

MobiFace: A Lightweight Deep Learning Face Recognition on Mobile Devices

Chi Nhan Duong¹, Kha Gia Quach¹, Ibsa Jalata³, Ngan Le², Khoa Luu³

¹ Computer Science and Software Engineering, Concordia University, Canada

² Electrical and Computer Engineering, Carnegie Mellon University, USA

³ Computer Science and Computer Engineering, University of Arkansas, USA

¹{dcnhan, kquach}@ieee.org, ²thihoanl@andrew.cmu.edu,

³{ikjalata, khoaluu}@uark.edu

Abstract

Deep neural networks have been widely used in numerous computer vision applications, particularly in face recognition. However, deploying deep neural network face recognition on mobile devices has recently become a trend but still limited since most high-accuracy deep models are both time and GPU consumption in the inference stage. Therefore, developing a lightweight deep neural network is one of the most practical solutions to deploy face recognition on mobile devices. Such the lightweight deep neural network requires efficient memory with small number of weights representation and low cost operators. In this paper, a novel deep neural network named MobiFace, a simple but effective approach, is proposed for productively deploying face recognition on mobile devices. The experimental results have shown that our lightweight MobiFace is able to achieve high performance with 99.73% on LFW database and 91.3% on large-scale challenging Megaface database. It is also eventually competitive against large-scale deep-networks face recognition while significant reducing computational time and memory consumption.

1. Introduction

Deep Convolutional Neural Network (CNNs) have revolutionized numerous machine learning applications with high accuracy, even in some applications, the performance exceeds human level of accuracy. Most of the computer vision applications, e.g. object detection [9, 39, 41, 8, 38, 40, 24], object classification [15, 6, 5], object segmentation [14, 27, 22, 23, 11], and modeling [7] etc., deploy CNNs with other Deep Neural Network framework to achieve the highest accurate performance. However, using deeper neural network with hundreds of layer and millions of parameters to achieve higher accuracy comes at cost. The networks require high computational resources beyond the ca-

pabilities of many mobile and embedded applications. To deploy such networks and to get higher performance, powerful GPUs with larger memory size are needed. As a result of this limitation, recently, there is an advancement in compressing deep neural networks known as compressed networks. Some familiar compressed networks are Mimic Networks [25, 35], Pruning [13, 12, 26], Depth-wise Convolution [16, 31], BinaryNets [20, 2, 29, 3]. These networks improve the speedup of the inference stage without significant lose of accuracy. Meanwhile, the improvements have not benchmarked in face recognition like in image classification and/or object detection. Moreover, face recognition problems are expected to be robust such as the very deep features of millions of facial subjects discriminated. In this problem, a noticeable number of neural layers should be included in the model unlike detection or classification.

In this work, our main contribution is a novel lightweight but high performance deep neural network for face recognition on mobile devices. Our network highly minimizes the number of operations and memory required while retaining the same accuracy mobile tailored computer vision models. In contrast to previous work, we present our contributions in MobiFace approach as follows: (1) we improve the well-known framework MobileFaceNet [1] by changing to further lighter-weight with higher accuracy performance. (2) The proposed method MobiNet is then applied in face recognition and optimized within an end-to-end deep learning framework. In addition, we extensively experiment our proposed MobiNet on both mobile-based network and large-scale deep-network on face recognition tasks with two state-of-the-art face recognition databases, i.e. Labeled Faces in the Wild (LFW) and large-scale challenging Megaface databases.

2. Prior Work

There has been a rising interest in lightweight deep network design mainly in tuning deep neural architectures to

strike an optimal balance between accuracy and performance for the past decade. There are many lightweight deep networks that can be categorized into designed compact modules and pruned networks, binarized networks, quantized networks and mimicked networks. Our mainly focus in this section will be the first two categories where our proposed framework mainly related with.

In MobileNet [16], Andrew et al. proposed depthwise separable convolutions to build light weight deep neural networks. In this work, the standard depthwise convolution is factorized into a depthwise convolution where a convolution applies to a single filter for each channel and a point-wise convolution where 1×1 convolution applied to combine the output from the depthwise convolution. VGG-16 with 138 million parameters and MobileNet with 4.2 million parameters achieve about the same accuracy on ImageNet [4]. Sandler et al. in [31] known as MobileNet-V2 further improves MobileNet on different benchmarks and multiple tasks. This work presents inverted residuals and linear bottlenecks [15] based on an inverted residual structure where the shortcut connections are between the thin bottleneck layers. In addition, the intermediate expansion layer uses lightweight depthwise convolutions to filter features as a source of non-linearity. It slightly improves the performance of MobileNet [16] where the accuracy is 70.6% to 72% on ImageNet [4] with only 3.4 million parameters. To reduce the memory and computational cost, FD-MobileNet [28] and MobileFaceNets [1] were presented based on MobileNet-V2 framework but still there are so much work to be done to effectively run deep learning framework on CPU.

Along side of compression methods, Han et al. [13] proposed to prune trivial connections by an absolute value without losing accuracy performance. Liu et al. [26] proposed a novel learning scheme to slim a network. In their work, instead of using absolute values of weights, they impose sparsity induced regularization on the scaling factor in Batch Normalization to slim the network and thus insignificant channels can be spotted. Overall, slimming a network proved to be successful in producing better results in ResNet [15], DenseNet [18] and VGG-16 [33]. However, for each pruned connection, a list of indices needs to be stored to memory, leading to a very low progress for both training and testing.

3. Our Proposed MobiNet

This section starts with introducing network design strategies to construct a lightweight deep network. Then, by adopting these strategies, the architecture of MobiFace for face recognition on mobile devices is introduced. Thanks to the concise and precise deep network architecture, the proposed framework is efficient in terms of small computational cost and high accuracy in comparison against

other deep networks on several large-scale face recognition databases.

3.1. Network Design Strategy

Bottleneck Residual block with the expansion layers.

The use of Bottleneck Residual block is introduced in [30] where a block consists of three main transformation operators, i.e. two linear transformations and one non-linear per-channel transformation. There are three key factors of this type of block: (1) the non-linear transformation to learn complex mapping functions; (2) the layer expansion with increasing number of feature maps in the inner layers; and (3) shortcut connections to learn the residual. Formally, given an input \mathbf{x} with the size of $h \times w \times k$, a bottleneck residual block can be represented as follows,

$$\mathcal{B}(\mathbf{x}) = [\mathcal{F}_1 \circ \mathcal{F}_2 \circ \mathcal{F}_3](\mathbf{x}) \quad (1)$$

where $\mathcal{F}_1 : \mathbb{R}^{w \times h \times k} \mapsto \mathbb{R}^{w \times h \times tk}$ and $\mathcal{F}_3 : \mathbb{R}^{w \times h \times k} \mapsto \mathbb{R}^{\frac{w}{s} \times \frac{h}{s} \times k_1}$ are the linear function represented by 1×1 convolution operator, and t denotes the expansion factor. $\mathcal{F}_2 : \mathbb{R}^{w \times h \times tk} \mapsto \mathbb{R}^{\frac{w}{s} \times \frac{h}{s} \times tk}$ is the non-linear mapping function which is a composition of three operators, i.e. ReLU, 3×3 depthwise convolution with stride s , and ReLU.

The residual learning connection is employed in a bottleneck block. This type of blocks is shown to have the capabilities of preventing manifold collapse during transformation and also increasing the expressiveness of the feature embedding [30].

Fast Downsampling. Under the limited computational resource of mobile devices, a compact network should *maximize the information transferred from the input image to output features while avoiding the high computational cost due to the large spatial dimensions* of feature maps. In the large-scale deep networks, the detail information flow is usually ensured by the *slow downsampling strategy*, i.e. the spatial dimensions are slowly reduced between blocks by downsampling operator. Consequently, these networks maintain so many feature maps with large spatial size and result in a heavy-size network. On the other hand, under the limited computational budgets, a light-weight network adopting that slow downsampling may suffer both issues of weak feature embedding and high processing time.

In these cases, *fast downsampling strategy* can be considered as an efficient replacement of the slow downsampling technique. In particular, in fast downsampling, the downsampling steps are consecutively applied in the very beginning stage of the feature embedding process to avoid large spatial dimension of the feature maps. Then in the later stage, more feature maps are added to support the information flow of the whole network. By this way, more complex mapping functions are learned to generate more details feature. Notice that, in this strategy, even more feature maps were added to the later feature, i.e. increase the

Table 1. Model Architecture for facial feature embedding. /2 means the operator has stride of 2. “Block” and “RBlock” indicate the Bottleneck Block and Residual Bottleneck block, respectively.

Input	Operator
$112 \times 112 \times 3$	3×3 Conv, /2, 64
$56 \times 56 \times 64$	3×3 DWconv, 64
$56 \times 56 \times 64$	Block 1 $\times \begin{cases} 1 \times 1 \text{ Conv, } 128 \\ 3 \times 3 \text{ DWconv, } /2, 128 \\ 1 \times 1 \text{ Conv, Linear, } 64 \end{cases}$
$28 \times 28 \times 64$	RBlock 2 $\times \begin{cases} 1 \times 1 \text{ Conv, } 128 \\ 3 \times 3 \text{ DWconv, } 128 \\ 1 \times 1 \text{ Conv, Linear, } 64 \end{cases}$
$28 \times 28 \times 64$	Block 1 $\times \begin{cases} 1 \times 1 \text{ Conv, } 256 \\ 3 \times 3 \text{ DWconv, } /2, 256 \\ 1 \times 1 \text{ Conv, Linear, } 128 \end{cases}$
$14 \times 14 \times 128$	RBlock 3 $\times \begin{cases} 1 \times 1 \text{ Conv, } 256 \\ 3 \times 3 \text{ DWconv, } 256 \\ 1 \times 1 \text{ Conv, Linear, } 128 \end{cases}$
$14 \times 14 \times 128$	Block 1 $\times \begin{cases} 1 \times 1 \text{ Conv, } 512 \\ 3 \times 3 \text{ DWconv, } /2, 512 \\ 1 \times 1 \text{ Conv, Linear, } 256 \end{cases}$
$7 \times 7 \times 256$	RBlock 6 $\times \begin{cases} 1 \times 1 \text{ Conv, } 512 \\ 3 \times 3 \text{ DWconv, } 512 \\ 1 \times 1 \text{ Conv, Linear, } 256 \end{cases}$
$7 \times 7 \times 256$	1×1 Conv, 512
$7 \times 7 \times 512$	512-d FC

number of channels, the computational cost is maintained to be low since the spatial dimensions of these feature maps are small.

3.2. MobiFace

In this section, we present a novel MobiFace approach, simple but efficient deep network for face recognition. Given an input facial image with the size of $112 \times 112 \times 3$, this light-weight network aims at maximizing the information embedded in final feature vector while maintaining the low computational cost. Inspired by the strategies presented in the previous section, the Residual Bottleneck block with expansion layers is adopt as the building block of MobiFace. Table 1 represents the main architecture of MobiFace that consists of one 3×3 convolutional layer, one 3×3 depthwise separable convolutional layer, followed by a sequence of Bottleneck blocks and Residual Bottleneck blocks, one 1×1 convolutional layer, and a fully connected layer. The structures of Residual Bottleneck blocks and Bottleneck blocks are very similar except a shortcut is added to connect the input and the output of the 1×1 convolution layer. Moreover, the stride s is set to 2 in Bottleneck blocks while that parameter is set to 1 in every layers of Residual Bottleneck blocks.

Moreover, we adopt the fast downsampling strategy in our network architecture by quickly reducing the spatial dimensions of layers/blocks with the input size larger than

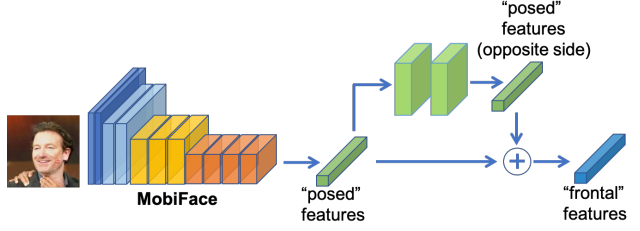


Figure 1. The structure of the proposed Flipped-MobiFace. The Flipped-MobiFace is an extended structure of MobiFace with two additional layers that learn to predict the deep features of the posed face in the opposite side.

14×14 . As one can easily see that given an input image with the size of $112 \times 112 \times 3$, the spatial dimension is reduced by half within the first two layers and become 8×8 smaller after the other 7 bottleneck blocks. The expansion factor is kept to 2 whereas the number of channels is double after each Bottleneck block in later feature embedding stage.

A batch normalization together with a non-linear activation are applied after each convolutional layer except the one marked as “Linear”. In our implementation, PReLU is used for the non-linear activation function due to its accuracy improvement over ReLU function. In the last layer of MobiFace, rather than employing the Global Average Pooling (GAP) layer as in previous approaches [17, 30, 1], we use the Fully Connected (FC) layer in the last stage of embedding process. Compared to GAP which treats very units in the last convolutional layer equally (*which is not very efficient since the information in the center pixel should play more important role than the one in the corner of the input*), the FC layer can learn different weights to these units and gain the information embedded in the final feature vector.

3.3. Flipped-MobiFace

Now we describe a simple but computationally efficient approach to further improve the performance of our proposed MobiFace against pose variation. It is worth noting that during learning process of MobiFace, the deeper layers tend to extract deep features that favours the linear separability between classes to support the linearization process in latent domain. In other words, after embedding the faces of different subjects from images to feature domain, they become linear separable. This linearity property of deep feature domain inspires us to a hypothesis that the averaging of the deep features of left- and right-posed faces within the same angle can approximate the deep features of the frontal face. Thus, given a face at a particular pose, by simply flipping the input image (i.e. approximate the face of the opposite side) and computing their average deep features, the effects of pose variation can be minimized in this new features. However, the extra computation will occur

when extract features of the mirror image in addition to the original image. Therefore, we proposed the flipped version of MobiFace, namely Flipped-MobiFace, that requires to run feature extraction network once. By this way, it can be efficiently deployed on mobile devices.

In particular, let I be the input posed face image and \bar{I} be the mirror image of I ; f_I and $f_{\bar{I}}$ be their deep features extracted by MobiFace structure, respectively. As illustrated in Fig. 1, the Flipped-MobiFace is an extended structure of MobiFace with two additional layers that learn to predict $f_{\bar{I}}$ from f_I directly without rerunning \bar{I} through the MobiFace structure again. Then the new feature $f = \frac{1}{2}(f_I + f_{\bar{I}})$ can be used as the final “frontal” features of I . We employ fully-connected layers with ReLU for these extended layers. In order to obtain the weights for these layers, for each training image I , we compute its flipped image \bar{I} by mirroring the face along x -axis. Then MobiFace is adopted to extract f_I and $f_{\bar{I}}$. Finally, the new layers are trained to obtain $f_{\bar{I}}$ from f_I . The loss function consists of two main terms: (1) ℓ_2 -norm between the ground-truth $f_{\bar{I}}$ and the predicted features; and (2) the cross-entropy loss to ensure the predicted features maintain the correct ID of the subject.

4. Experimental results

We first train the network using the cleaned training set of MS-Celeb-1M [10] including 3.8 million photos from 85K subjects. Then the trained network is evaluated on two common large-scale face verification benchmarks in unconstrained environments such as Labeled Faces in the Wild (LFW) [19], and Megaface [21] datasets. This training data has no overlapping with the testing data.

The databases that are used for training and testing are first described in next subsections. Then the comparisons between different models in terms of both accuracy and model sizes are represented. MobiFace can achieve very high performance, even competitive against other large-scale deep networks for face recognition.

4.1. Databases

MS-Celeb-1M [10] is introduced as a large-scale face dataset with 10 million photos of 100K celebrities. However, it also contains a large number of noisy image or wrong ID labels. To obtain a high-quality training data, the MS-Celeb-1M cleaned up the MS-Celeb-1M by computing the center feature of each subject and ranking their face images using the distance to identity center. The ones far from the center are automatically removed. Some manual checks are also employed. The refined MS-Celeb-1M consists of 3.8M photos from 85K identities.

Labeled Faces in the Wild (LFW) [19] is one of the common testing dataset for face verification. LFW consists 13,233 in-the-wild facial images of 5749 subjects collected

Table 2. Performance of Different face matching methods on LFW benchmark. * stands for our re-implementation. The inference time is measured on an Intel Core i7-6850K CPU @3.6GHz.

Methods	# Training images	Model Size	Speed (ms)	Accuracy (%)
Google-FaceNet [32]	200M	30MB	—	99.63%
CosFace [34]	5M	—	—	99.73%
LightCNN [36]	4M	50MB	—	99.33%
MobilenetV1 [16] *	3.8M	112MB	56ms	99.50%
MobileFaceNet [1] *	3.8M	4MB	30ms	99.48%
MobiFace	3.8M	9.3MB	26ms	99.72%
Flipped-MobiFace	3.8M	11.3MB	28ms	99.73%

from the web. The face variations include pose, expression and illuminations. According to the testing protocol of LFW, there are 6000 face pairs where half of them are positive pairs.

MegaFace [21] is one of largest publicly available testing dataset for face verification. This testing protocol is very challenging with million scale of distractors, i.e. subjects are not in the testing set. There are two main sets in Megaface, i.e. gallery and probe set. The gallery set is collected from Flickr photos and consists of more than 1 millions images from 690K identities. The probe set in Megaface are collected from two existing databases: Facescrub and FG-NET. As the Facescrub probe set aims at the robustness of face recognition systems on large number of identity, this set includes 100K photos of 530 subjects. Meanwhile, the FG-NET probe set focuses on the robustness of the system against age changing, with 1002 images of 82 identities from 0 to 69 years old. In this paper, we evaluate the performance of our light-weight network on the Facescrub probe set.

4.2. Implementation details

In the preprocessing step, MTCNN method [37] is applied to detect all faces and their five landmark points, i.e. two eye centers, nose and two mouth corner, in both training and testing photo. Then, using the information from five landmark points, each face is aligned and cropped into a template with the size of $112 \times 112 \times 3$. This template is then normalized into $[-1, 1]$ by subtracting the mean pixel value, i.e. 127.5, and divided by 128. For Flipped-MobiFace, we use two 512-dim fully-connected layers with ReLU.

During training stage, we adopt Stochastic Gradient Descent (SGD) optimizer with the batch size of 1024. The momentum parameter is set to 0.9. The learning rate is initialized to 0.1 and decreases by a factor of 10 periodically at 40K, 60K, and 80K iterations. The training stage is stopped at 100K iteration.

4.3. Face Verification accuracy

LFW benchmark. We first compare our MobiFace against many existing face recognition approaches includ-

Table 3. Performance of Different face matching methods on the refined version of Megaface benchmark with one million distractors. All the models are train with Large training dataset, i.e. > 0.5M. Model size is large when its size is greater than 20MB. * stands for our re-implementation.

Methods	Model Size	Accuracy (%)
MobilenetV1 [16] *	Large	92.65%
Google-FaceNet [32]	Large	86.47%
MobileFaceNet [1]*	Small	90.71%
MobiFace (Ours)	Small	90.9%
Flipped-MobiFace (Ours)	Small	91.3%

ing both large-scale deep models and small-scale one. Table 2 represented the performance of different matching methods on LFW benchmark. From these results, one can easily see that our MobiFace achieves 99.72% with the model size of only 9.3MB. With this performance, our MobiFace outperforms other small-size models and achieves competitive results to other large-scale deep models. MobiFace also outperforms most of other approaches in Table 2.

Megaface benchmark. We further validate the performance of our light-weight MobiFace on the challenging Megaface benchmark against millions of distractors. Table 3 illustrates the verification results of different methods on Megaface. The accuracy is reported on the True Accepted Rate (TAR) at the False Accepted Rate (FAR) of 10^{-6} . These results again emphasize the performance of our MobiFace when it outperforms the other light-weight MobileFaceNet model. Compared to other large-scale deep networks, our MobiFace has the advantages of both comparable performance to these models while maintaining low computational cost. Therefore, our MobiFace is easy to be deployed on mobile devices.

5. Conclusion

This paper has reviewed different lightweight deep network structures and approaches for mobile devices where the computational resource is very limited. Inspired by different network design strategies, this paper has further presented a novel simple but high-performance deep network for face recognition, named MobiFace. Experiments on two common large-scale face verification benchmarks with photo in unconstrained environment have shown the efficiency of our MobiFace in terms of both accuracy and small model size. Although the model is very small, its performance on both testing benchmarks is competitive against other large-scale deep face recognition network.

References

[1] S. Chen, Y. Liu, X. Gao, and Z. Han. Mobilefacenets: Efficient cnns for accurate real-time face verification on mobile devices. *arXiv preprint arXiv:1804.07573*, 2018.

[2] M. Courbariaux and Y. Bengio. Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1. *CoRR*, abs/1602.02830, 2016.

[3] M. Courbariaux, Y. Bengio, and J. David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *NIPS*, pages 3123–3131, 2015.

[4] J. Deng, W. Dong, R. Socher, L. jia Li, K. Li, and L. Fei-fei. Imagenet: A large-scale hierarchical image database. In *In CVPR*, 2009.

[5] C. N. Duong, K. Luu, K. Quach, and T. Bui. Beyond principal components: Deep boltzmann machines for face modeling. In *CVPR*, 2015.

[6] C. N. Duong, K. Luu, K. Quach, and T. Bui. Longitudinal face modeling via temporal deep restricted boltzmann machines. In *CVPR*, 2016.

[7] C. N. Duong, K. Luu, K. Quach, and T. Bui. Deep appearance models: A deep boltzmann machine approach for face modeling. *Intl Journal of Computer Vision (IJCV)*, 2018.

[8] C. N. Duong, K. G. Quach, K. Luu, T. H. N. Le, and M. Savvides. Temporal non-volume preserving approach to facial age-progression and age-invariant face recognition. In *ICCV*, 2017.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. 2014.

[10] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision*, pages 87–102. Springer, 2016.

[11] M. S. H. N. Le, R. Gummadi. Deep recurrent level set for segmenting brain tumors. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pages 646–653. Springer, 2018.

[12] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. *CoRR*, abs/1510.00149, 2015.

[13] S. Han, J. Pool, J. Tran, and W. J. Dally. Learning both weights and connections for efficient neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1, NIPS’15*, pages 1135–1143, Cambridge, MA, USA, 2015. MIT Press.

[14] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[16] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[17] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[18] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *CVPR*, pages 2261–2269, 2017.

- [19] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In *Workshop on faces in 'Real-Life' Images: detection, alignment, and recognition*, 2008.
- [20] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks. In *NIPS*, pages 4107–4115, 2016.
- [21] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4873–4882, 2016.
- [22] H. N. Le, C. N. Duong, K. Luu, and M. Savvides. Deep contextual recurrent residual networks for scene labeling. In *Journal of Pattern Recognition*, 2018.
- [23] H. N. Le, K. G. Quach, K. Luu, and M. Savvides. Reformulating level sets as deep recurrent neural network approach to semantic segmentation. In *Trans. on Image Processing (TIP)*, 2018.
- [24] H. N. Le, C. Zhu, Y. Zheng, K. Luu, and M. Savvides. Robust hand detection in vehicles. In *Intl. Conf. on Pattern Recognition (ICPR)*, 2016.
- [25] Q. Li, S. Jin, and J. Yan. Mimicking very efficient network for object detection. *2017 IEEE Conference on CVPR*, pages 7341–7349, 2017.
- [26] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang. Learning efficient convolutional networks through network slimming. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2755–2763, 2017.
- [27] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*.
- [28] Z. Qin, Z. Zhang, X. Chen, C. Wang, and Y. Peng. Fd-mobilenet: Improved mobilenet with a fast downsampling strategy. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 1363–1367. IEEE, 2018.
- [29] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV (4)*, volume 9908 of *Lecture Notes in Computer Science*, pages 525–542. Springer, 2016.
- [30] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.
- [31] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *CoRR*, abs/1801.04381, 2018.
- [32] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.
- [33] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.
- [34] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. Cosface: Large margin cosine loss for deep face recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [35] Y. Wei, X. Pan, H. Qin, and J. Yan. Quantization mimic: Towards very tiny cnn for object detection. *CoRR*, abs/1805.02152, 2018.
- [36] X. Wu, R. He, Z. Sun, and T. Tan. A light cnn for deep face representation with noisy labels. *IEEE Transactions on Information Forensics and Security*, 13(11):2884–2896, 2018.
- [37] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.
- [38] Y. Zheng, C. Zhu, K. Luu, H. N. Le, C. Bhagavatula, and M. Savvides. Towards a deep learning framework for unconstrained face detection. In *BTAS*, 2016.
- [39] C. Zhu, Y. Ran, K. Luu, and M. Savvides. Seeing small faces from robust anchor’s perspective. In *CVPR*, 2018.
- [40] C. Zhu, Y. Zheng, K. Luu, H. N. Le, C. Bhagavatula, and M. Savvides. Weakly supervised facial analysis with dense hyper-column features. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, 2016.
- [41] C. Zhu, Y. Zheng, K. Luu, and M. Savvides. Enhancing interior and exterior deep facial features for face detection in the wild. In *Intl Conf. on Automatic Face and Gesture Recognition (FG)*, 2018.