# CHILD FACE RECOGNITION SYSTEM USING MOBILEFACENET

**Shun Lei Myat Oo[(1)], Aung Nway Oo[(2)]**
University of information technology, Myanmar

Email: [(1)]shunleimyatoo@uit.edu.mm, [(2)]aungnwayoo@uit.edu.mm

## ABSTRACT

Face recognition is a kind of identifying people in image. It matches the database of known faces and input image of unknown face. Deep learning is one of the state-of-art technologies which achieve state-of-art performance on face recognition. In this paper, we develop child face recognition using MobileFaceNet. MobileFaceNet is efficient *Convolutional Neural Network* (CNN) models and it uses more than 1 million parameters. MobileFaceNet is used for feature extractions. Since MobileFaceNet is one of the types of light weights models, we can apply this face recognition system on mobile and embedded devices. Dlib is used for preprocessing and K-Nearest Neighbors (KNN) is used for classification process. MobileFaceNet is trained by ArcFace loss and it achieve the 96% accuracy on child face dataset.

**KEYWORDS:** *Face recognition, MobileFaceNet, Convolutional Neural Network, deep learning, K-Nearest Neighbors*

## 1. INTRODUCTION

Nowadays, face recognition system plays as important authentication system. The research area of face recognition is still active and open area. It is used in video surveillance, access control, social media and finding a person in crowded area. Some mobile devices use also face recognition technology for many purposes such as face unlock. Face recognition technology play a role for children life. It is used for finding the missing children, school safety and school social network activities.

Traditional face recognition systems face several errors in real time applications. Deep learning is a state-of-art technology which acts like a human brain, i.e., learn by itself. Over the past years, deep learning is not so popular in visual recognition due to the limited amount of hardware resources. Nowadays, large amount of data and powerful computational resources is available for deep learning. In traditional method, people need domain knowledge for visual recognition. Analyst and scientist create features based on their business or domain knowledge in traditional way. Deep learning eliminates the manual features extraction process and it appear as automated features engineering.

In early 1990s and late 2000s, holistic learning approach and local handcrafted approach dominated face recognition area respectively [3]. This approach faced worse result for unconstrained facial changes, lighting, expression and pose since it used two or three feature descriptors. In 2012, AlexNet won the ImageNet competition by reducing the top-5 error from 26% to 15.3% on ImageNet, achieving a top-1 error rate of 37.5% using a deep learning technique and it overcomes the previous methods performance. Convolutional neural network (CNN) is one of the types of deep learning methods. I use multiple layers feature descriptors for feature extraction and transformation. Generally, early layers extract the basic features of face and later layers extract the detail features of face. DeepFace and DeepID achieve the high accuracy on LFW (Labeled Face in-the-Wild) dataset around 90% in 2012. In 2015, FaceNet, was trained by triplet loss, achieves 99.63% on LFW and 95.12% on YouTube Faces DB. It achieves state-of-art face recognition performance and use 128 embeddings per face.

In this paper, we use MobileFaceNet for extracting features from faces. Since it is a lightweight model, our system can work fast and efficient on embedded devices. MobileFaceNet achieve significantly superior accuracy as well as more than 2 times actual speed up over MobileFaceNetV2 due to its global depth wise convolution [2]**.** It is trained by Additive Angular Margin Loss (ArcFace) to obtain highly discriminative features for face recognition. Compared to softmax loss, it optimize maximal diversity for inter-class samples and similarity for intra class. ArcFace consistently outperforms the state-of-art and can easily implemented with low computational overhead. Most computational cost of triplet loss is choosing the pairs, anchor, positive and negative. We use K-Nearest Neighbors classifier for classify features vectors by calculating the distance between them.

## 2. Proposed Approach

### 2.1 System flow

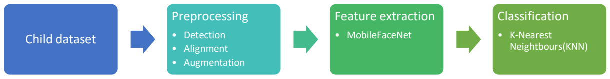


Figure 1. System flow of proposed system

Child face raw dataset are preprocessed using an face detection landmark algorithm attempted by dlib. Features are extracted from preprocessed images by using CNN based MobileFaceNet model. K-Nearest Neighbors (KNN) are used to classify children based on embeddings distances. Figure 1 illustrate the system flow of my proposed system.

### *2.1.1 Preprocessing*

Preprocessing is the critical part of the purposed system. Train set and test set are preprocessed to extract a face from image using the Face Landmark detection approach attempted by dlib. Dlib is implemented based on Ensemble of Regression Trees (ERT) presented by Vahid Kazemi and Josephine Sullivan in 2014[1]. Face landmark localization or face alignment are made using ensemble of regression trees which estimate the face's landmark positions from a spare subset of pixel intensities [1]. Cascade of regressors refine the estimated point with an iterative process. Regressors produces a new estimated from the previous one to reduce the alignment error of estimated points in each iteration. This achieve the real time performance with high quality predictions. It takes about 2 or 3 milliseconds to detect(align) a set of 68 landmarks per face.

### *2.1.2 Feature extraction (MobileFaceNet)*

One of the lightweight and extremely efficient CNN model, MobileFaceNet, which give high-accuracy for real-time face recognition on mobile and embedded device [2]. It use less than 1 million parameters. It is 4.0MB size and gives high accuracy compared with others hundreds MB size model[2]. Input aligned face images is 112x112 size and normalize each pixel in RGB image by subtracting 127.5 and then divided by 128. Finally, it map the each aligned face to a feature vector.

Compared with MobileNetV2, MobileFaceNet use stride=1 with input size 112x112 instead of stride=2 with input size 224x224, where the latter layers leads to very poor accuracy. MobileFaceNet use collection of bottleneck layers based on the architecture of depth wise separable convolutions found in MobileNetV2. Depth wise sepereable convolution effectively reduces computation compared to traditional layers by almost a factor of $k^2$ , where k is kernel size [5]. MobileNetV1 started to use depth wise separable convolution with two parts; depth wise convolution and point wise convolution [4]. MobileNetV2 is based on inverted residuals block with linear bottleneck [5]. A bottleneck residual block contains three layers convolution with shortcut connection bottleneck (see in Figure 2). Expansion factors in MobileFaceNet are much smaller than those in MobileNetV2 [2]. Bach normalization is applied during training process. A linear 1x1 convolutional layer follows linear global depth wise convolutional layer as feature output layer. MobileFaceNet use global depth wise convolution layer rather than a global average pooling layer or a fully connected layer to output a discriminative feature vector after a last convolutional layer of a face feature embedding. The detail architecture sees in Table 1.

Table 1. Architecture of MobileFaceNet. c is the number of output channels. t is the expansion rate of channel. n is the blocked repeated time. s is the stride.

| Input | Operator | *t* | *c* | *n* | *s* |
|---|---|---|---|---|---|
| $112^2 \times 3$ | conv3x3 | - | 64 | 1 | 2 |
| $56^2 \times 64$ | depthwise conv3x3 | - | 64 | 1 | 1 |
| $56^2 \times 64$ | bottleneck | 2 | 64 | 5 | 2 |
| $28^2 \times 64$ | bottleneck | 4 | 128 | 1 | 2 |
| $14^2 \times 128$ | bottleneck | 2 | 128 | 6 | 1 |
| $14^2 \times 128$ | bottleneck | 4 | 128 | 1 | 2 |
| $7^2 \times 128$ | bottleneck | 2 | 128 | 2 | 1 |
| $7^2 \times 128$ | conv1x1 | - | 512 | 1 | 1 |
| $7^2 \times 512$ | linear GDConv7x7 | - | 512 | 1 | 1 |
| $1^2 \times 512$ | linear conv1x1 | - | 128 | 1 | 1 |

A depth wise convolution layer (GDConv layer) with kernel size equaling the input size, pad = 0, stride = 1 [2]. The output of GDConv layer is computed as:

$$G_m = \sum_{i,j} K_{i,j,m} F_{i,j,m}$$

Where F is the input feature map of size W x H x M, K is the depth wise convolution kernel of size W x H x M, G is the output of size 1 x 1 x M, $G_m$ is m^th channel in G. (i,j) denotes the spatial position in F and K and m refer channel index. GDConv layer has W.H.M computational cost.

### *2.1.3 Additive Angular Margin (ArcFace) Loss*

MobileFaceNet are trained by Additive Angular Margin Loss (ArcFace loss) [6] which obtain highly discriminative features for face recognition. Arcface loss is proposed by Jiankang Den, Jia Guo and Niannan Xue in 2019[6]. ArcFace overcome drawbacks of previous loss functions. For softmax loss, learned features aremseperable for closed-set classification problem but not discriminative enough for open-set face recognition problem [6]. For triplet loss, combinatorial explosion of face triplets especially for large datasets leads to increase in number of iteration steps and semi-hard triplet mining is difficult for effective model training because most of its computation cost are large due to find the a pair of triplets,.i.e., anchor, positive and negative. ArchFace loss achieves state-of-art performance and also easy to implement in computational-graph-based deep learning frameworks [6]. It can easily converge on any training dataset and have a stable performance Its computational complexity is low during training. ArcFace has a constant linear angular margin throughout the whole interval whereas CosFace[7] and SphereFace[8] only have non-linear angular margin[6].

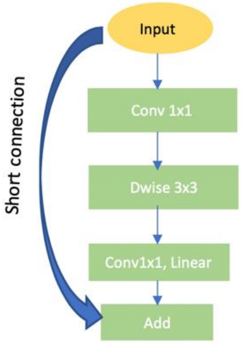


Figure2.Bottleneck residual block

### *2.1.4 K-Nearest Neighbours (KNN)*

This classify the features vectors generated form MobileFaceNet by calculating the distance between embeddings. It calculates the distance of an unknown feature from its k closest neighbors features and then take the class that appeared most times. The smaller the distance is the most similar between these two data. KNN algorithm is generally based on features similarity. Generated features from MobileFaceNets are constructed as a graph and calculate the distance of out-of-sample features from others features inside graph. We use simple classification algorithm because optimal features for faces had already generated from MobileFaceNet and the task of classification algorithm is only calculated distances. KNN algorithm implemented by scikit-learn is fast and easy to perform classification task.

## 3. EXPERIMENT

### *3.1 Datasets*

We used private child datasets for training and testing. Child data are collected from schools, sport teams and social media. We practically took videos and photos of children from schools and other sport teams. We also got some celebrated children photos from social media. Dataset mixed with real world children photos and celebrated children. It includes about 300 children face and 30 photos for each. Facial emotion is generally normal, smile, laugh. Facial position is frontal, left and right sides. Age is range from 5 to 14 years.

### *3.2 Experimental settings*

We use face landmark detection algorithm offered by dlib to detect, align and crop 112x112 size images using 68 landmark points. We use CNN architecture, MobileFaceNet. We used a pretrained model, not learn from scratch. We applied fine-tuning, an approach of transfer learning, for a pretrained model. ArcFace loss are used to train the model and convolution layer is followed by batch normalization. We train our model on Geforce GTX 1080 GPU. The training process is finished at 8.25K iterations. We start our learning rate at 0.005 and generally divided by 10 after 10i epochs, where i = {1, 2, 3, …}. We follow to set feature scale s to 64 and choose the angular margin m of ArcFace at 0.5 [6]. We use adam as optimization algorithm. We use last output layer as global depthwise convolution to generate 128 embedding per face. Weights decay is 5e-5.

### *3.3 Evaluation results*

We get the 99% accuracy on our private child dataset (See in Figure 3).

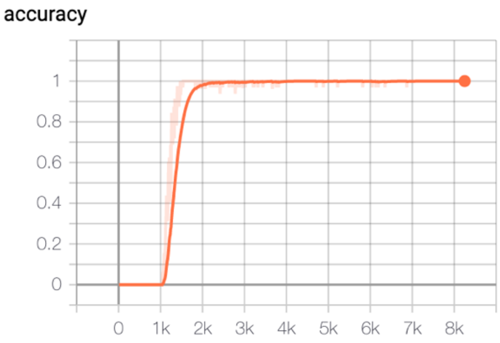


Figure 3. Accuracy graph of model. Column represents the value of accuracy and row represents number of iterations. Accuracy obtained 99%.

Loss value was nearly 0.007 on our private child dataset (See in Figure 4). We can see the 128 embedding features generated from MobileFaceNet model by using PCA (See in Figure 5).

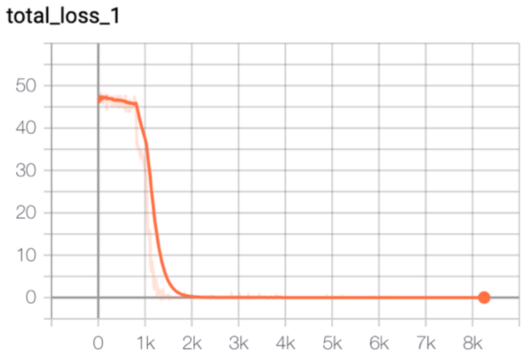


Figure 4. Total loss graph of model. Column represents the value of loss and row represents number of iterations. Loss value obtained nearly 0.007.

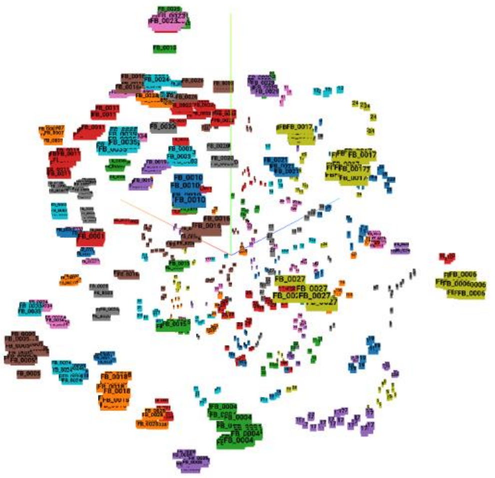
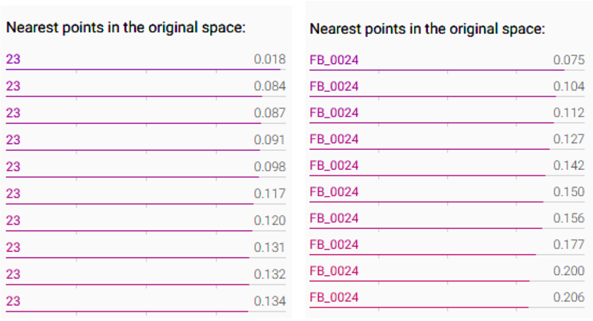


Figure 5. PCA of 128 feature vectors generated from MobileFaceNet model. One color represents one class.

One color represents one class. Same class has same color. We can see the distance of same classes and different classes from this graph, united feature vectors were same classes (same color) and different classes had a certain distance from each other (See in Figure 6(a) and (b). Different people must have far distance and same people must have close distance.



Nearest points in the original space:

| | |
|---|---|
| 23 | 0.018 |
| 23 | 0.084 |
| 23 | 0.087 |
| 23 | 0.091 |
| 23 | 0.098 |
| 23 | 0.117 |
| 23 | 0.120 |
| 23 | 0.131 |
| 23 | 0.132 |
| 23 | 0.134 |

Nearest points in the original space:

| | |
|---|---|
| FB_0024 | 0.075 |
| FB_0024 | 0.104 |
| FB_0024 | 0.112 |
| FB_0024 | 0.127 |
| FB_0024 | 0.142 |
| FB_0024 | 0.150 |
| FB_0024 | 0.156 |
| FB_0024 | 0.177 |
| FB_0024 | 0.200 |
| FB_0024 | 0.206 |

(a) Class name is 23. (b) Class name is FB_0024.

Figure 6. 10 closest neighbors' class name and distance of a selected class. (a) Selected class is 23 and its 10 closest neighbors are also 23. (b) Selected class is FB_0024 and its 10 closest neighbors are also FB_0024.

## 4. CONCLUSIONS

This research shows that MobileFaceNet trained by ArcFace loss achieves better accuracy on child face dataset with few parameters. Its computational complexity is efficient. Parameters tuning and iteration steps determine the results of model and classification. This proposed system can give extreme efficiency for real-time face recognition on mobile and embedded devices.

## ACKNOWLEDMENT

I would like to express our gratitude to all people for their suggestion and supports throughout this research work. I would like to express my thanks to Codigo (Myanmar) company which allows me to use their child dataset, and supports hardware and software resources for my research. I also thank Artificial Intelligence team of Codigo (Myanmar) company for their suggestion and supports. Finally, I also thank all of my colleague for their participation and contribution for this study.

## REFERENCES

[1] Vahid Kazemi and Josephine Sullivan, "One Millisecond Face Alignment with an Ensemble of Regression Trees", CVPR, 2014, pp. 1-2.

[2] Sheng Chen, Yang Liu, Xiang Gao and Zhen Han, "MobileFaceNets:Efficient CNNSs for Accurate Real-Time Face Verification on Mobile Devices", arXiv, 2018, pp.1-7.

[3] Mei Wang and Weihong Deng, "Deep Face Recognition: A Survey", arXiv, 2019, pp. 1-3.

[4] Andrew G. Howard Weijun Wang, Menglong Zhu Tobias Weyand, Bo Chen Macro Andreetto and Dmitry Kalenichenko Hartwig Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Version Applications", arXiv, 2017, pp. 1-4.

[5] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov and Liang-Chieh Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks", arXiv, 2018, pp. 1-5.

[6] Jiankang Deng, Jia Guo and Niannan Xue, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition", arXiv, 2019, pp. 1-6.

[7] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li and Wei Liu, arXiv, 2018, pp. 1-2.

[8] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj and Le Song, arXiv, 2018, pp. 1-2.