IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 February 2023, accepted 1 April 2023, date of publication 6 April 2023, date of current version 17 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3265357

RESEARCH ARTICLE

Benchmarking Neural Network Compression Techniques for Ocular-Based User Authentication on Smartphones

ALI ALMADAN AND AJITA RATTANI, (Member, IEEE)
School of Computing, Wichita State University, Wichita, KS 67260, USA

Corresponding author: Ajita Rattani (ajita.rattani@wichita.edu)

This work was supported in part by the National Science Foundation (NSF) under Award no. 2129173, and in part by the Defense University Research Instrumentation Program (DURIP) from the Air Force Office of Scientific Research under Award no. 13106715.

ABSTRACT With the unprecedented mobile technology revolution, mobile devices have transcended from being the primary means of communication to an all-in-one platform. Consequently, an increasing number of individuals are accessing online services for e-commerce and banking via smartphones instead of traditional desktop computers. However, smartphones can be easily misplaced, lost, or stolen more often than other computing devices, thereby demanding effective user authentication mechanisms for device unlocking and secured transactions. Ocular biometrics has obtained significant attention from academia and industry because of its accuracy, security, and ease of use in mobile devices. Several studies have demonstrated the efficacy of deep learning models for ocular-based user authentication on smartphones. However, these high-performing models require enormous space and computational complexity due to the millions of parameters and computations involved. These requirements make their deployment on resource-constrained smartphones challenging. To this end, a handful of studies have been proposed for compact-size ocular-based deep-learning models to facilitate on-device deployment. In this paper, we conduct a thorough analysis of the existing neural network compression techniques applied as a standalone and in combination for ocular-based user authentication. Extensive experimental validation is performed on the two latest large-scale ocular biometric datasets collected using smartphones, namely, UFPR and VISOB 2.0 datasets. This study benchmarks the results of advanced compression techniques for further research and development in lightweight models for ocular-based user authentication on smartphones.

INDEX TERMS On-device AI, edge computing, ocular biometrics, mobile biometrics, neural network compression.

I. INTRODUCTION

Ocular biometrics consists of regions in the eye and those around it, such as the iris, conjunctival vasculature, and periocular region. Ocular biometrics has obtained significant attention from the research community and industry alike because of its accuracy, security, robustness against facial expressions, and ease of use in mobile devices [1], [2], [3], [4], [5], [6]. An added advantage of ocular biometrics over face biometrics is that it can be scanned and captured in the presence of a facial mask worn in case of pollution, surgery, or a pandemic like COVID-19. Further, visible light ocular biometric modalities can be acquired using many imaging devices, ranging from digital single-lens reflex (DSLR) cameras to ubiquitous smartphones equipped with RGB cameras. Both academia [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17] and industry, such as EyeVerify, Inc. [18] and Samsung [19], have utilized ocular-based user authentication in smartphones to unlock the device and perform secured transactions.

However, the operation in an unconstrained mobile environment can result in significant variations in the acquired

The associate editor coordinating the review of this manuscript and approving it for publication was Joey Tianyi Zhou.

samples due to a variety of factors, such as lighting conditions, distance, motion blur, occlusions (e.g., hair and glasses), front-facing imaging sensors and optical aberrations, and other imaging issues, such as inaccurate white balance and exposure metering. Consequently, the performance of the system may degrade in a mobile environment. With advances in deep learning, deeply coupled autoencoders and convolutional neural network (CNN) architectures, such as VGG-19, ResNet-50, and MobileNet have been used for enhanced performance of ocular-based user authentication on smartphones [8], [20], [21], [22], [23], [24].

Further, pairwise filters and siamese neural networks have been employed for multi-class classification and multi-task learning from ocular images captured via mobile devices [25], [26]. Large-scale VISOB [21], [27] and UFPR [25], [26] datasets that consist of RGB ocular images acquired using smartphones have been assembled for promoting research and development in this field.

However, these high-performing deep learning models have enormous space and high computational requirements due to the millions of parameters and computations involved [28], [29], [30]. The high computational cost of deep learning models makes deployment challenging in resource-constrained mobile environments [29], [31], [32], [33]. The size and computational cost of the deep learning models should be low for real-time and frequent on-device biometric authentication, for instance, in the case of unlocking the smartphone device. A faster authentication mechanism is required for enhanced *user experience*, but over-parameterized models would be slower to execute. In addition, compact models would result in *battery-friendly biometric authentication*.

Common compression techniques, such as those based on knowledge distillation (KD) [34], pruning [35], [36], and quantization [37], [38] have been used to reduce the size of deep learning models for the general image classification task. The KD technique is the process of transferring knowledge from a large model to a smaller one by minimizing the loss. Rational KD (RKD) [39] and Probabilistic Knowledge Transfer (PKT) [40]) are popular variants of KD. Neural network pruning techniques reduce the size of a deep neural network by removing its weight connections. These pruning techniques can be classified into structured [41] and unstructured pruning [30]. Structured pruning techniques aim to remove structures like filters and layers, while unstructured pruning techniques set individual weight connections to 0. They result in sparse matrices that require special sparse manipulation libraries at inference time. Quantization techniques reduce the number of bits needed to represent model parameters. The two popular types of quantization can be static and dynamic quantization. Static quantization quantizes the weights and activations of the model [42] offline using a calibration set. Dynamic quantization quantizes the weights ahead of time, but dynamically quantizes the activations during inference, as described in [43].

A handful of studies have been proposed for compressing deep learning models (reducing the size and complexity) for *ocular-based mobile user authentication* [2], [6], [28], [44]. In our recent study in [28], we evaluated and compared KD and five unstructured pruning techniques (namely, Global Magnitude, Layerwise Magnitude, Global Gradient Magnitude, Layerwise Gradient Magnitude, and random pruning) on the ResNet-50 model for smartphone-based user authentication using ocular biometrics on VISOB 2.0 and UFPR datasets. The study suggested the efficacy of KD over unstructured pruning techniques in terms of verification accuracy. To further advance the state-of-the-art, this paper aims to analyze thoroughly and compare standalone neural network compression techniques and their combinations for obtaining lightweight models to facilitate on-device mobile user authentication using ocular biometrics. To this aim, the following **research questions (RQs)** are asked and addressed for the first time for ocular-based user authentication models in this study:

- **RQ1:** When using KD to obtain compact models for ocular biometrics, how well do different KD techniques generalize across different architectures for teacher and student models?
- **RQ2:** What are the advantages and disadvantages of structured pruning over unstructured pruning for ocular biometrics?
- **RQ3:** Can the combination of compression techniques provide any additional benefit (in terms of performance and size) over standalone methods for ocular biometrics?

The main **contributions** of this work over [28] are summarized as follows (also illustrated in Figure 1):

1) Comparative evaluation of the advanced KD-based techniques (the vanilla KD [34], RKD [39], and PKT [40]) for compressing multiple variants of VGG [45] and ResNet architectures [46] for ocular-based user authentication. The impact of architectural changes between teacher and student models is also evaluated for all the KD techniques. The fine-tuned VGG and ResNet architectures are widely adopted for mobile ocular biometrics [21], [25].
2) Comparison of neural network structured and unstructured pruning techniques using L2-norm-based channel pruning [41], and the Layerwise Gradient Magnitude pruning [30] techniques, respectively, for ocular-based user authentication. These pruning techniques are applied to variants of VGG and ResNet models (trained for ocular recognition) to better understand the trade-off between the accuracy and size at different compression ratios.
3) Evaluation of the combinations of neural network compression techniques resulting in sequences: Pruning-KD, KD-Pruning, Pruning-KD-Quantization, and KD-Pruning-Quantization. These sequences are based on the order of applied compression techniques. These

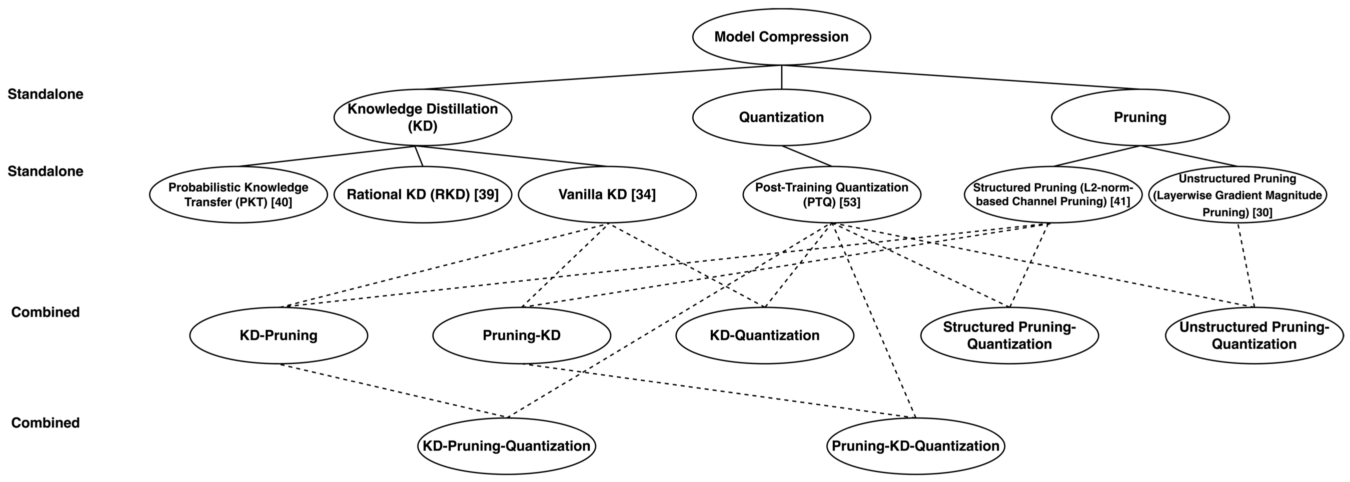
Model Compression
Standalone
Knowledge Distillation (KD)
Quantization
Pruning
Standalone
Probabilistic Knowledge Transfer (PKT) [40]
Rational KD (RKD) [39]
Vanilla KD [34]
Post-Training Quantization (PTQ) [53]
Structured Pruning (L2-norm-based Channel Pruning) [41]
Unstructured Pruning (Layerwise Gradient Magnitude Pruning) [30]
Combined
KD-Pruning
Pruning-KD
KD-Quantization
Structured Pruning-Quantization
Unstructured Pruning-Quantization
Combined
KD-Pruning-Quantization
Pruning-KD-Quantization

FIGURE 1. **This figure summarizes the individual and combined compression techniques evaluated and compared in this work.**

combinations are compared with the standalone neural network compression techniques.

4) Thorough subject-independent evaluation, where the subjects do not overlap between training and test sets [21], [28], of all the compressed models is performed on the two latest and large-scale UFPR [25] and VISOB 2.0 [21] mobile ocular biometric datasets.
5) Computation of the run-time performance of the baseline models along with compressed models in terms of extraction time (ET), theoretical speed-up, and memory usage when deployed on an Apple iPhone 14 smartphone.

This paper is organized as follows: Section II discusses the prior work on compact models for ocular-based user authentication on mobile devices. Section III discusses the state-of-the-art network pruning, KD, and quantization techniques used in this study. Datasets and experimental protocol are discussed in Sections IV, and V. Results are analyzed in Section VI. Key findings are listed in Section VII. Conclusions are drawn in Section VIII.

II. PRIOR WORK

In this section, we discuss existing studies on developing lightweight (compact) CNN models for ocular-based user authentication on smartphones.

Boutros et al. [44] proposed a lightweight DenseNet-20 model with only 1.1M trainable parameters obtained via KD. Experiments performed on the VISPI dataset showed that DenseNet-20 trained using KD outperformed the same model trained without KD with EER reduction from 8.36% to 4.54%. DenseNet-121, DenseNet-169, and DenseNet-201 models containing 7.1, 12.6, and 18.2M of trainable parameters, respectively, were used as a teacher to distill the knowledge to a student model, i.e., DenseNet-20.

In another study, Boutros et al. [47] proposed a novel template-driven KD approach that optimized the distillation process so that the student model learned to produce templates similar to those produced by the teacher model. This was obtained by introducing an additional loss function over the original KD loss operating on the feature extraction layer. The proposed distillation method outperformed the model distilled using the conventional (vanilla) KD approach.

Reddy et al. [2] proposed patch-based OcularNet, a CNN model that used patches from the eye images for user authentication in smartphones. For the OcularNet model, six registered overlapping patches were extracted from the ocular region, and a small CNN was trained for each patch to extract feature descriptors. The verification performance of the proposed OcularNet having 1.5M parameters was compared to the popular ResNet-50 model that had 23.4M parameters. When trained on the VISOB dataset, the proposed OcularNet model obtained equivalent performance over ResNet-50 in the subject-independent verification setting.

In another study by Reddy et al. [6], the authors proposed a customized version of the MobileNet-v2 architecture obtained by removing the last convolutional layers from the original implementation without affecting the accuracy while reducing the model size by 3.4x compared to the original MobileNet-v2 and 36x compared to the popular ResNet-50 for mobile ocular authentication.

Recently, Almadan and Rattani [28] evaluated and compared two neural network compression techniques, i.e., KD, and five unstructured pruning-based methods at six compression rates. These two compression techniques were applied to the ResNet-50 model for ocular-based mobile user authentication as a standalone. The KD experiments were performed with ResNet-50 as the teacher and ResNet-8, ResNet-20, MobileNet-V2, MobileNet-V3, and ShuffleNet-V2 as the student models. The five unstructured pruning techniques applied to the ResNet-50 model, include Global Magnitude, Layerwise Magnitude, Global Gradient Magnitude, Layerwise Gradient Magnitude, and random pruning. In addition, the real-time feature extraction time of the compressed models was measured on five mobile devices.

Algorithm 1 Iterative Pruning With Fine-Tuning Method

```
Require: $N$, the number of iterations of pruning, and
         $x$, the dataset on which to train and fine-tune
1: $W \leftarrow initialize()$
2: $W \leftarrow trainToConvergence(f(x; W))$
3: $M \leftarrow 1^{|W|}$
4: for $i$ in 1 to $N$ do
5:     $M \leftarrow prune(M, score(W))$
6:     $W \leftarrow fineTune(f(x; M \odot W))$
7: end for
8: return $M, W$
```

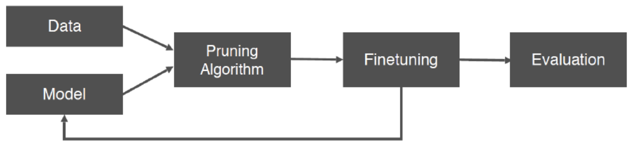

FIGURE 2. The steps involved in pruning a neural-network architecture.

III. NEURAL NETWORK COMPRESSION TECHNIQUES

This section reviews three compression techniques for obtaining lightweight (compact) models, namely, network pruning, KD, and quantization.

A. NETWORK PRUNING TECHNIQUES

The neural network pruning technique reduces the size of a deep neural network by removing weight connections [48]. The weight connection removal is done using criteria to evaluate the importance of a connection relative to the whole network. The primary purpose is to generate a smaller network and obtain an accuracy identical to the initial network in an ideal case.

Algorithm 1 shows the steps involved in pruning an initially trained model $f(x; W)$ where W denotes the neural network parameters, M is a binary mask with certain parameters to 0, and $\odot$ is the elementwise product operator. In detail, after the network $(f(x; W))$ has been trained until convergence, each parameter is given a score based on criteria. This score indicates the importance of the associated weight connections. The network is pruned by removing the weight connections obtaining the lowest score ranking. Pruned models generally undergo a fine-tuning step (shown in Figure 2) to recover the performance loss during the pruning procedure by retraining the unpruned connections. Usually, the pruning and fine-tuning steps follow an iterative procedure until a desired performance is obtained.

The performance of the pruning technique is typically evaluated at various compression rates. The Compression Rate, known as the Compression Ratio (CR)(%) metric, is defined as the percentage fraction of initial parameters to prune. This can also be defined as the compressed size divided by the new size: $\frac{\text{compressed size}}{\text{new size}}$.

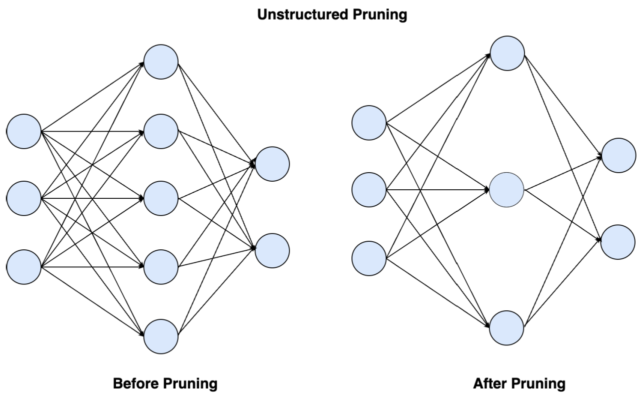

FIGURE 3. An illustration of the network before (left) and after unstructured pruning (right). Unstructured network pruning removes connections at the neuron level.

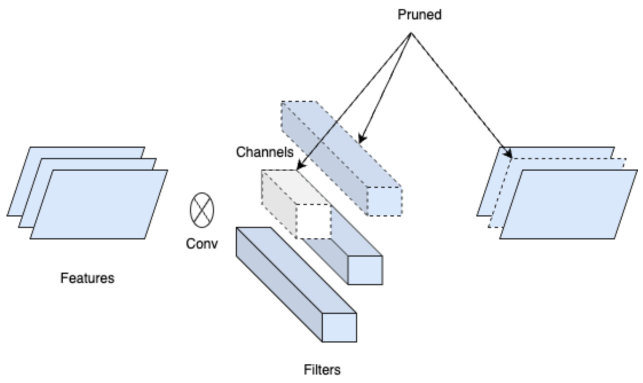

FIGURE 4. An illustration of structured pruning where the connections are removed at filter and channel level. The removed filters and channels are shown with dashed lines.

Further, pruning techniques can be classified into unstructured and structured pruning (shown in Figures 3 and 4), respectively.

1) UNSTRUCTURED PRUNING

In brief, the unstructured pruning approach prunes away individual weight connections targeted for removal by setting them to zero values. Thus, this technique results in a sparse weight matrix. Evaluating weight connections to be pruned away can depend on different criteria, for instance, pruning the weights with the lowest absolute value or the lowest absolute value of the product of the weight and its gradients [30]. However, unstructured pruning methods have the drawback of requiring a particular framework and hardware capable of accelerating the computation of sparse matrices [49]. In other words, unstructured pruning methods result in sparse weight matrices and thus leading to inefficiency in speedup and compression on CPUs and GPUs. In this study, we evaluated Layerwise Gradient Magnitude-based unstructured pruning [30]. This method compares scores locally and prunes a fraction of the parameters with the lowest scores within each structural sub-component of the network (e.g., layers).

2) STRUCTURED PRUNING

In contrast, structured pruning aims to prune structures entirely by removing weight matrices for an entire channel or a filter, as illustrated in Figure 4. Unlike unstructured pruning, structured pruning can run with the same hardware and framework as the original network due to its capability to change the structure. For channel pruning, the importance of the channel is measured with a related score.

One of these methods is based on computing the L2-norm of the filters. The L2 norm-based importance score is computed as $\mathcal{I}_{norm} = \|\mathbf{W}_i\|_2$, where $\mathbf{W}$ denotes the weights and i is the i-th output channel of the network. Therefore, the filters with smaller norms are targeted to be pruned since they result in a smaller magnitude of the output feature map. This computation aids in removing redundant channels by identifying the important channels. As a result, this process reduces the model size and speeds up the inference time [50]. In this study, we evaluated the structured pruning technique based on L2-norm-based channel pruning.

B. KNOWLEDGE DISTILLATION

KD is a method of transferring the knowledge learned by a larger model in size (i.e., teacher) to a smaller model (i.e., student) [34]. The key idea behind the transfer process is that the student mimics the soft probabilities outputs of a trained teacher network by minimizing a loss function. The three essential elements of a KD approach are knowledge, a distillation algorithm, and a teacher-student architecture [51]. Knowledge refers to the learned weights where the sources of knowledge can be the logits or the weights of intermediate layers.

1) CONVENTIONAL KD

In the conventional (vanilla) KD algorithm [34], the teacher model assumes the knowledge to be transferred as a learned mapping from inputs to outputs; thus, the learning is based on a point-wise approach. Then, the teacher model transfers the knowledge to the student by training it with the teacher's outputs referred to as soft labels. Figure 5 shows the generic architecture of the KD approach using a teacher-student model consisting of the three essential elements (knowledge, a teacher-student network, and an algorithm for transferring knowledge). Concretely, given an input image x with a label y in a dataset represented by $\{x_i, y_i\}, i = 1, \ldots, n$, the teacher model outputs a vector of scores $s^T(x)$ to be converted into a class probability distribution [34]:

$$p^T_\tau = \frac{e^{s^T(x)/\tau}}{\sum_j e^{s^S(x)/\tau}} \tag{1}$$

where τ is a hyper-parameter that softens the proprieties.

After a student model returns probabilities, p^T_τ, the distillation loss is minimized with:

$$L = (1-\lambda)L_{ce} + \tau^2 \lambda L_{kD}, \tag{2}$$

where $L_{ce} = -yi \sum_i logp(yi|xi)$ is the cross-entropy loss with student network prediction $p(yi|xi) = \text{softmax} f(xi)$ and f(.) is the logit of the network. L_{KD} is the KD loss defined as:

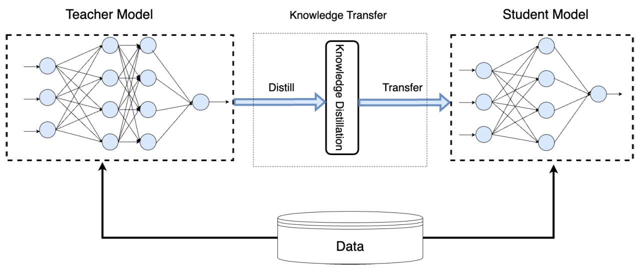

FIGURE 5. Generic architecture of KD using a teacher-student model.

$$L_{KD} = \sum_i KL(p^T_\tau(y_i|x_i)||p^S_\tau(y|x_i)) \tag{3}$$

Here, $p_\tau(y|x) = \text{softmax}~(f(x)/\tau)$ and $p_\tau(y|x) = \text{softmax}~(f(x)/\tau)$ are the smoothened student's and teacher's prediction, respectively. The smoothness of the prediction is regulated by the temperature term τ, and KL is defined as the KL divergence. The hyperparameter λ decides the contribution of the KD loss.

2) RKD AND PKT

RKD [39] aims to transfer structural knowledge through mutual relations of data examples in the output presentation of the teacher. Unlike traditional approaches, it computes a relational potential function for each n-tuple of data examples. The information is transferred from the teacher to the student through the learned function that penalizes the difference between the teacher and the student models using a loss function. Thus, the knowledge transferred from the teacher to the student model relies on structure-to-structure rather than point-to-point with two distillation losses: distance and angel-wise distillations. The PKT [40] method models the data samples in the feature space as a probability distribution. The knowledge transferred minimizes the probability distributions between the teacher and student models. Therefore, the knowledge transfer mechanism in the PKT approach allows the student network to learn and match the probability distribution of the teacher model's feature representation rather than the actual outputs of the teacher.

This study evaluated three existing KD approaches, namely, the vanilla KD [34], RKD [39], and PKT [40].

C. QUANTIZATION

Quantization techniques can reduce the size of deep neural networks and speed up inference latency [42]. Quantization provides alternative formats to replace the dominant format, i.e., 32-bit single-precision floating-point for a deep learning model. As a result, the size of quantized models is smaller compared to floating-point-based models since the weights are stored as low-bit width integers. Thus, quantization is a method aiming to map input values with a large continuous set to output values with a smaller finite set using techniques such as rounding and truncation (shown in Figure 6).

Quantization methods can be classified into two different classes: static and dynamic quantization methods. Unlike static quantization, where the weights and activations of the network are both quantized to integers in an offline manner, only the weights of the model are quantized to integers for dynamic quantization. However, the activations are quantized dynamically during the run-time inference. Static quantization requires the use of a representative dataset to calculate the quantization scale, which is a symmetric range mapping technique used to map an input to a range prior to inference, whereas dynamic quantization does not require calibration data because the quantization scale is calculated during inference.

1) QUANTIZATION AWARE TRAINING (QAT) AND POST-TRAINING QUANTIZATION (PTQ)

In QAT [52], a pre-trained model is quantized, and then the model is fine-tuned by using the training data. Fine-tuning allows the model to restore the accuracy degradation caused by quantifying the weights. On the hand, PTQ [53] is an effective model compression technique that can directly quantize neural network models without fine-tuning. The computation of the float32 ranges occurs during the calibration step of quantization. The clipping range of weights is determined at quantization time, but it follows different approaches for activations. In dynamic PTQ, the range for activations is calculated at run-time, whereas the range is computed in advance by passing representative data in static PTQ.

PTQ is a high-speed method for neural network quantization compared to QAT because it adjusts the weights without fine-tuning. Further, QAT requires sufficient training data, whereas PTQ can also operate with limited or unlabeled data to appropriately represent the activation values [52], [53], [54]. QAT technique, in general, performs better compared to the PTQ technique. But, the QAT might not be applicable to real-world scenarios where the data is limited. Therefore, the PTQ has recently received a lot of attention due to its high practicality [55].

In this study, we applied the PTQ with dynamic quantization, where the clipping range was fixed for all inputs. The range for each activation was computed on the fly at run-time. The aim was to compress the models and compare the performance with the pre-quantized versions of the same models. We chose to quantize the models with PTQ over QAT since PTQ can operate with limited training data, and a recent study suggests a very minimal performance gain with QAT over PTQ [56].

IV. DATASETS

In this section, we describe the datasets (UFPR and VISOB 2.0) used in this study. Sample images from these both datasets are shown in Figure 7.

A. UFPR

UFPR [25] is the latest ocular biometric dataset containing samples from 1, 122 subjects with a total of 33, 660 images

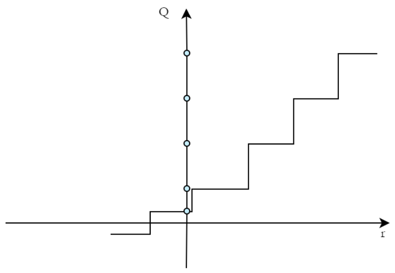

FIGURE 6. Real values in the continuous domain *r* are mapped to discrete values in the quantized domain *Q*, marked with the blue bullets.

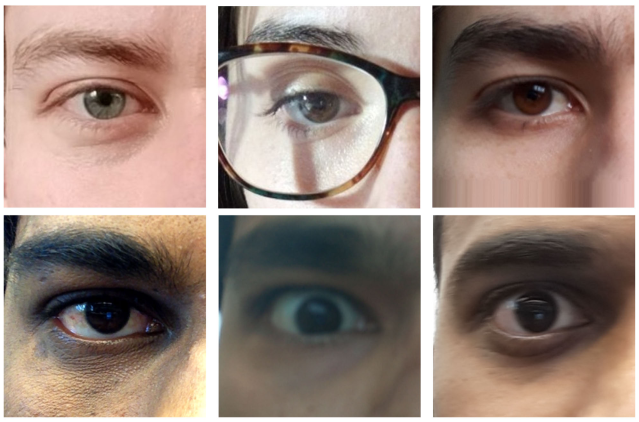

FIGURE 7. Sample ocular images from the UFPR [25] (top row) and VISOB 2.0 [21] (bottom row) datasets.

acquired in 3 sessions by 196 different mobile devices. The data is captured across race, age, and gender. The gender distribution of the subjects is (53.65%) male and (46.35%) female, and approximately 66% of the subjects are under 31 years old. The eye corners of the ocular images are annotated with 4 points per image (inside and outside eye corners) to normalize the ocular region across scales and rotation angles.

B. *VISOB* 2.0

VISOB 2.0 [21] is the second version of VISOB 1.0 dataset [27] used in the IEEE ICIP 2016 competition that facilitates subject-independent analysis. This publicly available dataset consists of a stack of eye images captured using the burst mode via two mobile devices: Samsung Note 4 and OPPO N1. During the data collection, the volunteers were asked to take their selfie images in two visits, 2 to 4 weeks apart from each other. At each visit, the selfie-like images were captured using the front-facing camera of the mobile devices under three lighting conditions (daylight, office light, and dark light) and two sessions (about 10 to 15 minutes apart). The stack consisting of five consecutive eye images was extracted from the stack of full-face frames selected such that the correlation coefficient between the center frame and the remaining four images is greater than 90%.

V. EXPERIMENTAL PROTOCOL AND IMPLEMENTATION DETAILS

In this section, we discuss the experimental protocol used in this study. We added batch normalization, dropout, and a fully connected layer after the last convolutional layer to all models. For all the models, the weights were initialized with Xavier weight initialization. The deep feature vector of size 512 was extracted from the fully connected layer of these trained models for each ocular image. The matching scores were calculated using the cosine similarity metric [57] between the deep features extracted from a pair of templates and test images. The average of the scores from the five template images per test image was used as the final score for the subject identity verification.

A. EXPERIMENTS CONDUCTED

We summarize the experiments performed in this paper as follows:

- **Exp #1:** Evaluation of the performance of the baseline models (ResNet-50, ResNet-20, VGG-13, VGG-11, and VGG-8) trained from scratch on 100 subjects from UFPR [25] and VISOB 2.0 [21] datasets (see Table 1). Evaluation of the quantized versions of the same baseline models obtained using the PTQ method (Table 2).
- **Exp #2:** Evaluation and comparison of the three KD approaches (the vanilla KD [34], RKD [39], and PKT [40]) using ResNet-50, VGG-13, and ShuffleNet-V1 as the KD teacher models and ResNet-20 and VGG-8 as the student models. This experiment assessed the performance of the different KD techniques for subject-independent user authentication (see Table 3).
- **Exp #3:** Comparative evaluation of the performance of structured pruning (based on L2-norm-based channel pruning [41]) and unstructured pruning (based on Layerwise Gradient Magnitude [30] pruning) using the large models: ResNet-50, VGG-13, and VGG-11 (Tables 4 and 5). The pruned models are further quantized using the PTQ technique (see Tables 9 and 10). This analysis explains the trade-off between the performance and size of the baseline and pruned models obtained using structured and unstructured pruning techniques. In addition, we also evaluated the performance vs. size trade-off after applying quantization of the pruned models.
- **Exp #4:** Evaluation of the combinations of neural network compression techniques (vanilla KD, structured pruning, and PTQ-based quantization) for mobile ocular-based user authentication. The combinations evaluated include; KD-Pruning (Table 6), Pruning-KD (Table 7), KD-Quantization (Table 8), Pruning-Quantization (Table 9) and (Table 10) for structured and unstructured pruning, respectively. The quantized versions of combinations of the pruned and quantized versions of ResNet-20 and VGG-8 models, resulting in KD-Pruning-Quantization and Pruning-KD-Quantization were also evaluated (Table 11).

B. IMPLEMENTATION DETAILS ON MODEL COMPRESSION

We discuss the implementation details of the compression techniques applied to the models used in this study.

1) INDIVIDUAL COMPRESSION TECHNIQUES

- **KD:** The ResNet-50, VGG-13, and ShuffleNet-V1 were used as teacher models for all KD experiments based on the vanilla KD, PKT, and RKD approaches. The two KD student models were ResNet-20 and VGG-8. The training process was based on offline distillation, where the teacher model was first pre-trained on 100 subjects from the two datasets, and then the gained knowledge was transferred and guided from the teacher model to the student model. The distillation training process was performed for 100 epochs.
- **Structured and Unstructured Pruning:** The two pruning techniques were applied separately to the baseline models trained for 100 epochs. Then, multiple pruned versions were generated from the baseline models using the L2-norm-based channel pruning (structured pruning) and Layerwise Gradient Magnitude pruning (unstructured pruning) at the following compression rates: 10, 30, and 50%. The pruned models were fine-tuned iteratively for 100 epochs for accuracy recovery.
- **Quantization:** The pre-trained compact models (VGG-8 and ResNet-20) and larger models (ResNet-50, VGG-13, and VGG-11) were quantized dynamically using the PTQ technique. The pre-trained models were further quantized to 8-bit precision for CPU execution to further reduce memory consumption and computation.

2) COMBINED COMPRESSION TECHNIQUES

- **KD-Pruning:** This combination indicates the knowledge-distilled student models (ResNet-20 and VGG-8) were further pruned at different compression rates using L2-norm-based structured pruning technique. The knowledge-distilled compact models were generated using the vanilla KD approach for 100 epochs, with ResNet-50 as the teacher model using Adam optimizer and cross-entropy loss function.
- **Pruning-KD:** ResNet-20 and VGG-8 were pruned with L2-norm-based channel pruning technique. The pruned models were distilled (compressed) using the vanilla KD approach, with ResNet-50 as the teacher model for 100 epochs using Adam as an optimizer and cross-entropy loss function.
- **Pruning-Quantization:** ResNet-50, VGG-13, and VGG-11 pruned using L2-norm-based structured channel pruning, and Layerwise Gradient Magnitude-based unstructured pruning techniques were further quantized using the PTQ technique. The pruned models were fine-tuned for 100 epochs before being quantized.

- **KD-Quantization:** The compact models (ResNet-20 and VGG-8) distilled using ResNet-50 as the teacher model were further quantized using the PTQ technique. The models were distilled using the vanilla KD approach for 100 epochs before being pruned.
- **KD-Pruning-Quantization:** The student models (ResNet-20 and VGG-8) distilled using the vanilla KD approach with ResNet-50 as the teacher model were further pruned using the L2-norm-based structured channel pruning technique. Finally, PTQ was performed on the pruned version of the models. The KD process was performed for 100 epochs.
- **Pruning-KD-Quantization:** The models (ResNet-20 and VGG-8) were initially pruned with L2-norm-based structured channel pruning at multiple CR without any training. The pruned models were distilled using the vanilla KD technique with ResNet-50 as the teacher model at different compression rates. The distilled models were quantized using the PTQ technique.

3) MODEL DEPLOYMENT ON SMARTPHONE

To evaluate the run-time performance of the model on the edge device, we deployed the deep-learning models, both uncompressed and compressed versions, on Apple iPhone 14. The PyTorch framework was used for the development and training of all the models. In order to run and deploy a PyTorch-based deep model in a mobile environment, a conversion of the model to TorchScript [58] is required for execution in a C++ environment. Therefore, we used TorchScript to further optimize and deploy the deep learning models on a real Apple iPhone 14 device. The memory usage of the deployed models on the device was computed using the Apple Xcode integrated development environment (IDE), which has the Clang compiler as the default compiler [59]. Xcode contains development tools for several operating systems, including tvOS, macOS, iOS, and iPadOS. Coding in Xcode is typically done using the Swift programming language. iOS measures the on-device app's memory usage by counting the number of memory pages that are being used and multiplying that number by the page size, which is typically 16 KB [60].

VI. EXPERIMENTAL RESULTS

In this section, we evaluate the subject-independent verification performance of the deep learning models on UFPR and VISOB 2.0 datasets before and after applying compression techniques, both standalone and combined. The choice of the ResNet and VGG model architectures is motivated by the fact that most of the existing studies have used these models for benchmarking baseline results on mobile ocular datasets [21], [25].

A. BASELINE PERFORMANCE OF THE MODELS

Table 1 demonstrates the subject-independent user verification performance of the baseline CNN models used in this study. The models are evaluated using the Equal Error Rate (EER), Genuine Match Rates (GMR) at two different False Match Rates (FMRs), and the Area Under Curve (AUC). As can be seen, ResNet-50 and VGG-13 (large-sized models) obtained the lowest average EER of about 6.24% and 6.57% for the two datasets: UFPR and VISOB 2.0, respectively. In terms of GMR, ResNet-50 obtained the highest GMR on both datasets. For instance, the ResNet-50 model obtained an average GMR of 33.63% and 52.87% at FMR of 0.0001 and 0.001, respectively, across the two datasets. The EER obtained by the compact models trained from scratch, i.e., ResNet-20 and VGG-8 averaged on both datasets were 11.59% and 9.00%, respectively. However, the GMR of the small-sized models (ResNet-20 and VGG-8) on the two datasets were in the range of [8.87-25.35]% at FMR of 0.0001 and an average of 31.56% was obtained at FMR of 0.001.

To understand the trade-off between the performance and model size, we measured the model size by the no. of parameters (MB) as indicated in Table 1. With ResNet-20 being the most compact model, it obtained an increase of 5.69% in EER when compared to the large model ResNet-50. It is worth noting that ResNet-20 is 80x smaller than ResNet-50. The performance of the VGG-8, a compact model, reduced with an EER difference of 2.08% and 1.93% compared to VGG-13 and VGG-11. VGG-8 is 2.30x smaller than both models (VGG-13 and VGG-11).

InTable 2, we show the performance of the quantized version of the baseline models obtained using the PTQ technique. Overall, the quantized models obtained a size reduction of about 1.02× with a minimal drop in performance of less than 1.00% EER.

TABLE 1. Baseline: Subject-independent analysis of baseline models on UFPR [25] and VISOB 2.0 [21] datasets.

Dataset	CNN	Model Size (MB)	EER (%)	GMR (%) @ FMR		AUC
				0.0001	**0.001**	
UFPR	ResNet-50 (large-size)	97.39	6.15	27.92	39.50	0.99
	ResNet-20 (compact)	1.233	11.62	15.94	27.49	0.94
	VGG-13 (large-size)	38.46	6.99	26.19	38.97	0.98
	VGG-11 ((large-size)	37.71	7.85	26.51	38.47	0.98
	VGG-8 (compact)	16.45	9.33	25.34	37.48	0.95
VISOB 2.0	ResNet-50 (large-size)	97.39	5.65	39.33	66.23	0.99
	ResNet-20 (compact)	1.233	11.55	8.87	21.68	0.93
	VGG-13 (large-size)	38.46	6.83	39.09	58.78	0.99
	VGG-11 (large-size)	37.71	6.91	37.24	62.44	0.98
	VGG-8 (compact)	16.45	8.66	22.12	39.57	0.96

TABLE 2. Quantization: Subject-independent analysis of quantized baseline models on UFPR [25] and VISOB 2.0 [21] datasets. The PTQ method was used.

Dataset	Quantized Model	Model Size (MB)	EER (%)	GMR (%) @ FMR		AUC
				0.0001	**0.001**	
UFPR	ResNet-50 (large-size)	95.08	7.66	26.65	38.41	0.98
	ResNet-20 (compact)	1.162	12.54	15.32	27.91	0.93
	VGG-13 (large-size)	37.88	7.49	26.15	38.01	0.97
	VGG-11 ((large-size)	37.18	8.11	26.15	38.01	0.96
	VGG-8 (compact)	15.87	10.12	24.57	35.79	0.92
VISOB 2.0	ResNet-50 (large-size)	95.08	5.90	38.93	65.26	0.98
	ResNet-20 (compact)	1.162	11.65	8.48	21.38	0.93
	VGG-13 (large-size)	37.88	7.54	38.05	58.37	0.98
	VGG-11 ((large-size)	37.18	7.17	36.88	62.37	0.98
	VGG-8 (compact)	15.87	8.36	21.66	38.70	0.96

TABLE 3. KD: Subject independent analysis of the compact models generated using KD methods on UFPR [25] and VISOB 2.0 [21] datasets.

Dataset	Student	Model Size (MB)	Teacher	KD	EER (%)	GMR (%) @ FMR		AUC
						0.0001	**0.001**	
UFPR	ResNet-20 *(Baseline)*	1.233	-	-	11.62	15.94	27.49	0.94
	ResNet-20	1.233	ResNet-50	Vanilla	8.20	20.24	31.83	0.99
			VGG-13		8.47	14.71	27.02	0.99
			ShuffleNet-V1		9.13	19.06	30.69	0.99
			ResNet-50	PKT	7.70	18.97	31.49	0.99
			VGG-13		8.44	17.03	27.70	0.99
			ShuffleNet-V1		8.80	18.93	30.19	0.99
			ResNet-50	RKD	7.83	19.86	33.85	0.99
			VGG-13		9.13	13.91	25.93	0.99
			ShuffleNet-V1		8.99	19.52	31.58	0.99
	VGG-8 *(Baseline)*	16.45	-	-	9.33	25.34	37.48	0.95
	VGG-8	16.45	ResNet-50	Vanilla	7.71	21.29	34.70	0.99
			VGG-13		7.57	20.49	33.35	0.98
			ShuffleNet-V1		8.50	20.57	31.66	0.97
			ResNet-50	PKT	7.11	24.07	38.24	0.98
			VGG-13		7.85	20.70	34.23	0.98
			ShuffleNet-V1		8.14	20.87	31.58	0.97
			ResNet-50	RKD	7.49	22.05	35.58	0.99
			VGG-13		7.48	21.63	34.57	0.98
			ShuffleNet-V1		8.53	21.12	32.38	0.97
VISOB 2.0	ResNet-20 *(Baseline)*	1.233	-	-	11.55	8.87	21.68	0.93
	ResNet-20	1.233	ResNet-50	Vanilla	9.35	25.28	43.52	0.98
			VGG-13		9.50	27.54	46.72	0.98
			ShuffleNet-V1		11.37	13.65	30.97	0.96
			ResNet-50	PKT	8.89	25.80	45.68	0.98
			VGG-13		9.32	26.77	46.63	0.98
			ShuffleNet-V1		10.82	15.92	33.62	0.96
			ResNet-50	RKD	9.13	27.24	46.46	0.98
			VGG-13		8.83	29.56	47.85	0.98
			ShuffleNet-V1		10.24	12.03	29.76	0.95
	VGG-8 *(Baseline)*	16.45	-	-	8.66	22.12	39.57	0.96
	VGG-8	16.45	ResNet-50	Vanilla	6.99	40.69	63.34	0.99
			VGG-13		7.28	34.48	54.67	0.98
			ShuffleNet-V1		9.73	16.76	35.00	0.97
			ResNet-50	PKT	8.00	29.85	51.55	0.98
			VGG-13		7.74	33.48	54.54	0.98
			ShuffleNet-V1		10.38	13.85	33.94	0.96
			ResNet-50	RKD	8.29	31.08	50.03	0.97
			VGG-13		6.84	36.36	55.98	0.98
			ShuffleNet-V1		10.83	17.64	34.69	0.96

B. MODEL COMPRESSION USING KNOWLEDGE DISTILLATION METHODS ACROSS ARCHITECTURES

This experiment aims to compare the performance of the lightweight models (ResNet-20 and VGG-8) distilled using the vanilla KD approach with teacher models having the same and different (cross) architectures compared to the student models.

Table 3 shows EER and GMR at two different FMR points. These error metrics are shown for compact models (ResNet-20 and VGG-8) obtained using three KD approaches, namely the vanilla KD, RKD, and PKT. Overall, PKT obtained the least error rates among the other KD approaches on UFPR with an EER of 8.00%, and was decreased by 0.25% over the vanilla KD and RKD. The PKT and vanilla methods obtained an identical EER of 9.20%, which is lower than the RKD method by 1.32% averaged on VISOB 2.0. In terms of GMR, all the methods obtained similar performance on VISOB 2.0 for GMR@FMR=0.001 and GMR@FMR=0.0001 in the range [24.28-26.40] and [44.12-45.70], respectively. As can be seen, VGG-8 outperformed the other KD student, ResNet-20, on UFPR and VISOB 2.0 datasets. In the case of UFPR, VGG-8 obtained an overall EER of 7.82%, GMR@FMR=0.0001 of 21.42%, and GMR@FMR=0.001 of 34.03%. ResNet-20 obtained a slightly higher EER of 8.52% and GMR of 18.03% and 30.03% at FMR=0.001 and FMR=0.0001, respectively.

In terms of architectural variations, the highest GMR was obtained by the student VGG-8 model for the VGG-13 teacher model for both datasets. VGG-8 obtained 27.87% GMR@FMR=0.001% and 44.56% GMR@FMR=0.0001. At the same points, the performance dropped by an average of 1.14 when distilled with ResNet-50 as the teacher. Similarly, the ResNet-20 student model distilled with ResNet-50 as the teacher obtained a lower EER of 8.52% in comparison to distilled with VGG-13 as the teacher model (with an EER of 8.95%). Further, both students obtained their least performance with an average EER of 9.62%, GMR@FMR=0.001% of 17.49%, and GMR@FMR=0.0001 of 32.17% across all KD approaches when trained with ShuffleNet-V1 as the teacher model. This performance drop is due to architectural differences between the teacher and the student model.

Overall, the best performance improvement in terms of GMR and EER was obtained for VGG-8 and ResNet-20 student models when distilled using KD over the baseline. For instance, ResNet-20 post-distillation obtained an improvement of 2.48%, 7.94%, and 11.10% at EER, GMR@FMR=0.0001, and GMR@FMR=0.001, respectively, over the baseline. However, the improvement in the VGG-8 model was lower than in the ResNet-20 model. VGG-8 performance improved by 0.81%, 0.60%, and 1.89% in terms of EER, and GMR@FMR=0.0001, 0.001.

The PKT method obtained the lowest error rate on ResNet-20 and VGG-8 with ResNet-50 as the teacher model. Overall, the three KD approaches (vanilla KD, PKT, and RKD) obtained similar performance across all teacher-student combinations. For all the KD techniques, the same architecture between the teacher and the student model obtained better results over cross-architecture. In terms of performance-size trade-offs, ResNet-20, when distilled with ResNet-50 as a KD student, obtained an increase in EER of only 2.62% over ResNet-50 while being 80x smaller. Similarly, VGG-8

TABLE 4. Structured Pruning: Subject-independent analysis of the compact models generated using L2-norm based channel pruning on ResNet and VGG networks at different Compression Ratios (CR) on UFPR [25] and VISOB 2.0 [21] datasets.

Dataset	CNN	CR (%)	Model Size (MB)	EER (%)	GMR (%) @ FMR		AUC
					0.0001	0.001	
UFPR	ResNet-50	0 *(Baseline)*	97.39	6.15	27.92	39.50	0.99
	Pruned	10	62.54	9.28	23.86	36.30	0.98 (↓ 0.01)
		30	41.56	9.91	21.59	33.98	0.97 (↓ 0.02)
		50	28.23	10.44	16.86	27.49	0.94 (↓ 0.05)
	VGG-13	0 *(Baseline)*	38.46	6.99	26.19	38.97	0.98
	Pruned	10	31.29	9.68	26.10	39.25	0.97 (↓ 0.01)
		30	19.11	10.15	24.20	37.77	0.97 (↓ 0.01)
		50	9.841	11.88	21.50	36.00	0.96 (↓ 0.02)
	VGG-11	0 *(Baseline)*	37.71	7.85	26.51	38.47	0.98
	Pruned	10	30.68	10.07	25.97	38.66	0.97 (↓ 0.02)
		30	18.74	10.36	24.83	37.52	0.97 (↓ 0.02)
		50	9.650	12.05	20.91	33.39	0.96 (↓ 0.03)
VISOB 2.0	ResNet-50	0 *(Baseline)*	97.39	5.65	39.33	66.23	0.99
		10	62.54	8.22	38.82	55.96	0.98 (↓ 0.01)
		30	41.56	9.05	33.68	50.52	0.97 (↓ 0.02)
		50	28.23	9.41	33.90	47.84	0.97 (↓ 0.02)
	VGG-13	0 *(Baseline)*	38.46	6.83	39.09	58.78	0.99
		10	31.29	7.04	38.23	57.60	0.98 (↓ 0.01)
		30	19.11	7.07	35.62	57.53	0.98 (↓ 0.01)
		50	9.841	8.99	25.26	46.57	0.97 (↓ 0.02)
	VGG-11	0 *(Baseline)*	37.71	6.91	37.24	62.44	0.98
		10	30.68	7.13	37.78	53.13	0.98 (↓ 0.00)
		30	18.74	7.32	32.94	55.37	0.98 (↓ 0.00)
		50	9.650	8.11	33.01	54.02	0.98 (↓ 0.00)

TABLE 5. Unstructured Pruning: Subject-independent evaluation of the Layerwise Gradient Magnitude-based unstructured pruning on ResNet-50, VGG-13 and VGG-11 at three different Compression Ratios (CR) on UFPR [25] and VISOB 2.0 [21] datasets.

Dataset	CNN	CR (%)	Model Size (MB)	EER (%)	GMR (%) @ FMR		AUC
					0.0001	0.001	
UFPR	ResNet-50	0 *(Baseline)*	97.39	6.15	27.92	39.50	0.99
	Pruned	10	97.39	9.44	21.88	36.17	0.98 (↓ 0.01)
		30	97.39	9.41	22.09	36.30	0.98 (↓ 0.01)
		50	97.39	9.58	22.13	36.55	0.98 (↓ 0.01)
	VGG-13	0 *(Baseline)*	38.46	6.99	26.19	38.97	0.98
	Pruned	10	38.46	7.37	19.39	32.21	0.97 (↓ 0.02)
		30	38.46	7.31	19.52	32.29	0.97 (↓ 0.01)
		50	38.46	7.38	19.44	31.87	0.97 (↓ 0.01)
	VGG-11	0 *(Baseline)*	37.71	7.85	26.51	38.47	0.98
	Pruned	10	37.71	9.90	21.21	32.76	0.97 (↓ 0.02)
		30	37.71	9.77	21.12	32.97	0.97 (↓ 0.01)
		50	37.71	9.73	21.16	32.88	0.97 (↓ 0.01)
VISOB 2.0	ResNet-50	0 *(Baseline)*	97.39	5.65	39.33	66.23	0.99
		10	97.39	8.80	30.08	49.16	0.97 (↓ 0.02)
		30	97.39	8.80	29.44	49.03	0.97 (↓ 0.02)
		50	97.39	8.98	27.8	49.08	0.97 (↓ 0.02)
	VGG-13	0 *(Baseline)*	38.46	6.83	39.09	58.78	0.99
		10	38.46	7.43	27.29	49.79	0.98 (↓ 0.01)
		30	38.46	7.46	27.49	50.08	0.98 (↓ 0.01)
		50	38.46	7.37	27.15	50.05	0.98 (↓ 0.01)
	VGG-11	0 *(Baseline)*	37.71	6.91	37.24	62.44	0.98
		10	37.71	8.31	27.12	45.82	0.97 (↓ 0.01)
		30	37.71	8.26	27.11	45.92	0.97 (↓ 0.01)
		50	37.71	8.32	27.00	45.89	0.97 (↓ 0.01)

obtained an increase in average EER of only 1.34% when distilled by VGG-13 while being 2x smaller than VGG-13.

These results demonstrated the efficacy of the KD approaches in the performance enhancement of compact-size models using KD.

C. STRUCTURED VS. UNSTRUCTURED PRUNING

This experiment aims to compare the performance of the models (ResNet-50, VGG-13, and VGG-11) pruned using structured and unstructured pruning techniques based on L2-norm-based Channel Pruning (Table 4) and Layerwise Gradient Magnitude (Table 5) based pruning techniques, respectively.

As can be seen, the performance of the ResNet-50 model using unstructured pruning outperformed the structured pruning counterpart. It was evident as the mean EERs of 9.65% and 17.17% were obtained for the unstructured and structured pruning techniques across all CR. Overall, the AUC of ResNet-50 using the unstructured pruning dropped by about 0.08. The difference in EER of 1.18% and 0.16% was obtained for ResNet-50 using unstructured and structured pruning techniques, respectively, at a compression rate of 10% and 50%. The reason could be structured pruning on the residual connections prunes away filters inside the residual connections resulting in the number of output channels left unchanged [61].

Across both VGG models, the unstructured and structured pruning methods obtained an identical mean performance across GMR and AUC, and EER was lower for the unstructured pruning technique. VGG-8 and VGG-13 models obtained a difference of 1.65% in EER between the structured and unstructured pruning. Specifically, the difference in EER on VGG-11 between the 10% and 50% CR points ranged from [9.11-9.03]% and [8.60-10.08]% for the unstructured and structured pruning techniques, respectively. VGG-13 obtained a minimal performance drop between the two pruning methods. To investigate the impact of backbone architectures on the performance of structured pruning, we applied the L2-norm-based channel pruning on different variations of VGGs and ResNet (namely, ResNet-50, VGG-13, and VGG-11) as illustrated in Table 4. The average reduction in EER of the pruned models compared to the original versions across all CRs for ResNet-50, VGG-13, and VGG-11 was 1.7x, 0.50x, and 0.40x, respectively.

In the case of GMR, ResNet-50 performance degraded by an average of about 0.40x lower than the original model. However, VGG-11 and VGG-13 obtained a drop of less than 0.20x compared to the performance of the original models. Our observation regarding the poor performance of pruning techniques for ResNet architecture is in line with the literature [61], where it has been shown that pruning residual connections is very difficult. Pruning methods usually prune away filters inside the residual connections which leaves the number of output channels unchanged, leading the middle layers to be handicapped.

In terms of size-performance trade-off, ResNet-50 at CR of 10% obtained an EER of 9.12% and 8.75%, respectively, using unstructured and structured pruning. The size of structured-pruned ResNet-50 was 1.6x smaller at this CR than the baseline model, while the size of ResNet-50 with the unstructured pruning method did not change. At CRs of [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47], [48], [49], [50]%, ResNet-50 using structured pruning obtained an average EER of 9.7% while being 2.80x smaller in size over the baseline model. While ResNet-50 performance using the unstructured pruning technique was 9.2% with no change in the model size. The pruned versions of VGG-13 and VGG-11 at CR of 10% obtained an EER of about 8.25% and 8.48% using

TABLE 6. KD-Pruning: Subject-independent analysis of ResNet-20 and VGG-8 pruned using (L2-norm channel structured pruning) after being distilled using KD with ResNet-50 being the teacher on UFPR [25] and VISOB 2.0 [21] datasets.

Dataset	CNN	CR (%)	Model Size	EER(%)	GMR (%) @ FMR		AUC
					0.0001	0.001	
UFPR	ResNet-20	0 *(Baseline)*	1.233 MB	11.62	15.94	27.49	0.94
	Pruned	10	801.7 KB	25.96	5.82	10.54	0.81 (↓ 0.13)
		30	280.1 KB	33.38	2.40	5.61	0.73 (↓ 0.21)
		50	85.68 KB	47.74	0	0	0.52 (↓ 0.42)
	VGG-8	0 *(Baseline)*	16.45 MB	9.33	25.34	37.48	0.95
	Pruned	10	13.42 MB	16.94	13.95	23.69	0.91 (↓ 0.04)
		30	8.270 MB	16.94	11.72	21.12	0.91 (↓ 0.04)
		50	4.322 MB	20.89	8.68	16.78	0.88 (↓ 0.07)
VISOB 2.0	ResNet-20	0 *(Baseline)*	1.233 MB	11.55	8.87	21.68	0.93
	Pruned	10	801.7 KB	23.70	10.73	27.50	0.85 (↓ 0.08)
		30	280.1 MB	32.63	0	0	0.74 (↓ 0.19)
		50	85.68 KB	37.14	0	0	0.67 (↓ 0.26)
	VGG-8	0 *(Baseline)*	16.45 MB	8.66	22.12	39.57	0.96
	Pruned	10	13.42 MB	12.51	20.95	37.34	0.95 (↓ 0.01)
		30	8.270 MB	13.48	20.79	35.84	0.94 (↓ 0.02)
		50	4.322 MB	17.30	12.43	24.18	0.91 (↓ 0.05)

TABLE 7. Pruning-KD: Subject-independent analysis of pruned ResNet-20 and VGG-8 (using L2-norm-based channel pruning) at three different CR using Vanilla KD with ResNet-50 as the teacher on UFPR [25] and VISOB 2.0 [21] datasets under subject-independent analysis.

Dataset	CNN	CR (%)	Model Size	EER (%)	GMR (%) @ FMR		AUC
					0.0001	0.001	
UFPR	ResNet-20	0 *(Baseline)*	1.233 MB	11.62	15.94	27.49	0.94
	Pruned	10	801.7 KB	12.54	13.45	23.27	0.94 (0.0)
		30	280.1 KB	19.84	5.86	13.03	0.89 (↓ 0.05)
		50	85.68 KB	28.91	0.21	1.05	0.77 (↓ 0.17)
	VGG-8	0 *(Baseline)*	16.45 MB	9.33	25.34	37.48	0.95
	Pruned	10	13.42 MB	9.02	27.29	41.70	0.98 (↑ 0.03)
		30	8.270 MB	9.74	25.72	38.62	0.98 (↑ 0.03)
		50	4.322 MB	10.70	22.93	37.27	0.97 (↑ 0.02)
VISOB 2.0	ResNet-20	0 *(Baseline)*	1.233 MB	11.55	8.87	21.68	0.93
	Pruned	10	801.7 KB	11.21	16.17	33.31	0.96 (↑ 0.03)
		30	280.1 KB	15.50	9.02	19.67	0.92 (↓ 0.01)
		50	85.68 KB	25.48	0.30	2.01	0.82 (↓ 0.11)
	VGG-8	0 *(Baseline)*	16.45 MB	8.66	22.12	39.57	0.96
	Pruned	10	13.42 MB	6.98	39.57	59.84	0.98 (↑ 0.02)
		30	8.270 MB	6.78	36.96	56.33	0.98 (↑ 0.02)
		50	4.322 MB	7.04	38.00	58.64	0.98 (↑ 0.02)

unstructured and structured pruning techniques, respectively. The EER of the VGG models degraded by 0.05% and 1.01% at [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47], [48], [49], [50]% CRs compared to CR of 10% using structured and unstructured pruning methods, respectively.

Overall, structured and unstructured pruning techniques obtained similar performances. Pruning the architectures with residual connections may result in poor performance over other architectures. The models' size did not change using unstructured pruning because it results in having sparse weight matrices, thus leading to inefficiency in speed-up and compression on CPUs and GPUs if no dedicated hardware is available [49].

D. EVALUATION OF THE COMBINATION OF KD AND PRUNING

In Tables 6 and 7, we show the subject-independent verification performance of the models compressed using a combination of vanilla KD and structured pruning represented as *KD-Pruning* and *Pruning-KD*. Both the combinations are evaluated in terms of GMR and EER on UFPR and VISOB 2.0 datasets at three compression rates: [10, 30, 50%]. For all experiments in this section, the vanilla KD approach is used. This is because all the different KD techniques equally faired overall, as discussed in Section III-B.

As can be seen, ResNet-20 and VGG-8 models obtained better performance using Pruning-KD over KD-Pruning. For instance, ResNet-20 obtained an average EER of 39.42% using KD-Pruning and 26.27% using Pruning-KD. However, both combinations scored very similar GMR. For example, Pruning-KD achieved an increment of 2.6, 4.94, and 0.16 percentage difference in GMR@FMR=0.0001, GMR@FMR=0.001, and AUC, respectively, over KD-Pruning. Similarly, Pruning-KD outperformed KD-Pruning for VGG-8 as well. The performance of VGG-8 using Pruning-KD enhanced by 11.73, 17.24, and 0.1 for GMR@FMR=0.0001, GMR@FMR=0.001, and AUC, respectively, over KD-Pruning combination.

Furthermore, we measured the percentage change between the performance of the original models for both combinations. The EER of the original ResNet-20 model increased by 3.0x over CRs at [10-30%] using KD-Pruning, and less than 1x using Pruning-KD. Likewise, the EER over 50% CR increased by 3.77x and 2.4x for KD-Pruning and Pruning-KD, respectively, over the original model. On the other hand, VGG-8 obtained less degradation compared to ResNet-20. It was evident as VGG-8 degraded approximately by 1.7x over CRs at [10-30%] and 2.0 at 50% CR using KD-Pruning over the original model. The gap in degradation decreased to 0.09x using Pruning-KD for the [10-30%] CRs and less than 0.50x for 50% CR.

In terms of compression rates, the lowest performance for both models was obtained for 50% CR as ResNet-20 obtained an average EER of 35.70% and AUC of 0.68 for both combinations. However, VGG-8 obtained a better overall EER and AUC of 13.99% and 0.94, respectively.

Regarding the accuracy-size trade-off for Pruning-KD over KD, the ResNet-20 model obtained an average EER of 9.12% when distilled using KD. However, the performance dropped to 11.88% at CR of 10%, with the reduction in the model size of around 1.5x smaller using Pruning-KD. Similarly, VGG-8 obtained an EER of 8.19% using KD, but it reduced to 8.0% using Pruning-KD, and the model size reduced up to 1.22x smaller.

Overall, Pruning-KD performed better across all models compared to KD-Pruning. The reason is pruning technique prunes away the established trained connections. Therefore, performing KD on pruned models as students act as a fine-tuning mechanism on unpruned connections resulting in improved generalization. However, Pruning-KD and KD-Pruning obtained similar performance compared to standalone KD and pruning techniques with an added advantage of 1.22x further model size reduction.

E. EVALUATION OF THE COMBINATION OF KD AND QUANTIZATION

In order to understand the efficacy of quantization on the various models compressed using different KD tech-

TABLE 8. KD-Quantization: Subject-independent analysis of applying PTQ-based quantization on the compact models after distilled using Vanilla KD, RKD, and PKT techniques on ResNet-20 and VGG-8 models using UFPR [25] and VISOB 2.0 [21] datasets.

Dataset	Quantized Student	Model Size (MB)	Teacher	KD	EER (%)	GMR (%) @ FMR		AUC
						0.0001	**0.001**	
UFPR	ResNet-20	1.162	ResNet-50	Vanilla	8.19	19.98	31.75	0.98
			VGG-13		8.52	14.66	27.01	0.98
			ShuffleNet-V1		9.20	19.01	30.52	0.98
			ResNet-50	PKT	7.66	18.97	31.45	0.98
			VGG-13		8.42	16.91	27.70	0.98
			ShuffleNet-V1		8.84	18.38	30.06	0.97
			ResNet-50	RKD	7.86	19.81	33.81	0.98
			VGG-13		9.08	13.87	25.84	0.98
			ShuffleNet-V1		9.21	19.44	31.58	0.97
	VGG-8	15.87	ResNet-50	Vanilla	7.71	21.33	34.78	0.98
			VGG-13		7.62	20.49	33.31	0.98
			ShuffleNet-V1		8.58	20.69	31.55	0.98
			ResNet-50	PKT	7.48	24.03	38.11	0.98
			VGG-13		7.83	20.78	34.15	0.98
			ShuffleNet-V1		8.18	20.70	31.49	0.97
			ResNet-50	RKD	7.48	21.96	35.75	0.98
			VGG-13		7.50	21.67	34.53	0.98
			ShuffleNet-V1		8.57	21.02	32.30	0.97
VISOB 2.0	ResNet-20	1.162	ResNet-50	Vanilla	8.96	26.92	45.97	0.97
			VGG-13		8.89	21.74	40.71	0.97
			ShuffleNet-V1		11.40	13.82	31.21	0.95
			ResNet-50	PKT	8.71	26.48	48.27	0.97
			VGG-13		8.49	25.41	45.39	0.97
			ShuffleNet-V1		11.17	16.32	34.85	0.94
			ResNet-50	RKD	9.07	25.47	48.31	0.97
			VGG-13		9.14	19.43	38.14	0.97
			ShuffleNet-V1		10.46	12.82	29.91	0.95
	VGG-8	15.87	ResNet-50	Vanilla	7.88	31.55	50.63	0.98
			VGG-13		7.28	34.54	54.81	0.98
			ShuffleNet-V1		10.02	16.93	25.31	0.96
			ResNet-50	PKT	8.02	29.91	51.58	0.98
			VGG-13		7.75	33.62	54.58	0.98
			ShuffleNet-V1		10.98	14.06	34.05	0.96
			ResNet-50	RKD	8.32	31.18	49.94	0.97
			VGG-13		6.83	36.50	56.02	0.98
			ShuffleNet-V1		11.26	18.07	35.01	0.95

niques, we evaluated the impact of the PTQ technique on ResNet-20 and VGG-8 models. Table 8 shows the subject-independent verification accuracy of the quantized models after being distilled using different KD techniques (vanilla KD, RKD, and PKT) and teachers (ResNet-20, VGG-13, and ShuffleNet-V1).

Overall, the KD-Quantization combination performed similarly to the models distilled using KD with very minimal degradation in the performance. After quantization, the mean EER across all student-teacher models was 8.63%, which was higher by 0.05 percentage points over distilled models without quantization. The GMR of the distilled and quantized models decreased by 0.68 and 0.87 at GMR@FMR=0.0001 and GMR@FMR=0.001, respectively, over the distilled models. The distilled models were further reduced in size by an average of 0.33 MB after being quantized. These results suggested that the combination of KD and quantization techniques further reduced the model size without significantly impacting the performance.

TABLE 9. Structured Pruning-Quantization: Subject-independent analysis of PTQ-based quantization applied on the pruned models using L2-norm-based channel pruning method on ResNet-50, VGG-13 and VGG-11 models using UFPR [25] and VISOB 2.0 [21] datasets.

Dataset	Quantized Model	CR (%)	Model Size (MB)	EER (%)	GMR (%) @ FMR		AUC
					0.0001	**0.001**	
UFPR	ResNet-50	0 *(Baseline)*	95.08	7.66	26.65	38.41	0.98
	Pruned	10	61.03	9.28	23.82	36.21	0.97 (↓ 0.02)
		30	40.61	9.85	21.42	34.02	0.97 (↓ 0.02)
		50	27.68	10.40	16.78	27.40	0.94 (↓ 0.05)
	VGG-13	0 *(Baseline)*	37.88	7.49	26.15	38.01	0.97
	Pruned	10	30.77	9.65	26.01	38.74	0.97 (↓ 0.01)
		30	18.70	10.15	24.11	37.90	0.97 (↓ 0.01)
		50	9.55	11.85	21.71	36.17	0.96 (↓ 0.02)
	VGG-11	0 *(Baseline)*	37.18	8.11	26.15	38.01	0.96
	Pruned	10	30.16	10.41	24.75	39.25	0.97 (↓ 0.01)
		30	18.34	10.07	25.84	37.56	0.97 (↓ 0.01)
		50	9.364	12.13	20.87	33.35	0.96 (↓ 0.02)
VISOB 2.0	ResNet-50	0 *(Baseline)*	95.08	5.90	38.93	65.26	0.98
		10	61.03	8.21	39.00	56.37	0.98 (↓ 0.01)
		30	40.61	8.96	33.11	50.17	0.97(↓ 0.02)
		50	27.68	9.66	30.4	47.24	0.97 (↓ 0.02)
	VGG-13	0 *(Baseline)*	37.88	7.54	38.05	58.37	0.98
		10	30.77	7.52	35.53	56.71	0.98 (↓ 0.01)
		30	18.70	7.94	37.24	57.48	0.98 (↓ 0.01)
		50	9.55	8.12	29.6	48.13	0.97 (↓ 0.02)
	VGG-11	0 *(Baseline)*	37.18	7.17	36.88	62.37	0.98
		10	30.16	7.33	35.04	55.72	0.98 (0.00)
		30	18.34	8.09	33.68	53.77	0.97 (↓ 0.01)
		50	9.364	7.83	35.21	53.36	0.98 (0.00)

F. EVALUATION OF THE COMBINATION OF PRUNING AND QUANTIZATION

Tables 9 and 10 show the performance of subject-independent analysis of the PTQ technique applied on the pruned version of ResNet-50, VGG-13, and VGG-11 models. L2-norm-based channel structured pruning and Layer-wise Gradient Magnitude-based unstructured pruning were applied for model compression. Overall, the combination of unstructured pruning and quantization obtained a lower EER score compared to structured pruning with quantization. Mostly, the average EER, GMR, and AUC of standalone pruned and those quantized after pruning, i.e., the Pruning-Quantization combination was equivalent. For instance, ResNet-50 obtained no change across EERs (an average of 9.39%) between the standalone pruned using structured pruning and those quantized after being pruned across the compression rates. The average size of the combination of pruned and quantized models was 1 MB smaller than the model pruned using the unstructured pruning technique. Recall that unstructured pruning does not have any impact on the model size. The combination of quantization and structured pruning on VGG-11 and VGG-13 models obtained an increased EER of 0.11% over the standalone pruned model using structured pruning, while the size of the

TABLE 10. Unstructured Pruning-Quantization: Subject-independent analysis of the PTQ-based quantized models after using layer-wise magnitude-based unstructured pruning on ResNet-50, VGG-13, and VGG-11 models using UFPR [25] VISOB 2.0 [21] and datasets.

Dataset	Quantized Model	CR (%)	Param. Size (m)	EER (%)	GMR (%) @ FMR		AUC
					0.0001	**0.001**	
UFPR	ResNet-50	0 *(Baseline)*	95.08	7.66	26.65	38.41	0.98
	Pruned	10	95.08	9.44	21.75	36.21	0.98 (↓ 0.01)
		30	95.08	9.39	22.26	36.26	0.98 (↓ 0.01)
		50	95.08	9.56	22.13	36.59	0.98 (↓ 0.01)
	VGG-13	0 *(Baseline)*	37.88	7.49	26.15	38.01	0.97
	Pruned	10	37.88	7.36	19.35	32.25	0.97 (↓ 0.02)
		30	37.88	7.28	19.56	32.29	0.97 (↓ 0.02)
		50	37.88	7.41	19.52	31.87	0.97 (↓ 0.02)
	VGG-11	0 *(Baseline)*	37.18	8.11	26.15	38.01	0.96
	Pruned	10	37.18	9.86	21.25	32.80	0.97 (↓ 0.01)
		30	37.18	9.79	21.16	32.93	0.97 (↓ 0.01)
		50	37.18	9.77	21.16	32.88	0.97 (↓ 0.01)
VISOB 2.0	ResNet-50	0 *(Baseline)*	95.08	5.90	38.93	65.26	0.98
		10	95.08	8.80	30.00	49.17	0.97 (↓ 0.02)
		30	95.08	8.80	29.39	49.05	0.97 (↓ 0.02)
		50	95.08	9.01	27.62	49.00	0.97 (↓ 0.02)
	VGG-13	0 *(Baseline)*	37.88	7.54	38.05	58.37	0.98
		10	37.88	7.42	27.26	49.79	0.98 (↓ 0.01)
		30.00	37.88	7.45	27.41	50.03	0.98 (↓ 0.01)
		50	37.88	7.36	27.24	50.09	0.98 (↓ 0.01)
	VGG-11	0 *(Baseline)*	37.18	7.17	36.88	62.37	0.98
		10	37.18	8.29	27.2	45.80	0.97 (↓ 0.02)
		30	37.18	8.26	27.15	46.02	0.97 (↓ 0.02)
		50	37.18	8.33	27.08	45.94	0.97 (↓ 0.02)

TABLE 11. Pruning-KD-Quantization // KD-Pruning-Quantization: Subject-independent analysis of PTQ-based quantization on the *Pruning-KD* and *KD-Pruning* on ResNet-20 and VGG-8 models using UFPR [25] and VISOB 2.0 [21] datasets. For all these experiments Vanilla KD and L2-norm-based structured pruning has been used.

Dataset	Quantized Model	Combination	CR(%)	Model Size (MB)	EER (%)	GMR (%) @ FMR		AUC
						0.0001	**0.001**	
UFPR	ResNet-20	-	0 *(Baseline)*	1.162 MB	12.54	15.32	27.91	0.93
		Pruning-KD-Quantization	10	753.1 KB	14.54	13.32	23.27	0.94 (0.0)
			30	261.8 KB	19.93	5.78	13.15	0.89 (↓ 0.05)
			50	81.98 KB	28.79	0.21	1.05	0.77 (↓ 0.17)
		KD-Pruning-Quantization	10	753.1 KB	26.13	5.73	10.67	0.81 (↓ 0.13)
			30	261.8 KB	33.29	0.63	2.78	0.72 (↓ 0.22)
			50	81.98 KB	47.20	0	0	0.53 (↓ 0.41)
	VGG-8	-	0 *(Baseline)*	15.87 MB	10.12	24.57	35.79	0.92
		Pruning-KD-Quantization	10	12.91 MB	9.02	27.11	41.82	0.98 (↑ 0.03)
			30	7.868 MB	9.77	25.67	38.49	0.98 (↑ 0.03)
			50	4.035 MB	10.69	22.81	37.02	0.97 (↑ 0.02)
		KD-Pruning-Quantization	10	12.91 MB	16.94	13.87	23.65	0.91 (↓ 0.04)
			30	7.868 MB	16.94	11.85	21.21	0.91 (↓ 0.04)
			50	4.035 MB	20.79	8.52	16.69	0.88 (↓ 0.07)
VISOB 2.0	ResNet-20	-	0 *(Baseline)*	1.162 MB	11.65	8.48	21.38	0.93
		Pruning-KD-Quantization	10	753.1 KB	10.88	16.67	32.95	0.96 (↑ 0.03)
			30	261.8 KB	15.77	8.26	20.43	0.92 (↓ 0.01)
			50	81.98 KB	26.76	0.21	1.30	0.81 (↓ 0.12)
		KD-Pruning-Quantization	10	753.1 KB	23.12	4.03	10.59	0.85 (↓ 0.08)
			30	261.8 KB	33.47	0.51	2.42	0.73 (↓ 0.20)
			50	81.98 KB	40.07	0.00	0.00	0.62 (↓ 0.31)
	VGG-8	-	0 *(Baseline)*	15.87 MB	8.36	21.66	38.70	0.96
		Pruning-KD-Quantization	10	12.91 MB	6.47	41.63	62.15	0.98 (↑ 0.02)
			30	7.868 MB	6.71	41.37	60.59	0.98 (↑ 0.02)
			50	4.035 MB	7.06	39.35	58.98	0.98 (↑ 0.02)
		KD-Pruning-Quantization	10	12.91 MB	13.57	19.67	36.89	0.94 (↓ 0.02)
			30	7.868 MB	13.48	16.27	32.09	0.94 (↓ 0.02)
			50	4.035 MB	17.42	12.55	23.84	0.91 (↓ 0.05)

combination of pruned and quantized models decreased by about 0.44 MB.

Further, Table 11 shows the subject-independent analysis of the quantized models after being compressed using a combination of KD and pruning techniques (i.e., Pruning-KD-Quantization and KD-Pruning-Quantization). The models were quantized using PTQ after being distilled and pruned using vanilla KD and L2-norm-based structured pruning. The results suggested that the PTQ method could preserve the equivalent performance across all points (EER, GMR, and AUC). The quantized models resulted in a further size reduction of 23.66 KB for ResNet-20 and 0.43 MB for VGG-8 over both KD-Pruning and Pruning-KD. Note that KD does not impact the original size of the model since it is a training and learning mechanism for performance enhancement of the compact student models.

Overall, the combination of quantized and pruned models (using unstructured pruning) obtained almost close to zero performance difference (about 0.04% difference) compared to the pruned models. Likewise, applying quantization to models compressed using structured pruning obtained an EER decrease of approximately 0.14%. This analysis suggested that the combination of pruning and quantization techniques further reduced the models' sizes while maintaining a performance similar to those of pruned models.

G. EVALUATION OF RUN-TIME PERFORMANCE OF THE COMPRESSED MODELS ON SMARTPHONE

This section aims to evaluate and compare the run-time performance of the compressed and original models when deployed on an Apple iPhone-14 smartphone. For implementation details on the deployment of the models on smartphones, readers are referred to section V-B2. The run-time performance is measured in terms of extraction time in seconds, theoretical speed-up, and the memory usage in (MB) of the baseline models and their compressed versions on an Apple iPhone-14 smartphone with iOS version 16 (Table 12). The extraction time is estimated based on the time taken for feature extraction from the last fully connected layer of the models. The inference efficiency is measured by theoretical speed-up. Theoretical speed-up was computed using floating point operations per second (FLOPs) before and after pruning [62].

The baseline ResNet-50 model with the EER of 5.90% obtained the slowest feature extraction time of 2.98 seconds on iPhone-14 with a memory use of 248 MB. The pruned version of ResNet-50 using structured pruning while obtaining an increased EER of 8.75%, obtained 2.0x faster extraction time with a theoretical speed-up of about 4.37 across the CR points.

Further, the quantized version of the baseline ResNet-50 obtained a further extraction speed of 1.67x and 1.11x reduction in memory usage compared to the baseline ResNet-50. Moreover, applying quantization to the pruned version of ResNet-50 (using structured pruning) further resulted in a speed-up compared to the pruned models. Specifically, the combination of the quantized and pruned model obtained an increase of 1.1x in the feature extraction time and an average 18 MB drop in memory usage across the CR points. Note that theoretical speed-up calculation is not applicable for quantized models because the quantization process does not impact the no. of floating-point operations performed (see Table 12).

Similarly, the baseline VGG-11 with an EER of 7.06% obtained a feature extraction time of 1.60 seconds. The structured pruned version of VGG-11 obtained a mean theoretical speed-up of 2.55, and the memory usage dropped almost 9.0% across the CR points. Furthermore, the quantized version of the baseline VGG-11 obtained a further increase in the extraction time of 1.11x and a reduction in memory usage of 1.17x compared to the baseline VGG-11 model.

TABLE 12. Run-time Performance: Theoretical speed-up, Extraction Time (ET) in seconds (s), and memory usage in MB of the Baseline and compressed models on Apple iPhone 14 smartphone.

Model	Compression Method	CR (%)	Theoretical Speed-up	ET (s)	Memory Usage (MB)
ResNet-50	*Baseline*	N/A	1.0	2.98	248
	Quantization	N/A	N/A	1.80	235
	Unstructured Pruning	10	1.0	1.80	248
		30	1.0	1.74	248
		50	1.0	1.70	248
	Structured Pruning	10	1.55	1.82	217
		30	3.42	1.36	186
		50	8.13	1.33	170
	Structured Pruning-Quantization	10	N/A	1.68	202
		30	N/A	1.29	170
		50	N/A	1.24	155
VGG-13	*Baseline*	N/A	1.0	1.86	183
	Quantization	N/A	N/A	1.71	168
	Unstructured Pruning	10	1.0	1.75	183
		30	1.0	1.68	183
		50	1.0	1.61	183
	Structured Pruning	10	1.26	1.63	177
		30	2.07	1.58	169
		50	4.14	1.47	160
	Structured Pruning-Quantization	10	N/A	1.42	152
		30	N/A	1.35	148
		50	N/A	1.29	145
VGG-11	*Baseline*	N/A	1.0	1.60	178
	Quantization	N/A	N/A	1.44	152
	Unstructured Pruning	10	1.0	1.47	178
		30	1.0	1.29	178
		50	1.0	1.26	178
	Structured Pruning	10	1.24	1.37	172
		30	2.10	1.32	163
		50	4.20	1.22	153
	Structured Pruning-Quantization	10	N/A	1.24	146
		30	N/A	1.19	142
		50	N/A	1.16	140
ResNet-20	*Baseline*	N/A	1.0	1.50	160
	Quantization	N/A	N/A	1.25	145
	Structured Pruning	10	1.25	1.37	154
		30	4.0	1.30	150
		50	N/A	1.28	146
	Structured Pruning-Quantization	10	N/A	0.60	59
		30	N/A	0.36	59
		50	N/A	0.26	58
VGG-8	*Baseline*	N/A	1.0	1.81	171
	Quantization	N/A	N/A	1.39	153
	Structured Pruning	10	1.25	1.48	168
		30	2.0	1.30	149
		50	5.0	1.23	145
	Structured Pruning-Quantization	10	N/A	1.05	45
		30	N/A	0.88	41
		50	N/A	0.54	36

Note that the pruned versions of ResNet-50 and VGG-11 using unstructured pruning obtained identical memory usage with a faster extraction latency than their original versions. Further, the KD approach majorly enhances the classification performance of the compact models and does not impact their run-time performance. Therefore, we did not include the KD approaches in this analysis.

Overall, the feature extraction time was up to 1.50x faster with a 1.20x reduction in memory usage and an increase in the theoretical speed of 2.22 could be obtained using structured pruning techniques. The combination of quantization and structured pruning could further speed up the extraction time by 1.40x and memory usage by 1.15x when deployed on a smartphone.

VII. KEY FINDINGS

The key findings from all the experiments are summarized as follows:

- The compact models trained from scratch (ResNet-20 and VGG-8) obtained lower performance (an EER increase of 3.9%) over large-sized models (ResNet-50, VGG-13, and VGG-11) in user verification using ocular biometrics. The possible reason is due to the parameter size difference.
- Applying quantization on the baseline models could reduce the model's size by 1.02x with minimal to no degradation in the performance.
- KD techniques improved the performance of the compact models (ResNet-20 and VGG-8) by about 2.12% reduction in their EER values. Different KD approaches performed similarly, with PKT outperforming others for specific models. These KD approaches obtained better performance when the architecture of the teacher and student model was the same. This answers the research question (**RQ1**) posed in Section I.
- Among the pruning techniques, structured pruning may degrade the performance of the models based on residual connections (e.g., ResNets). On the other hand, unstructured pruning only enhances the feature extraction time of the compressed models without any change in the model size. That is due to the generation of sparse matrices. The overall size reduction of the large-size models (ResNet-50, VGG-13, and VGG-11) with structured pruning across all the CR points was 2.0x smaller, 1.37x higher EER, and 1.47x faster ET than the baseline models. This answers the research question (**RQ2**) posed in Section I.
- The combination of compression techniques could further reduce the model size without significantly impacting the performance. For instance, the application of quantization (Pruning-KD-Quantization) on the pruned and distilled model could further reduce the model size by 1.15x smaller than the standalone pruned model without impacting the performance significantly. Overall, the combination of compression techniques has the benefit of further reducing the model size by 2.15x without impacting the performance significantly over the standalone techniques. This answers the research question (**RQ3**) posed in Section I.
- The feature extraction time due to compression could be up to 1.50x faster, with a 1.20x reduction in memory usage and an increase in theoretical speed of 2.22 due to pruning. Quantization could further reduce the extraction time by 1.30x and memory usage by 1.10x when deployed on the smartphone.

VIII. CONCLUSION

In this paper, we have performed an extensive study of the neural network compression techniques as standalone and in combination for ocular-based user authentication on smartphones. Experimental investigations are performed on two latest large-scale ocular biometric datasets captured using different smartphones. The results provide detailed insight into the pros and cons of popular neural network compression techniques as standalone and combined for ocular biometrics. The run-time performance of these compression techniques is also evaluated on an Apple iPhone 14 device. This work benchmarks the results of standard compression techniques (standalone and in combination) on the latest datasets to aid further research and development on lightweight models for ocular-based user authentication on smartphones. As a part of future work, an ablation study will be performed with different weight initialization techniques, optimizers, and learning rate schedules [63] on the same test bed.

ACKNOWLEDGMENT

Ajita Rattani is co-organizer of the IEEE ICIP 2016 VISOB 1.0 and IEEE WCCI 2020 VISOB 2.0 international mobile ocular biometric competitions.

REFERENCES

[1] P. Kumari and K. R. Seeja, "Periocular biometrics: A survey," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 34, no. 4, pp. 1086–1097, 2019.
[2] N. Reddy, A. Rattani, and R. Derakhshani, "OcularNet: Deep patch-based ocular biometric recognition," in *Proc. IEEE Int. Symp. Technol. Homeland Secur. (HST)*. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers, Dec. 2018, pp. 1–6.
[3] M. De Marsico, M. Nappi, D. Riccio, and H. Wechsler, "Mobile iris challenge evaluation (MICHE)-I, biometric iris dataset and protocols," *Pattern Recognit. Lett.*, vol. 57, pp. 17–23, May 2015.
[4] A. Almadan and A. Rattani, "Towards on-device face recognition in body-worn cameras," in *Proc. IEEE Int. Workshop Biometrics Forensics (IWBF)*, May 2021, pp. 1–6.
[5] A. Almadan, A. Krishnan, and A. Rattani, "BWCFace: Open-set face recognition using body-worn camera," in *Proc. 19th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2020, pp. 1036–1043.
[6] N. Reddy, A. Rattani, and R. Derakhshani, "Generalizable deep features for ocular biometrics," *Image Vis. Comput.*, vol. 103, Nov. 2020, Art. no. 103996.
[7] A. Rattani and R. Derakhshani, "Ocular biometrics in the visible spectrum: A survey," *Image Vis. Comput.*, vol. 59, pp. 1–16, Mar. 2017.
[8] K. Raja, R. Ramachandra, and C. Busch, "Collaborative representation of blur invariant deep sparse features for periocular recognition from smartphones," *Image Vis. Comput.*, vol. 101, Sep. 2020, Art. no. 103979.
[9] F. Alonso-Fernandez, K. H. Diaz, S. Ramis, F. J. Perales, and J. Bigun, "Soft-biometrics estimation in the era of facial masks," in *Proc. Int. Conf. BIOSIG*, Darmstadt, Germany, 2020, pp. 1–6.
[10] F. Alonso-Fernandez, K. H. Diaz, S. Ramis, F. J. Perales, and J. Bigun, "Facial masks and soft-biometrics: Leveraging face recognition CNNs for age and gender prediction on mobile ocular images," *IET Biometrics*, vol. 10, no. 5, pp. 562–580, 2021.
[11] G. Lovisotto, R. Malik, I. Sluganovic, M. Roeschlin, P. Trueman, and I. Martinovic, "Mobile biometrics in financial services: A five factor framework," Dept. Comput. Sci., Univ. Oxford, Oxford, U.K., Tech. Rep. CS-RR-17-03, 2017.
[12] A. Avdić, "Use of biometrics in mobile banking security: Case study of Croatian banks," *Int. J. Comput. Sci. Netw. Secur.*, vol. 19, no. 10, pp. 83–89, 2019.
[13] K. A. Shakil, F. J. Zareen, M. Alam, and S. Jabin, "BAMCloud: A cloud based mobile biometric authentication framework," *Multimedia Tools Appl.*, pp. 1–30, Jul. 2022.
[14] S. H. Moi, P. Y. Yong, R. Hassan, H. Asmuni, R. Mohamad, F. C. Weng, and S. Kasim, "An improved approach of iris biometric authentication performance and security with cryptography and error correction codes," *JOIV, Int. J. Informat. Vis.*, vol. 6, no. 2, pp. 531–539, 2022.
[15] P. Drozdowski, F. Struck, C. Rathgeb, and C. Busch, "Detection of glasses in near-infrared ocular images," in *Proc. Int. Conf. Biometrics (ICB)*, Feb. 2018, pp. 202–208.
[16] A. Krishnan, A. Almadan, and A. Rattani, "Probing fairness of mobile ocular biometrics methods across gender on VISOB 2.0 dataset," in *Proc. Int. Conf. Pattern Recognit.* Milan, Italy: Springer, 2021, pp. 229–243.
[17] N. Reddy, A. Rattani, and R. Derakhshani, "A robust scheme for iris segmentation in mobile environment," in *Proc. IEEE Symp. Technol. Homeland Secur. (HST)*, May 2016, pp. 1–6.
[18] D. Hamilton. (2015). *Eyeverify Technology Uses Smartphone Cameras to Easily Verify ID by Eye Veins*. [Online]. Available: https://www.biometricupdate.com
[19] S. Maity, "Face ID vs iris scanner: Is iPhone X more secure than Galaxy S8," Tech. Rep., Sep. 2017.
[20] H. Proença and J. C. Neves, "Deep-PRWIS: Periocular recognition without the iris and sclera using deep learning frameworks," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 4, pp. 888–896, Apr. 2018.
[21] H. Nguyen, N. Reddy, A. Rattani, and R. Derakhshani, "VISOB 2.0—Second international competition on mobile ocular biometric recognition," in *Proc. IAPR ICPR*, Rome, Italy, 2020, pp. 1–8.
[22] A. Rattani and R. Derakhshani, "On fine-tuning convolutional neural networks for smartphone based ocular recognition," in *Proc. IEEE Int. Joint Conf. Biometrics (IJCB)*, Oct. 2017, pp. 762–767.
[23] D. Kerrigan, M. Trokielewicz, A. Czajka, and K. W. Bowyer, "Iris recognition with image segmentation employing retrained off-the-shelf deep neural networks," in *Proc. Int. Conf. Biometrics (ICB)*, 2019, pp. 1–7.
[24] Y.-H. Li, W. R. Putri, M. S. Aslam, and C.-C. Chang, "Robust iris segmentation algorithm in non-cooperative environments using interleaved residual U-Net," *Sensors*, vol. 21, no. 4, p. 1434, 2021.
[25] L. A. Zanlorensi, R. Laroca, D. R. Lucio, L. R. Santos, A. S. Britto Jr., and D. Menotti, "UFPR-periocular: A periocular dataset collected by mobile devices in unconstrained scenarios," 2020, *arXiv:2011.12427*.
[26] L. A. Zanlorensi, R. Laroca, D. R. Lucio, L. R. Santos, A. S. Britto, and D. Menotti, "A new periocular dataset collected by mobile devices in unconstrained scenarios," *Sci. Rep.*, vol. 12, no. 1, pp. 1–18, 2022.
[27] A. Rattani, R. Derakhshani, S. K. Saripalle, and V. Gottemukkula, "ICIP 2016 competition on mobile ocular biometric recognition," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 320–324.
[28] A. Almadan and A. Rattani, "Compact CNN models for on-device ocular-based user recognition in mobile devices," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2021, pp. 1–7.
[29] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proc. IEEE*, vol. 107, no. 8, pp. 1655–1674, Aug. 2019.
[30] D. Blalock, J. J. G. Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" *Proc. Mach. Learn. Syst.*, vol. 2, pp. 129–146, Mar. 2020.
[31] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 869–904, 2nd Quart., 2020.
[32] M. G. S. Murshed, C. Murphy, D. Hou, N. Khan, G. Ananthanarayanan, and F. Hussain, "Machine learning at the network edge: A survey," *ACM Comput. Surv.*, vol. 54, no. 8, pp. 1–37, Oct. 2021.
[33] S. B. Atitallah, M. Driss, W. Boulila, and H. B. Ghézala, "Leveraging deep learning and IoT big data analytics to support the smart cities development: Review and future directions," *Comput. Sci. Rev.*, vol. 38, Nov. 2020, Art. no. 100303.
[34] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.
[35] S. A. Janowsky, "Pruning versus clipping in neural networks," *Phys. Rev. A, Gen. Phys.*, vol. 39, no. 12, p. 6600, 1989.
[36] E. D. Karnin, "A simple procedure for pruning back-propagation trained neural networks," *IEEE Trans. Neural Netw.*, vol. 1, no. 2, pp. 239–242, Jun. 1990.
[37] Z. Lin, M. Courbariaux, R. Memisevic, and Y. Bengio, "Neural networks with few multiplications," 2015, *arXiv:1510.03009*.
[38] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2704–2713.
[39] W. Park, D. Kim, Y. Lu, and M. Cho, "Relational knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 3967–3976.
[40] N. Passalis and A. Tefas, "Probabilistic knowledge transfer for deep representation learning," *CoRR*, vol. abs/1803.10837, no. 2, p. 5, 2018.
[41] J. Ye, X. Lu, Z. Lin, and J. Z. Wang, "Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers," 2018, *arXiv:1802.00124*.

[42] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A survey of quantization methods for efficient neural network inference," 2021, *arXiv:2103.13630*.

[43] O. Zafrir, G. Boudoukh, P. Izsak, and M. Wasserblat, "Q8BERT: Quantized 8bit BERT," in *Proc. 5th Workshop Energy Efficient Mach. Learn. Cogn. Comput. NeurIPS, Ed., (EMC2-NIPS)*, 2019, pp. 36–39.

[44] F. Boutros, N. Damer, M. Fang, K. Raja, F. Kirchbuchner, and A. Kuijper, "Compact models for periocular verification through knowledge distillation," in *Proc. Int. Conf. Biometrics Special Interest Group (BIOSIG)*, 2020, pp. 1–5.

[45] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[47] F. Boutros, N. Damer, K. Raja, F. Kirchbuchner, and A. Kuijper, "Template-driven knowledge distillation for compact and accurate periocular biometrics deep-learning models," *Sensors*, vol. 22, no. 5, p. 1921, 2022.

[48] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 1389–1397.

[49] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "EIE: Efficient inference engine on compressed deep neural network," *ACM SIGARCH Comput. Archit. News*, vol. 44, no. 3, pp. 243–254, 2016.

[50] Z. Zhuang, M. Tan, B. Zhuang, J. Liu, Y. Guo, Q. Wu, J. Huang, and J. Zhu, "Discrimination-aware channel pruning for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–12.

[51] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *Int. J. Comput. Vis.*, vol. 129, pp. 1789–1819, Mar. 2021.

[52] H. D. Nguyen, A. Alexandridis, and A. Mouchtaris, "Quantization aware training with absolute-cosine regularization for automatic speech recognition," in *Proc. Interspeech*, 2020, pp. 3366–3370.

[53] B. Ron, Y. Nahshan, and D. Soudry, "Post training 4-bit quantization of convolutional networks for rapid-deployment," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, 2019, pp. 7948–7956.

[54] X. Wei, R. Gong, Y. Li, X. Liu, and F. Yu, "QDrop: Randomly dropping quantization for extremely low-bit post-training quantization," 2022, *arXiv:2203.05740*.

[55] I. Hubara, Y. Nahshan, Y. Hanani, R. Banner, and D. Soudry, "Improving post training neural quantization: Layer-wise calibration and integer programming," 2020, *arXiv:2006.10518*.

[56] H. Wu, P. Judd, X. Zhang, M. Isaev, and P. Micikevicius, "Integer quantization for deep learning inference: Principles and empirical evaluation," 2020, *arXiv:2004.09602*.

[57] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "SphereFace: Deep hypersphere embedding for face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 212–220.

[58] Z. DeVito, J. Ansel, W. Constable, M. Suo, A. Zhang, and K. Hazelwood, "Using Python for model inference in deep learning," 2021, *arXiv:2104.00254*.

[59] *Apple Developer Documentation*. Accessed: Feb. 1, 2023. [Online]. Available: https://developer.apple.com/documentation/xcode-release-notes/xcode-14-release-notes

[60] *Apple Developer Documentation*. Accessed: Feb. 1, 2023. [Online]. Available: https://developer.apple.com/documentation/metrickit/improving_your_app_s_performance/reducing_your_app_s_memory_use

[61] J.-H. Luo and J. Wu, "Neural network pruning with residual-connections and limited-data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 1458–1467.

[62] Q. Zhang, M. Zhang, M. Wang, W. Sui, C. Meng, J. Yang, W. Kong, X. Cui, and W. Lin, "Efficient deep learning inference based on model compression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2018, pp. 1695–1702.

[63] D. H. Le and B.-S. Hua, "Network pruning that matters: A case study on retraining variants," 2021, *arXiv:2105.03193*.

ALI ALMADAN received the M.S. degree in computer science from Wichita State University, Wichita, KS, USA, where he is currently pursuing the Ph.D. degree in computer science. His research interests include biometrics, computer vision, machine learning, and deep learning. He is also actively involved in research relative to edge computing and on-device AI.

AJITA RATTANI (Member, IEEE) received the postdoctoral degree from the Department of Computer Science and Engineering, Michigan State University, USA, and the Ph.D. degree in computer science engineering from the University of Cagliari, Italy. She is an Assistant Professor with the School of Computing, Wichita State University. She is the principal investigator to federal research grants from NSF and DOD. Her research interests include computer vision, image analysis, deep learning, machine learning, and biometrics. She was a recipient of the Best Paper and Poster Awards from IEEE IJCB 2014, IEEE HST 2017, 2019, and IAPR Biometric Summer School 2008. She is the Lead Editor of the Springer books titled *Adaptive Biometric Systems: Recent Advances and Challenges* and *Selfie Biometrics: Advances and Challenges*.