Filter-Pruning of Lightweight Face Detectors Using a Geometric Median Criterion *

Konstantinos Gkrispanis
CERTH-ITI
Thessaloniki, Greece, 57001
gkrispanis@iti.gr

Nikolaos Gkalelis[†]
CERTH-ITI
Thessaloniki, Greece, 57001
gkalelis@iti.gr

Vasileios Mezaris
CERTH-ITI
Thessaloniki, Greece, 57001
bmezaris@iti.gr

Abstract

Face detectors are becoming a crucial component of many applications, including surveillance, that often have to run on edge devices with limited processing power and memory. Therefore, there's a pressing demand for compact face detection models that can function efficiently across resource-constrained devices. Over recent years, network pruning techniques have attracted a lot of attention from researchers. These methods haven't been well examined in the context of face detectors, despite their expanding popularity. In this paper, we implement filter pruning on two already small and compact face detectors, named EXTD (Extremely Tiny Face Detector) and EResFD (Efficient ResNet Face Detector). The main pruning algorithm that we utilize is Filter Pruning via Geometric Median (FPGM), combined with the Soft Filter Pruning (SFP) iterative procedure. We also apply L1 Norm pruning, as a baseline to compare with the proposed approach. The experimental evaluation on the WIDER FACE dataset indicates that the proposed approach has the potential to further reduce the model size of already lightweight face detectors, with limited accuracy loss, or even with small accuracy gain for low pruning rates.

1. Introduction

Face detection technology is the backbone of numerous advanced applications, including but not limited to surveillance [10]. It has undergone significant evolution in the past decade. Especially with the rise of edge computing, where computations are performed on local devices with minimum computational power, efficient and compact face detectors have become necessary. While there is a need for these

models to be lightweight, so they can run on edge devices, they shouldn't compromise on accuracy.

Original model   Pruned model

Figure 1. The proposed approach is used to prune EResFD (an already very lightweight face detector) with 10% pruning rate. In this example, we see that the pruned model not only is more compact, but also detects faces, which were not detected with the original model. This may be attributed to the regularization effects of our approach for small pruning rates.

One promising avenue to achieve this balance is through network pruning [21]. Network pruning is a technique aimed at reducing the size of deep learning models without a significant drop in their performance. Over the years, various pruning techniques have been proposed and have achieved considerable success in tasks like image classification. Yet, their application and potential benefits in the domain of face detection remain largely uncharted. Specifically, to the best of our knowledge, only the method in [14] utilizes a pruning approach to a face detection network. However, in the above work a criterion is used to prune the "least important" filters in the layer, which is not always optimal [6].

Considering the above, this paper aims to examine the application of network pruning to face detectors, especially to the most lightweight architectures of them. Specifically, we adapt the Filter Pruning via Geometric Median (FPGM) [6] pruning algorithm and the Soft Filter Pruning (SFP) iterative procedure [5] to prune two already compact and small, in terms of parameters, face detectors, namely, EXTD (Extremely Tiny Face Detector) [25] and EResFD (Efficient ResNet Face Detector) [9]. FPGM identifies and prunes the filters with the "most redundancy", a principle that has

shown to provide improved performance over other pruning algorithms in the literature. Additionally, as baseline we compare with the widely used L1 Norm pruning criterion [11], as representative of the "less important" pruning principle. The experimental results on the WIDER FACE dataset [24] shows that the proposed approach has the potential to provide even more compact face detectors with competitive detection performance, especially when small pruning rates are used (e.g. see Figure 1).

In overall, we aim to provide a comparative analysis of the above algorithms and determine which of them, if any, offers a notable advantage in the face detection context. Through our research, we aspire to lay the foundations for more efficient and compact face detectors suitable for deployment on edge device, thereby broadening the horizons for real-time, resource constrained applications. In summary, we make the following contributions:

- We are the first to apply a redundancy-based pruning algorithm (FPGM) in order to prune the most redundant filters in lightweight face detection networks.

- The proposed approach yields a family of even more lightweight face detection networks that achieve superior detection accuracy in comparison to the previous state-of-the-art models of similar size.

The rest of the paper is structured as follows: The related work and proposed methodology are presented in Sections 2 and 3, respectively. Experimental results are discussed in Section 4 and conclusions are drawn in Section 5.

2. Related Work

2.1. Face Detection

In recent times, Convolutional Neural Networks (CNN) and other deep learning architectures, such as Transformers, have achieved notable success in a variety of computer vision tasks, including image classification, object detection and semantic segmentation.

Face detection, a sub-task of object detection, similarly benefits from the effectiveness of CNNs [1, 10]. Although Transformer-based architectures have shown state-of-the-art performance on object detection, the majority of state-of-the-art face detectors are extensions of CNN-based general-purpose object detectors [18]. PyramidBox [22] is based on Single-Shot Detector (SSD) [15]. RetinaFace [2], also based on SSD, is a robust single-stage face detector that performs pixel-wise face localization on various scales by leveraging both extra-supervised and self-supervised multi-task learning. TinaFace [30] is treating face detection as a one-class generic object detection and is using Deformable Convolution, Intersection over Union (IoU) aware branches and a Distance IoU Loss to enhance the model's capability.

YOLO5Face [20] is based on a set of state-of-the-art object detection models, named YOLO (You Only Look Once) known for their real-time processing capabilities. S3FD [27] is designed to efficiently detect faces across various scales, especially small ones, and achieves it by employing a scale-equitable framework (using anchors on a wide range of layers), a scale compensation anchor matching strategy and a max-out background. Dual Shot Face Detector (DSFD) [13] addressed the challenges in face detection through three main contributions: Feature Enhance Module, Progressive Anchor Loss and Improved Anchor Matching. Most of the above-discussed models are based on relatively heavy ResNet-50 and VGG16 networks. Thus, they are not suitable for resource-constrained environments, such as edge devices.

2.2. Lightweight Face Detectors

Designing lightweight face detectors that can operate on edge devices, and other resource-constrained environments, is an active research area. For instance, RetinaFace [2] has provided a lightweight version, implemented using MobileNet [8] as a backbone. Multi-task convolutional neural network (MTCNN) [23], used Multi-task Cascaded Convolutional Networks and employed a three stage cascade structure to predict face and landmark locations. Faceboxes [26] is based on SSD [15]; it combines Rapidly Digested Convolutional layers (RDCL) for swift input processing and Multiple Scale Convolutional Layers (MSCL) to handle faces of varying sizes. The authors in SCRFD [4], utilizes Sample Redistribution to augment training samples and Computation Redistribution to reallocate computational resources across the model's backbone and improve the computational efficiency. In [7], an anchor-free one-stage face detection method optimized for edge devices is presented. EXTD, proposed in [25], is able to detect faces at multiply scales by iteratively reusing a lightweight backbone network. In [9], EResFD emphasizes the effectiveness of standard convolution for lightweight face detection. That is, instead of using depthwise separable convolution (as in e.g. [25]), it is demonstrated that a ResNet with significantly reduced channels in combination with a standard convolution can achieve similar results. While many of the aforementioned detectors introduced novel techniques and achieved competitive performance on the WIDER FACE dataset [24], the latter two models have demonstrated superior performance in terms of a good trade-off between accuracy and model size. Consequently, in this study, we focus on EXTD and EResFD since, to the best of our knowledge, they offer the best accuracy-to-parameters-used trade-off. Our aim is to derive even smaller and more compact models without a significant drop in performance.

2.3. Network Pruning

Network pruning approaches can be roughly categorized to structured and non-structured. The latter, remove single weights, resulting in irregular weight sparsities, and thus require the use of specialized software and hardware to allow the efficient deployment of the pruned models. On the other hand, structured pruning methods remove entire model components, such as filters, yielding models that can be easily deployed. For this reason, structured pruning is receiving greater attention in the community [12, 16].

Due to the advantages described above, in this work we choose to perform structured pruning of very lightweight face detectors, and more specifically filter pruning. While filter pruning has been intensively investigated in several image classification tasks, the pruning of face detectors is a relatively unexplored topic. PruneFaceDet [14] is one of the few approaches in this domain. It employs a L1 regularization penalty imposed on the scaling factors of the Batch Normalization (BN) layers to perform structured pruning on the EagleEye face detector [28]. Due to this fact, this method can be only used on networks of specific structure, i.e., a one-to-one association of convolutional and BN layers is required. Additionally, pushing the values of the BN scaling factors towards zero, this approach is based on the "least importance" pruning principle. In contrary, here we use the FPGM pruning algorithm that formulates pruning from a redundancy reduction perspective, which has shown superior performance in several image classification tasks [3, 6].

3. Proposed Methodology

Consider an individual convolutional layer in a face detector with weight parameters,

$$F = [F_1, \ldots, F_n], \qquad (1)$$

where, $F_j \in \mathbb{R}^{k \times k \times c}$ is the jth filter with spatial size $k \times k$ and depth c, and n is the total number of filters in the layer. Based on the above formulation, the goal of the proposed approach is: given a filter pruning rate θ (common for all filters) prune the $n\theta$ filters in each layer of the face detector.

3.1. Backbone Networks

The first model that we choose to prune is the already compact EXTD [25], a state-of-the-art multi-scale face detector with an exceptionally small number of parameters. Unlike traditional multi-scale face detection models that extract feature maps of varying scales from a single backbone network, EXTD generates these feature maps by iteratively reusing a shared lightweight and shallow backbone network. This iterative sharing significantly reduces the model's parameters and provides abstract image semantics from higher network layers to the lower-level feature map. The key innovation is the ability to share the network in generating each feature map, which not only reduces the number of parameters but also enables the model to use more layers for detecting small faces. The architecture can be applied to both SSD and FPN (Feature Pyramid Network) based detection structures. Through experiments, it was demonstrated in [25] that this model can handle faces of various scales and conditions.

The second model we choose to prune is EResFD [9]. Here, it is shown that the combination of reduced channels with standard convolution can achieve similar results. EResFD consists of a modified ResNet backbone and feature enhancement modules: Separated Feature Pyramid Network and Cascade Context Prediction Module. To the best of our knowledge, this represents the face detector with the fewest parameters currently available, approximately 90,000 parameters. Thus, further pruning poses a significant challenge.

3.2. Pruning Algorithms

Redundancy-based pruning algorithms, i.e. algorithms that identify and discard the filters in a layer with the most similar characteristics, have shown superior performance in comparison to other criteria in the literature [3, 6]. To this end, we resort to the FPGM algorithm, which has been successfully used to prune different types of backbone networks and for different applications [3, 6]. Additionally, for comparison purposes, the well-known L1 Norm algorithm is utilized as our baseline. Both algorithms are briefly described in the following:

- **L1 Norm**: The L1 Norm pruning algorithm, as proposed in [12] and applied on various problems, e.g. [11], focuses on evaluating the significance of groups of weights (such as filters) in convolutional layers based on their L1 norm. It is based on the traditional "smaller-norm-less-important" criterion. Given filter F_j (1) its L1 norm is computed as:

$$\text{norm}(F_j, 1) = \sum_{i=1}^{ck^2} |f_{i,j}|, \qquad (2)$$

where, $f_{i,j}$ is the ith element of F_j. Filters with smaller L1 norm values, which indicate lower overall importance, are pruned. This methodology is inspired by the intuition that smaller-norm weight filters contribute less to the model's final prediction.

- **FPGM**: This algorithm has been originally proposed in [6]. It is based on the Geometric Median (GM), the classic robust estimator of centrality for data in Euclidean spaces, to prune redundant filters in a convolutional layer. Contrary to the L1 Norm algorithm, it

aims to identify filters that carry redundant information. As defined in [6], the GM of a set of filters, denoted as x^{GM}, is mathematically expressed as:

$$x^{\text{GM}} = \arg \min_{x \in \mathbb{R}^{k \times k \times c}} \sum_{j' \in [1,n]} \|x - F_{j'}\|_2, \quad (3)$$

where, x represents a filter in a layer and is used as a placeholder to denote any filter from the set of filters in that layer. Subsequently, the filter in the layer closest to this geometric median is given by:

$$F_{j^*} = \arg \min_{F_{j'}} \|F_{j'} - x^{\text{GM}}\|_2, \text{ s.t. } j' \in [1,n]. \quad (4)$$

To mitigate the computational cost of finding the GM, the following algorithm that identifies the filter minimizing the aggregate distance to all other ones is used:

$$F_{x^*} = \arg \min_x g(x), \text{ s.t. } x \in \{F_1, \ldots, F_n\}, \quad (5)$$

where the function $g(x)$ is defined as:

$$g(x) = \sum_{j'=1}^{n} \|x - F_{j'}\|_2. \quad (6)$$

The filter F_{x^*} that minimizes $g(x)$ can be then pruned with minimal impact on the network's redundancy. Experimental results in [6] validate the efficacy of FPGM, showing significant performance improvements on CIFAR-10 and ILSVRC2012 datasets. While the FPGM algorithm has been validated in several classification tasks (e.g. see [29], [3]), its utility in face detection is yet to be explored.

3.3. Soft Filter Pruning

We combine the FPGM algorithm with the Soft Filter Pruning (SFP) procedure [5] to prune the face detector in an iterative manner. Contrary to other methods in the literature that permanently prune filters, SFP allows pruned filters to be updated during subsequent model training. One main advantage offered by this approach is that it retains a larger model capacity since updating previously pruned filters provides a broader optimization space compared to permanently setting filters to zero. This larger optimization space allows the network to better learn from training data.

4. Experiments

4.1. Dataset and Metrics

The dataset used for training and evaluation in this work is the WIDER FACE dataset [24], a widely-used dataset for face detection research. It contains 32,203 images and embraces a wide variety of challenges, including large variations in scale, pose and occlusion. It is structured based on 60 event classes and the faces within it demonstrate significant variability in appearance, making it a challenging benchmark. Based on the the level of difficulty of the faces to be detected, the images are categorized into three subsets: Easy, Medium and Hard.

The performance evaluation is performed using the Mean Average Precision (mAP). We should note that this metric is typically used in conjunction with the WIDER FACE dataset to evaluate the performance of the different models across all three subsets.

4.2. Experimental Settings

Both EXTD [25] and EResFD [9], were evaluated using the following pruning rates $\theta = \{0.1, 0.2, 0.3, 0.4, 0.5\}$. The pruning rate refers to sparsity per pruned layer. All convolutional layers of the models are chosen to be pruned, except of those that are part of the detection head. The detection head is the last component of the model and is responsible for predicting the final bounding boxes and the classification of them - whether they contain a face or not; it is a crucial part of the network, hence we decided against pruning it.

The initial EXTD and EResFD models were created after we trained EXTD and EResFD from scratch with the setups that were described in their original papers. Subsequently, the iterative process of SFP [5] combined with the pruning algorithm (FPGM or L1 Norm) was conducted over 200 epochs to prune the different models. Specifically, a step learning rate schedule was adopted, with an initial learning rate of 1e-3. This was scaled down by a factor of 0.1 at epochs 50 and 100. For optimizer selection in each experiment, we strictly followed the configurations reported in the original publications. That is, the Stochastic Gradient Descent (SGD) with momentum 0.9, and the Adam optimizer, both with weight decay 5e-4 were used for the EXTD and EResFD, respectively. Upon reaching epoch 200, the SFP was halted, and further fine-tuning was conducted without updating the pruned weights during backpropagation (i.e. the pruned weights remained at zero). For this fine-tuning step, the learning rate was reset to 1e-3 for 5 epochs, followed by a decay to 1e-4 for the remaining 5 epochs of the process.

All the models and algorithms were implemented in PyTorch [19]. For the implementation of the FPGM and L1 Norm algorithms the NNI library [17] functions `FPGMPruner` and `L1NormPruner` were used, respectively.

4.3. Results

The pruned models produced using the proposed pruning approach are compared against the following methods that represent the state-of-the-art in face detectors (especially in lightweight ones, although the comparison is not
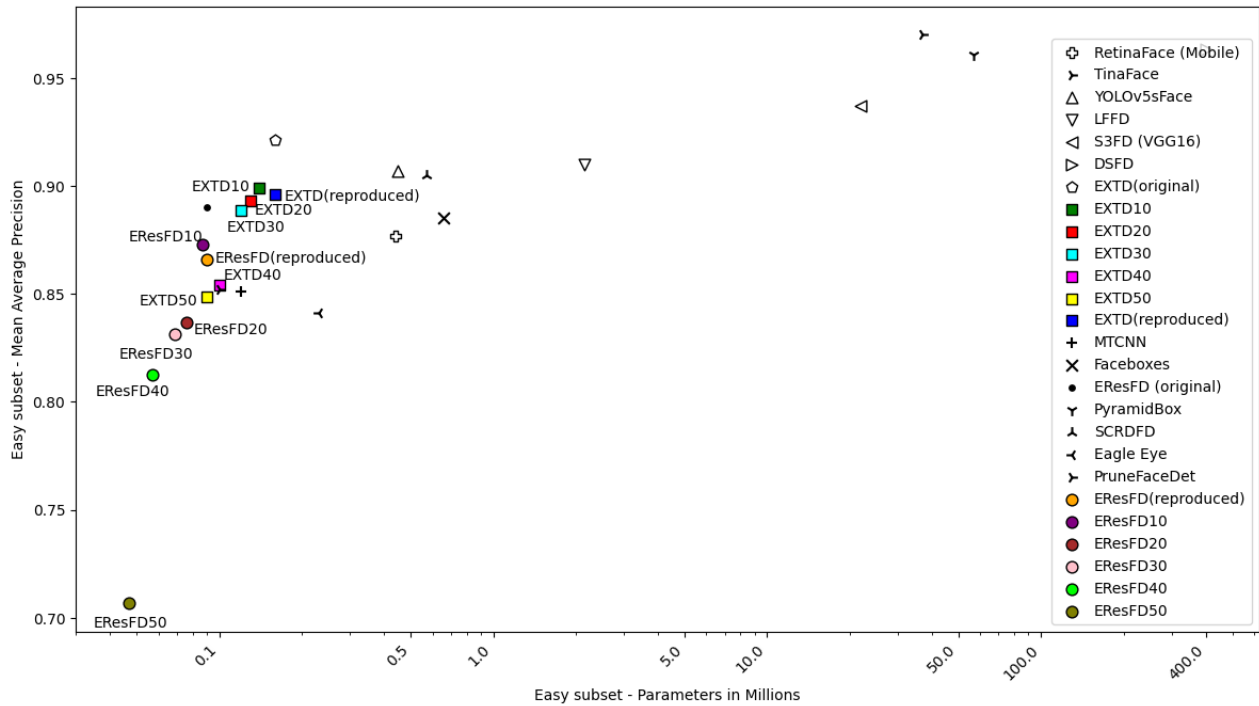
Figure 2. Model size and mAP across different face detectors on the Easy subset of WIDER FACE.
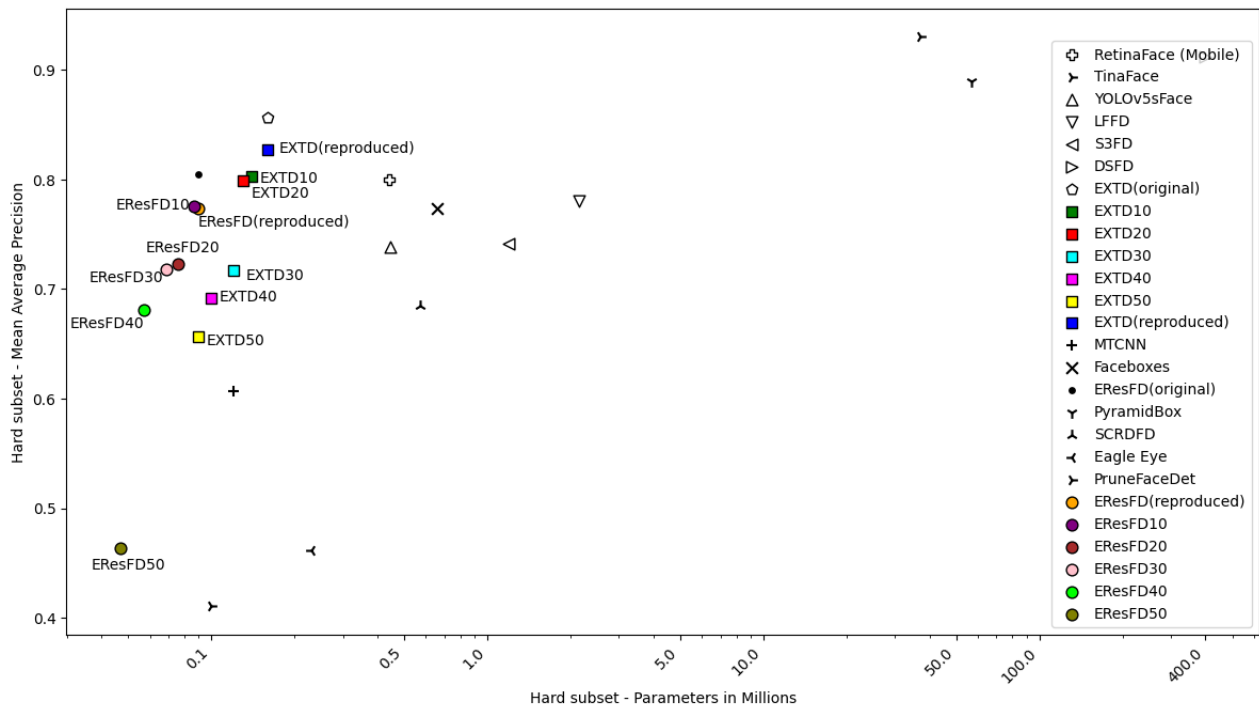

Figure 3. Model size and mAP across different face detectors on the Hard subset of WIDER FACE.

limited to lightweight models): RetinaFace (both with Mo-bileNet and ResNet50 as backbone) [2], TinaFace [30], Yolo5Face [20], LFFD [7], S3FD [27], Dual Shot Face Detector [13], MTCNN [23], FaceBoxes [26], PyramidBox

Table 1. Comparative results on EXTD between the proposed pruning approach (FPGM-based) and our baseline.

Method	Easy	Medium	Hard	# of Parameters	Real Sparsity
EXTD(original, from [25])	0.9210	0.9110	0.8560	162,352	0%
EXTD (reproduced)	0.8961	0.8868	0.8268		
FPGM 10%	0.8988	0.8828	0.8026	149,472	7.93%
L1 10%	0.8950	0.8766	0.7961		
FPGM 20%	0.8931	0.8789	0.7992	136,296	16.05%
L1 20%	0.8921	0.8766	0.7923		
FPGM 30%	0.8885	0.8588	0.7168	122,034	24.83%
L1 30%	0.8806	0.8522	0.6655		
FPGM 40%	0.8539	0.8213	0.6915	108,858	32.95%
L1 40%	0.8427	0.8068	0.6544		
FPGM 50%	0.8485	0.8118	0.6565	94,448	41.83%
L1 50%	0.8422	0.7971	0.6267		

Table 2. Comparative results on EResFD between the proposed pruning approach (FPGM-based) and our baseline.

Method	Easy	Medium	Hard	# of Parameters	Real Sparsity
EResFD(original, from [9])	0.8902	0.8796	0.8041	92,208	0%
EResFD (reproduced)	0.8660	0.8555	0.7731		
FPGM 10%	0.8728	0.8582	0.7757	87,368	5.25%
L1 10%	0.8470	0.8345	0.7410		
FPGM 20%	0.8369	0.8201	0.7230	76,677	16.84%
L1 20%	0.8263	0.8038	0.6723		
FPGM 30%	0.8311	0.8160	0.7175	69,746	24.36%
L1 30%	0.8218	0.8001	0.6663		
FPGM 40%	0.8124	0.7952	0.6807	57,055	35.95%
L1 40%	0.7603	0.7349	0.5800		
FPGM 50%	0.7103	0.6830	0.5254	47,284	48.72%
L1 50%	0.6992	0.6704	0.4824		

[22], SCRDFD [4], EagleEye [28], PruneFaceDet [14], and the original EXTD [25] and EResFD [9] models.

Figures 2 and 3 illustrate a comparative performance analysis of our pruned models against the referenced ones on the Easy and Hard WIDER FACE subsets, respectively. We should note that in these figures, EXTD and EResFD denote our reproduced results based on training and evaluating the corresponding models, EXTD(original) and EResFD(original) refer to the scores reported in the original publications, and EXTDθ and EResFDθ refer to the models produced by pruning them with rate $\theta\%$ using the proposed approach.

Tables 1 and 2 compare the proposed approach against our baseline approach, i.e., the SFP procedure combined with the L1 Norm pruning algorithm, across EXTD and EResFD, and for the five different pruning rates. In these tables, the Real Sparsity column reports the actual sparsity achieved using a specific pruning rate with the respective NNI pruner [17]. More specifically, the NNI pruner optimally prunes filters within each layer to ensure that the overall model sparsity does not exceed the user-defined threshold; this leads to real sparsity values being slightly lower than the target sparsity values as defined with the pruning rate input. On the other hand, the # of Parameters column provides the number of parameters of the model computed using a custom PyTorch [19] based routine. These tables, offer insights into the trade-off between model performance, sparsity and number of parameters, providing an overview of the effectiveness of the pruning strategies on those face detectors.

Finally, Figures 4 and 5 provide illustrative face detection examples of the original EXDT and EResFD models and the pruned models produced using the proposed approach with pruning rates 10% and 50%. From the obtained

Figure 4. Visualisation of face detection examples using the original EXTD model (first column) and its pruned variants, EXTD10 (second column) and EXTD50 (third column). We observe that for some examples, the model produced by the proposed approach with 10% pruning rate provides improved face detection performance.

results we conclude the following:

i) The family of models produced by the proposed approach provide a competitive detection performance with significantly reduced model size.

ii) Our pruned model with 10% sparsity exhibits a slightly improved performance when compared to our reproduced EXTD and EResFD. This improvement can be attributed to the pruning process acting as a form of regularization.

iii) From the overall results in Tables 1 and 2 we observe the superiority of the proposed approach (based on FPGM algorithm) over the baseline approach (based on the

L1 Norm algorithm), across all pruning rates.

iv) A similar conclusion to the above can be drawn by observing the qualitative face detection results in the middle column of both Figures 4 and 5. Specifically, we see that EXTD10 and EResFD10 are able to identify faces that would otherwise have been missed by the original model. Furthermore, our pruned model with 10% sparsity also manages to exclude some of the false positives that the original model produced, as it can be seen in the third row of Figure 5.

v) In overall, from the presented results it is evident that as sparsity increases, there is a pronounced decline in per-
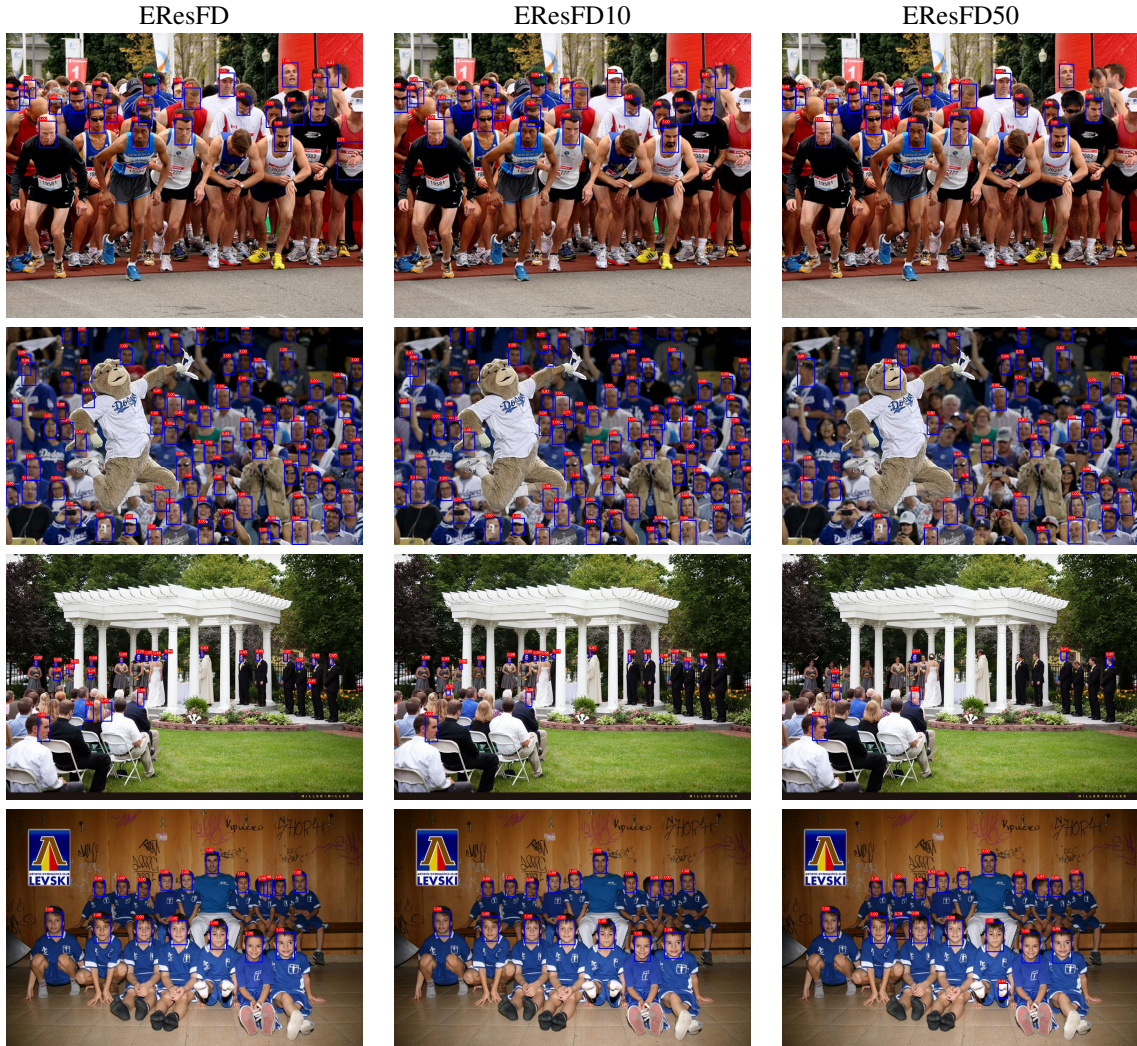
Figure 5. Visualisation of face detection examples using the original EResFD model (first column) and its pruned variants, EResFD10 (second column) and EResFD50 (third column). As with our EXTD examples (Figure 4), we see that in many cases, the model derived using the proposed approach with 10% pruning rate outperforms the original one in terms of detection performance.

formance within the 'Hard' subset. The drop in 'Easy' and 'Medium' subsets is more modest. By varying the pruning rate, our approach yields a family of lightweight detection models that represent different compromises between model size and detection accuracy. Furthermore, our findings suggest that FPGM has the potential to enhance the model's accuracy when enforcing small pruning rates.

5. Conclusions

Face detection is a rapidly evolving domain, and the demand for lightweight and efficient models that are suitable for edge devices is high. In this paper, we explored the potential of filter pruning techniques on two already compact face detectors, namely EXTD [25] and EResFD [9]. Our

research focused on two pruning algorithms: L1 Norm and Filter Pruning via Geometric Median (FPGM). Through experiments, we showed the superiority of the FPGM over the L1-Norm criterion. It is worth mentioning here, that the EResFD pruned model with 10% sparsity showcased a slight improvement in mAP, suggesting that the pruning process can act as a form of regularization. However, as sparsity increases, there is a notable decline in the mAP of the pruned models. By varying the pruning rate, we generated a family of even more compact face detectors for use in real-life applications where the model size is a critical factor.

References

[1] Laith Alzubaidi, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, J. Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie, and Laith Farhan. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, 2021. 2

[2] Jiankang Deng, Jia Guo, Yuxiang Zhou, Jinke Yu, Irene Kotsia, and Stefanos Zafeiriou. Retinaface: Single-stage dense face localisation in the wild. *arXiv preprint arXiv:1905.00641*, 2019. 2, 5

[3] Nikolaos Gkalelis and Vasileios Mezaris. Structured pruning of lstms via eigenanalysis and Geometric Median for mobile multimedia and deep learning applications. In *2020 IEEE International Symposium on Multimedia (ISM)*, pages 122–126, 2020. 3, 4

[4] Jia Guo, Jiankang Deng, Alexandros Lattas, and Stefanos Zafeiriou. Sample and computation redistribution for efficient face detection. *arXiv preprint arXiv:2105.04714*, 2021. 2, 6

[5] Yang He, Guoliang Kang, Xuanyi Dong, Yanwei Fu, and Yi Yang. Soft filter pruning for accelerating deep convolutional neural networks. *arXiv preprint arXiv:1808.06866*, 2018. 1, 4

[6] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via Geometric Median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 3, 4

[7] Yonghao He, Dezhong Xu, Lifang Wu, Meng Jian, Shiming Xiang, and Chunhong Pan. Lffd: A light and fast face detector for edge devices. *arXiv preprint arXiv:1904.10633*, 2019. 2, 5

[8] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 2

[9] Joonhyun Jeong, Beomyoung Kim, Joonsang Yu, and Youngjoon Yoo. EResFD: Rediscovery of the effectiveness of standard convolution for lightweight face detection. *arXiv preprint arXiv:2204.01209*, 2022. 1, 2, 3, 4, 6, 8

[10] Ashu Kumar, Amandeep Kaur, and Munish Kumar. Face detection techniques: a review. *Artificial Intelligence Review*, 52(2):927–948, 2019. 1, 2

[11] Aakash Kumar, Ali Muhammad Shaikh, Yun Li, Hazrat Bilal, and Baoqun Yin. Pruning filters with L1-norm and capped L1-norm for CNN compression. *Applied Intelligence*, 2021. 2, 3

[12] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016. 3

[13] Jian Li, Yabiao Wang, Changan Wang, Ying Tai, Jianjun Qian, Jian Yang, Chengjie Wang, Jilin Li, and Feiyue Huang. DSFD: dual shot face detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5060–5069, 2019. 2, 5

[14] Jingsheng Lin, Xu Zhao, Nanfei Jiang, and Jinqiao Wang. Prunefacedet: Pruning lightweight face detection network by sparsity training. In *Proceedings of the 2020 9th International Conference on Computing and Pattern Recognition*, ICCPR '20, page 181–186, New York, NY, USA, 2021. Association for Computing Machinery. 1, 3, 6

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer, 2016. 2

[16] Jian-Hao Luo, 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 (ICCV)*, Oct 2017. 3

[17] Microsoft. Neural Network Intelligence library, https://github.com/microsoft/nni, 2021. 4, 6

[18] Shervin Minaee, Ping Luo, Zhe Lin, and Kevin Bowyer. Going deeper into face detection: A survey. *arXiv preprint arXiv:2103.14983*, 2021. 2

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 4, 6

[20] Delong Qi, Weijun Tan, Qi Yao, and Jingfeng Liu. YOLO5Face: why reinventing a face detector. In *European Conference on Computer Vision*, pages 228–244. Springer, 2022. 2, 5

[21] R. Reed. Pruning algorithms-a survey. *IEEE Transactions on Neural Networks*, 1993. 1

[22] Xu Tang, Daniel K Du, Zeqiang He, and Jingtuo Liu. Pyramidbox: A context-assisted single shot face detector. In *Proceedings of the European conference on computer vision (ECCV)*, pages 797–813, 2018. 2, 6

[23] Jia Xiang and Gengming Zhu. Joint face detection and facial expression recognition with MTCNN. In *2017 4th International Conference on Information Science and Control Engineering (ICISCE)*, pages 424–427, 2017. 2, 5

[24] Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. WIDER FACE: A face detection benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5525–5533, 2016. 2, 4

[25] YoungJoon Yoo, Dongyoon Han, and Sangdoo Yun. EXTD: Extremely tiny face detector via iterative filter reuse. *arXiv preprint arXiv:1906.06579*, 2019. 1, 2, 3, 4, 6, 8

[26] Shifeng Zhang, Xiangyu Zhu, Zhen Lei, Hailin Shi, Xiaobo Wang, and Stan Z Li. Faceboxes: A CPU real-time face detector with high accuracy. In *2017 IEEE International Joint Conference on Biometrics (IJCB)*, pages 1–9. IEEE, 2017. 2, 5

[27] Shifeng Zhang, Xiangyu Zhu, Zhen Lei, Hailin Shi, Xiaobo Wang, and Stan Z Li. S3FD: Single shot scale-invariant face detector. In *Proceedings of the IEEE international conference on computer vision*, pages 192–201, 2017. 2, 5

[28] Xu Zhao, Xiaoqing Liang, Chaoyang Zhao, Ming Tang, and Jinqiao Wang. Real-time multi-scale face detector on embedded devices. *Sensors*, 2019. 3, 6

[29] Shangping Zhong, Wude Weng, Kaizhi Chen, and Jianhua Lai. Deep-learning steganalysis for removing document images on the basis of Geometric Median pruning. *Symmetry*, 2020. 4

[30] Yanjia Zhu, Hongxiang Cai, Shuhan Zhang, Chenhao Wang, and Yichao Xiong. Tinaface: Strong but simple baseline for face detection. *arXiv preprint arXiv:2011.13183*, 2020. 2, 5