ID, gender, and age. Images are only partially captured because the subject's eyes are partially closed, the subject is looking away, or the subject's eyelashes partly covered the image of the subject's eyes [26].

3.4 Proposed Algorithm

The iris recognition model must first undergo pruning to eliminate pointless weights and shrink the model. This can be accomplished using the approach of magnitude-based weight pruning. Then, through knowledge distillation, a smaller student model is taught to learn from the original, larger model which is the teacher model. The teacher model's output probabilities are imitated by the student model during training, which can speed up the student model's learning of iris pattern recognition. The student model is refined on the iris detection task to increase its accuracy on the test set after it has been pruned and distilled.

The following Algorithm 1 can be used to visualize the entire architecture:

```
Algorithm 1: Combination of pruning and knowledge distillation.

Input: Load the iris dataset
Output: final model
    i.      Train iris recognition model, epochs=30.
    ii.     Perform magnitude-based weight, ratio=0.5, pruned model
    iii.    Train knowledge distillation model known as a student model.
    iv.     Fine-tune the student model.
    v.      Evaluate the accuracy.
    vi.     Select a model with the best accuracy.
            If student model > pruned model
                Select the student model.
            Else
                Select pruned model.
```

In this study, two functions are crucial: create_large_model() and create_student_model(). For a Keras model, the prune_model() function was used to perform magnitude-based weight pruning. Finally, KnowledgeDistillationCallback() is a Keras callback function that performs knowledge distillation during the implementation of training.

4. Results and Discussion

This section presents the results and discussion of this study on the use of the VGG16 model for iris recognition, multimodal methods for model compression, and finally, the overall deduction will be presented.

4.1 VGG16 Model

The CASIA Iris V1 dataset was used to test each compression model technique in this experiment. Table 1 displays performance results based on accuracy, size for the VGG model, and inference time, which is the time taken to process an iris image.

Table 1
Result of performance parameter using VGG16 model

VGG16	Original	Pruning	Quantization	Knowledge distillation	Multimodal (Pruning & KD)
Accuracy (%)	90	89	89.71	90.19	91
Model size (MB)	528.95	127.34	121.22	107.5	90.22
Inference time (sec)	0.92	0.65	0.72	0.69	0.58

Based on Table 1, without considering the multimodal technique, the highest accuracy is by using knowledge distillation. This technique is not only able to maintain the original accuracy but can also increase the value of this performance to 90.19%. As for the size model, there is a significant reduction from 528.95MB to 107.5MB by using the same compression method. The pruning technique also has the effect of reducing inference time from 0.92 seconds to 0.65 seconds.

Therefore, due to the results obtained, pruning and knowledge distillation were chosen to be hybridized. As expected, the results obtained after implementing this compression technique are very encouraging. Figure 7 shows the accuracy and loss graph by using multimodal methods for compressing VGG16. The accuracy shows how well the compressed model performs on the training and validation dataset. In contrast, the loss graph shows how much the output of the compressed model differs from the ground truth labels during training and validation.

Based on Figure 7(a), the accuracy value for testing is higher at the beginning at 80% compared to training at 70%. However, the accuracy testing value ended with a lower value of 91% compared to training which was 93%. This indicates no sign of overfitting or other issues after compression of the model because both values are increased. The training loss starts at a higher value than the testing loss as depicted in Figure 7(b), which could indicate that the model does not fit the training data well initially [27]. However, it is essential to note that the value of the loss function can vary depending on the specific problem being solved and the choice of the loss function. As training progresses, both the training and testing losses decreased as expected, indicating that the model is learning to generalize well to new and unseen data. In this scenario, both the training and testing losses fell towards the end of training, it is a positive sign that the model is improving in terms of the model accuracy.

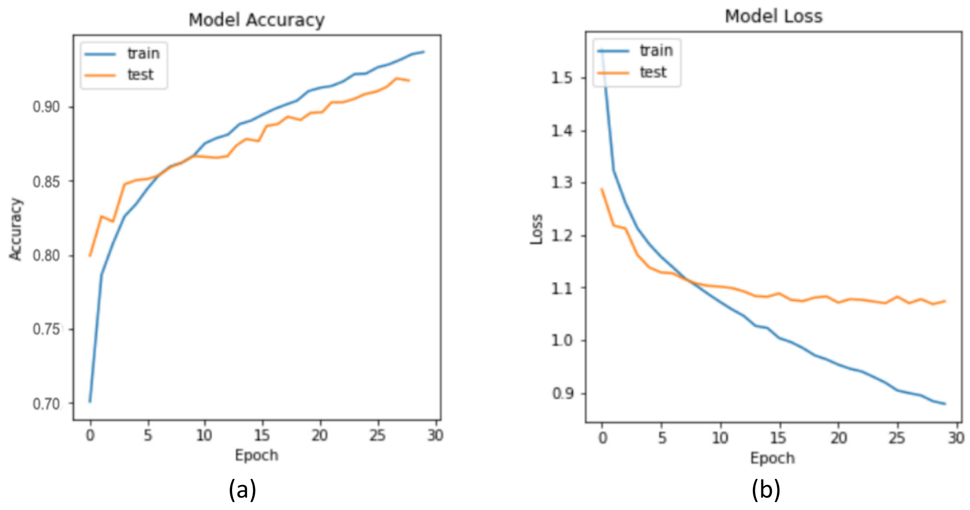

Fig. 7. (a) Graph of accuracy and (b) Graph of loss for multimodal method

Compared to other techniques, this multimodal technique can increase accuracy by up to 1% and simultaneously reduce the size of the model from 528.95MB to 90.22MB, or 5.9× compression rate. For the measurement of inference time, this combined technique reduces the processing time to one time from 0.92 seconds to 0.58 seconds.

4.2 Multimodal Method for Model Compression

This experiment is also continued by using two different types of iris datasets, UBIRIS.v1 and MMU Iris Database, to see if this combination of methods affects the performance of this model. Robustness in the iris recognition system refers to the ability of the system to accurately recognize iris patterns even in the presence of variations or distortions in the iris images. The iris recognition system should be able to handle different scenarios like changes in illumination, occlusion, blurring, rotation, and different imaging sensors. A robust iris recognition system should perform consistently even when the images are captured in different environments, at different times, and with different equipment. Hence, in this case, there are three publicly available databases used to examine the robustness of the compression model.

To further explain robustness in terms of the iris recognition system, let's look at this experiment. Table 2 shows the result when using three different iris datasets. The accuracy and inference time obtained while using the CASIA dataset is 91% and 0.58 seconds, respectively.

Table 2
Multimodal methods experiment on three iris datasets

Dataset	Accuracy (%)	Inference time (seconds)
CASIA	91.4	0.58
UBIRIS.v1	90.0	0.66
MMU Iris Database	90.9	0.75

The results of the experiment using the multimodal method on three iris datasets show that the performance matrix obtained is more or less the same. The accuracy obtained using CASIA, UBIRIS, and MMU Iris Dataset is 91.4%, 90%, and 90.9%, respectively. This model is robust because it can perform well on the three datasets even though using different characteristics.

In summary, the robustness of an iris recognition system is critical for ensuring accurate and reliable identification, especially in scenarios where environmental factors or medical conditions can affect the appearance of the iris.

4.3 Overall Deduction

This study aimed to investigate the impact of combination pruning and knowledge distillation in model compression on the performance of a CNN for iris recognition. The results show that compressing the model using a combination of pruning and knowledge distillation techniques can significantly reduce the number of parameters and computation while maintaining high accuracy on the test set. Based on the findings, we can infer that this model compression is an effective method for reducing the complexity of CNN models without affecting their performance. This finding is consistent with previous model compression research and emphasizes the potential benefits of compressing models for practical applications.

The accuracy obtained by different compression techniques can vary depending on the specific use case and the dataset being used. In general, it's difficult to say which technique gives the best accuracy across all cases. This study uses the pruning technique because this technique involves removing unnecessary weight from the trained model to reduce the size and computational cost. This can positively impact accuracy when the pruned model retains the most important weights. Knowledge distillation leads to high accuracy when the student model can effectively learn the representations learned by the teacher model. As a result, the multimodal technique can increase +1% accuracy by using the CASIA Iris V1 dataset.

Inference time may be simple enough for the model to process if it drops by only 0.34 seconds in an extensive dataset, indicating that it is highly optimized and efficient. This might happen when the model has previously experienced significant performance optimization and the data being processed is simple or low-dimensional. For instance, even if the model is highly tuned, the processing time may remain relatively high if the model is carrying out a simple classification task on a small dataset with few features.

Additionally, the model is already running in a high-performance computing environment with significant computational resources, and the 0.34 second decrease in inference time represents a relatively minor reduction in processing speed due to the already high level of optimization. While a

0.34 second reduction in inference time may appear minor, it can be a significant improvement in certain settings, such as real-time applications where speed is crucial. In other cases, such as batch processing of large datasets, significant improvements in processing speed may be required to attain optimal performance.

A comparison with the previous findings is shown in Table 3. According to the findings, accuracy is not affected by compression and can even increase to 1%, as opposed to CLIP-Q's accuracy increasing by 0.7% and ThiNet's accuracy value decreasing by 1%. While for compression rate, ThiNet obtained the highest value. It is crucial to remember that these findings are exclusive to ImageNet's image classification tasks and might not apply to other tasks or datasets. In comparison, this study focuses on image recognition and uses the iris dataset as input to the VGG16 model.

Table 3
Comparison with previous studies using VGG16 and MobileNet's model

Dataset	Model	Dataset	Accuracy (+/-)	Compression rate
ThiNet [28]	VGG16	ImageNet	-1.0%	16×
CLIP-Q [14]	MobileNet	ImageNet	+0.7%	7.6×
Proposed model	VGG16	CASIA	+1%	5.9×

In general, the best compression technique for a specific use case will depend on the particular requirements and constraints of that use case. It is often necessary to experiment with different techniques and compare the accuracy of the compressed models to determine the best approach. Previous studies also need to consider inference time to ensure that the model does not process an image for too long, affecting an application's performance.

5. Conclusions

The study has presented a new method for model compression of CNN that combines weight pruning and knowledge distillation in a single framework. The key contribution of this study is creating a lightweight model for an iris recognition system that can run on a smartphone. The model's accuracy can be increased to 1% by using a combination of compression methods in addition to maintaining the accuracy. The iris dataset was used as input to this model in this study, although most earlier studies employed CIFAR100 or ImageNet to pre-train the model before implementing the compression technique. The findings of this investigation will be carried forward to the following study, in which the compressed model will be embedded into the smartphone. The limitation of this study is that the researcher only used two combinations of compression models and there is still room to try combinations of other compression techniques. Generally, the findings demonstrated that the provided model compression techniques are effective in iris recognition compression and valuable in an iris biometric recognition system.

Acknowledgement
This research was not funded by any grant.

References
[1] Cheng, Yu, Duo Wang, Pan Zhou, and Tao Zhang. "A survey of model compression and acceleration for deep neural networks." *arXiv preprint arXiv:1710.09282* (2017).

[2] Chen, Xizi, Jingyang Zhu, Jingbo Jiang, and Chi-Ying Tsui. "Tight Compression: Compressing CNN Through Fine-Grained Pruning and Weight Permutation for Efficient Implementation." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 42, no. 2 (2022): 644-657. https://doi.org/10.1109/TCAD.2022.3178047

[3] Chen, Xizi, Jingyang Zhu, Jingbo Jiang, and Chi-Ying Tsui. "Tight compression: compressing CNN model tightly through unstructured pruning and simulated annealing based permutation." In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pp. 1-6. IEEE, 2020. https://doi.org/10.1109/DAC18072.2020.9218701

[4] Dupuis, Etienne, David Novo, Ian O'Connor, and Alberto Bosio. "Sensitivity analysis and compression opportunities in dnns using weight sharing." In *2020 23rd International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS)*, pp. 1-6. IEEE, 2020. https://doi.org/10.1109/DDECS50862.2020.9095658

[5] Oguntola, Ini, Subby Olubeko, and Christopher Sweeney. "Slimnets: An exploration of deep model compression and acceleration." In *2018 IEEE High Performance extreme Computing Conference (HPEC)*, pp. 1-6. IEEE, 2018. https://doi.org/10.1109/HPEC.2018.8547604

[6] Hu, Qingqiao, Siyang Yin, Huiyang Ni, and Yisiyuan Huang. "An end to end deep neural network for iris recognition." *Procedia Computer Science* 174 (2020): 505-517. https://doi.org/10.1016/j.procs.2020.06.118

[7] Nguyen, Kien, Clinton Fookes, Arun Ross, and Sridha Sridharan. "Iris recognition with off-the-shelf CNN features: A deep learning perspective." *IEEE Access* 6 (2017): 18848-18855. https://doi.org/10.1109/ACCESS.2017.2784352

[8] Jamil, Amirah Hanani, Fitri Yakub, Azizul Azizan, Shairatul Akma Roslan, Sheikh Ahmad Zaki, and Syafiq Asyraff Ahmad. "A Review on Deep Learning Application for Detection of Archaeological Structures." *Journal of Advanced Research in Applied Sciences and Engineering Technology* 26, no. 1 (2022): 7-14. https://doi.org/10.37934/araset.26.1.714

[9] Jalilian, Ehsaneddin, Heinz Hofbauer, and Andreas Uhl. "Iris Image Compression Using Deep Convolutional Neural Networks." *Sensors* 22, no. 7 (2022): 2698. https://doi.org/10.3390/s22072698

[10] Vyas, Ritesh. "Towards adept hand-crafted features for ocular biometrics." In *2020 8th International Workshop on Biometrics and Forensics (IWBF)*, pp. 1-6. IEEE, 2020. https://doi.org/10.1109/IWBF49977.2020.9107952

[11] Zhang, Yabo, Wenrui Ding, and Chunlei Liu. "Summary of convolutional neural network compression technology." In *2019 IEEE International Conference on Unmanned Systems (ICUS)*, pp. 480-483. IEEE, 2019. https://doi.org/10.1109/ICUS48101.2019.8995969

[12] Alqahtani, Ali, Xianghua Xie, and Mark W. Jones. "Literature review of deep network compression." In *Informatics*, vol. 8, no. 4, p. 77. MDPI, 2021. https://doi.org/10.3390/informatics8040077

[13] Fernandes Jr, Francisco E., and Gary G. Yen. "Pruning deep convolutional neural networks architectures with evolution strategy." *Information Sciences* 552 (2021): 29-47. https://doi.org/10.1016/j.ins.2020.11.009

[14] Tung, Frederick, and Greg Mori. "Deep neural network compression by in-parallel pruning-quantization." *IEEE transactions on pattern analysis and machine intelligence* 42, no. 3 (2018): 568-579. https://doi.org/10.1109/TPAMI.2018.2886192

[15] Qian, Liuchen, Yuzhuo Fu, and Ting Liu. "An Efficient Model Compression Method for CNN Based Object Detection." In *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, pp. 766-769. IEEE, 2018. https://doi.org/10.1109/ICSESS.2018.8663809

[16] Marinó, Giosué Cataldo, Alessandro Petrini, Dario Malchiodi, and Marco Frasca. "Deep neural networks compression: A comparative survey and choice recommendations." *Neurocomputing* 520 (2023): 152-170. https://doi.org/10.1016/j.neucom.2022.11.072

[17] Gheorghe, Ștefan, and Mihai Ivanovici. "Model-based weight quantization for convolutional neural network compression." In *2021 16th International Conference on Engineering of Modern Electric Systems (EMES)*, pp. 1-4. IEEE, 2021. https://doi.org/10.1109/EMES52337.2021.9484143

[18] Yang, Hong, Ya-sheng Zhang, Can-bin Yin, and Wen-zhe Ding. "Ultra-lightweight CNN design based on neural architecture search and knowledge distillation: A novel method to build the automatic recognition model of space target ISAR images." *Defence Technology* 18, no. 6 (2022): 1073-1095. https://doi.org/10.1016/j.dt.2021.04.014

[19] Cai, Gaoyuan, Juhu Li, Xuanxin Liu, Zhibo Chen, and Haiyan Zhang. "Learning and Compressing: Low-Rank Matrix Factorization for Deep Neural Network Compression." *Applied Sciences* 13, no. 4 (2023): 2704. https://doi.org/10.3390/app13042704

[20] Fang, Beihua, Yuanfu Lu, Zhisheng Zhou, Zhihui Li, Yuwen Yan, Linfeng Yang, Guohua Jiao, and Guangyuan Li. "Classification of genetically identical left and right irises using a convolutional neural network." *Electronics* 8, no. 10 (2019): 1109. https://doi.org/10.3390/electronics8101109

[21] Alaslni, Maram G., and Lamiaa A. Elrefaei. "Transfer learning with convolutional neural networks for iris recognition." *Int. J. Artif. Intell. Appl* 10, no. 5 (2019): 47-64. https://doi.org/10.5121/ijaia.2019.10505

[22] Singh, Arun, Akansha Pandey, Manik Rakhra, Dalwinder Singh, Gurasis Singh, and Omdev Dahiya. "An Iris Recognition System Using CNN & VGG16 Technique." In *2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 1-6. IEEE, 2022. https://doi.org/10.1109/ICRITO56286.2022.9965172

[23] Almutiry, Omar. "Efficient iris segmentation algorithm using deep learning techniques." *Journal of Electronic Imaging* 31, no. 4 (2022): 041202-041202. https://doi.org/10.1117/1.JEI.31.4.041202

[24] Alrifaee, Mustafa M. "A short survey of iris images databases." *Available at SSRN 3616735* (2020). https://doi.org/10.2139/ssrn.3616735
[25] Proença, Hugo, and Luís A. Alexandre. "UBIRIS: A noisy iris image database." In *Image Analysis and Processing–ICIAP 2005: 13th International Conference, Cagliari, Italy, September 6-8, 2005. Proceedings 13*, pp. 970-977. Springer Berlin Heidelberg, 2005. https://doi.org/10.1007/11553595_119
[26] Danlami, Muktar, Sapiee Jamel, Sofia Najwa Ramli, and Mustafa Mat Deris. "A framework for iris partial recognition based on Legendre wavelet filter." *International Journal of Advanced Computer Science and Applications* 10, no. 5 (2019). https://doi.org/10.14569/IJACSA.2019.0100506
[27] S. Klosterman, “Overfitting, underfitting, and the bias-variance tradeoff,” *Towards Data Science*, 2019.
[28] Luo, Jian-Hao, Jianxin Wu, and Weiyao Lin. "Thinet: A filter level pruning method for deep neural network compression." In *Proceedings of the IEEE international conference on computer vision*, pp. 5058-5066. 2017. https://doi.org/10.1109/ICCV.2017.541