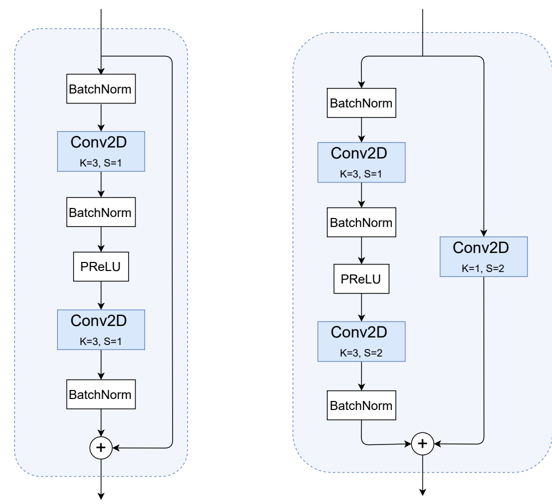
Supplementary Material for
The Impact of Racial Distribution in Training Data on Face Recognition Bias: A Closer Look

Manideep Kolla
HyperVerge Inc.
manideep@hyperverge.co

Aravinth Savadamuthu
HyperVerge Inc.
aravinth.muthu@hyperverge.co

1. Backbone Architecture

We use a modified ResNet-50 network [3, 1] with a total of 53 convolution layers as the backbone for our experiments. In this modified ResNet architecture, Rectified Linear Unit (ReLU) activation functions [5] from the original ResNet implementation [3] is replaced with the Parametric Rectified Linear Unit (PReLU) activation function [2] at all places in the network. In addition, the modified residual block has an additional batch normalization layer at the beginning. Apart from these two modifications, the rest of the layers and the overall network architecture are the same as that of the ResNet-50 network from [3]. The network consists of four stages, with 3 residual blocks in the first stage, 4 in the second, 14 in the third, and 3 in the fourth, and each stage begins with a downsampling block similar to [3]. So, among the 53 convolutional layers in the network, four convolutional layers with a stride of 2 and kernel size of 1 are used to downsample the feature maps in four stages. Fig. 1 depicts the modified residual block and the downsampling block. Our modified ResNet backbone consists of 43.6 million parameters. The output of the backbone network is a 512-dimensional vector that serves as the embedding vector of an input face.



(a) Residual Block (b) Downsampling block

Figure 1: The modified Residual blocks used in the backbone. Left: The modified residual block. Right: The modified downsampling block. Here, K is the kernel size ($K \times K$) of the 2D convolution layers, and S is the stride size.

Race	Faces	Identities
African	324,374	7000
Asian	325,155	7000
Caucasian	326,482	7000
Indian	274,855	7000

Table 1: Number of faces and identities in each racial cohort of the BUPT-BalancedFace dataset.

2. Training Data Settings

We use the BUPT-BalancedFace dataset [8] to prepare 15 distinct training datasets by choosing combinations of all faces from one race at a time, two races at a time, three races at a time, and all four races at a time. In each setting, we use all the faces available in the BUPT-BalancedFace dataset from those particular races present in the setting. For example, for the African+Asian setting, we use all the 324,374 faces (7,000 identities) from the African subset and all the 325,155 faces (7,000 identities) from the Asian subset of the BUPT-BalancedFace dataset, totaling 649,529 faces (14,000 identities) for training. Tab. 1 details the racial split of the BUPT-BalancedFace dataset.

3. False Match and Face Non-match Rates

A false match arises when two faces belonging to different people (a negative pair or a non-mated pair) are predicted as a match by a face recognition algorithm. It can also be referred to as false positive or false acceptance. A false non-match occurs when two faces belonging to the same person (a positive pair or a mated pair) are predicted as a mismatch. It can also be referred to as false negative or false rejection. The false match rate (FMR) is the ratio between the number of false matches and the total number of negative pairs available. False match rate is also called false positive rate or false acceptance rate. The false non-match rate (FNMR) is the ratio between the number of false non-matches and the total number of positive pairs available. False non-match rate is also called false negative rate or false rejection rate.

The false match rate (FMR) is expressed as,

$$FMR = \frac{\text{False Positives}}{\text{False Positives + True Negatives}}$$

The false non-match rate (FNMR) is expressed as,

$$FNMR = \frac{\text{False Negatives}}{\text{False Negatives + True Positives}}$$

4. Hardware and Software

We use a node with four Nvidia RTX 2080Ti GPUs with approximately 12 GB VRAM in each for training the face recognition models. We use the distributed data parallel (DDP) training from PyTorch to train the models [6].

We use PyTorch [7] as our deep learning training framework with native automatic mixed-precision mode. We use Distributed Data-Parallel (DDP) paradigm [4] to distribute our training over the four available GPUs. We use the model parallel training [4] to split the last classification layer across the four GPUs for faster training.

References

[1] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[4] Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, et al. Pytorch distributed: Experiences on accelerating data parallel training. *arXiv preprint arXiv:2006.15704*, 2020.

[5] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[6] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[8] Mei Wang and Weihong Deng. Mitigating bias in face recognition using skewness-aware reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9322–9331, 2020.