UNDER REVIEW

RESCIENCE C

Replication / ML Reproducibility Challenge 2022

[Re] Reproducing FairCal: Fairness Calibration for Face Verification

Jip Greven[1,†], Simon Stallinga[1,†], and Zirk Seljee[1,†]

[1]Informatics Institute, University of Amsterdam, The Netherlands – [†]Equal contributions

Edited by
(Editor)

Received
–

Published
–

DOI
–

Reproducibility Summary

Scope of Reproducibility — This reproducibility paper verifies the claim by Salvador et al. in "*FairCal: Fairness Calibration for Face Verification*" [1] that the FairCal and Oracle methods are fair with respect to sensitive attributes and obtain SOTA accuracy results in face verification when compared to FSN and FTC. The aim is to reproduce the relative values in Tables 2, 3 and 4 of the original paper for these methods. We also provide and empirically support an intuitive explanation of why FairCal outperforms Oracle.

Methodology — The authors provided partial code to create the results; Code to create and preprocess embeddings was missing, but code to run the experiments on these embeddings was provided. Nevertheless, we re-implement the code from scratch, keeping the data structure identical. Hardware used are personal laptops without GPU and a desktop with an MSI GeForce GTX 1060-3GB GPU.

Results — Compared to the data reported in the original paper, the reproduced results vary across embedding models and evaluation metrics, where some combinations perform very similarly to the original paper while other combinations deviate significantly. Despite this, the claims of the original paper have been confirmed, which include no loss of accuracy, fairly calibrated subgroups and predictive equality.

What was easy — Some parts of the reproduction went smoothly such as the accessibility of the data and models and the quick execution of the experiments. Furthermore, the paper was clear about evaluation metrics. Finally, code for the figures worked straight out of the box.

What was difficult — The exact steps of the original implementation were unclear to us because the provided code had few comments and its structure was not immediately obvious. Additionally, obtaining and correctly running the ArcFace model from its ONNX file was not successful because we never worked with ONNX and initially downloaded a broken instance.

Communication with original authors — We had indirect contact with the first author who provided an example of the required metadata structure and clarified that all unmentioned hyperparameters were kept at their default values.

Copyright © 2023 J. Greven, S. Stallinga and Z. Seljee, released under a Creative Commons Attribution 4.0 International license.
Correspondence should be addressed to Zirk Seljee (zirk.seljee@student.uva.nl)
The authors have declared that no competing interests exist.
Code is available at https://github.com/zseljee/re-faircal. – SWH swh:1:dir:95f2895ab0761e8a2341dc83e26cdcbbc5a0ecde.
Open peer review is available at https://openreview.net/forum?id=jDBYRwDpeW.

1 Introduction

Fairness of intelligent systems is a hot topic and the methods developed with fairness guarantees should be thoroughly tested for reproducibility to verify their performance and generalizability. One instance in which fairness could be of life-impacting importance is face verification by police forces in arresting criminals, as a false arrest can be a traumatic experience. For recidivism estimation, face verification should also be unbiased to minority groups like Asians in the United States.
Salvador et al. introduce *FairCal*, a post-training approach that increases the fairness of face calibration models [1]. To support the improvement of this method, we aim to reproduce the findings of the authors. We illustrate the difference in fairness between different models in Figure 1. The next section describes what we aim to reproduce comprehensively.

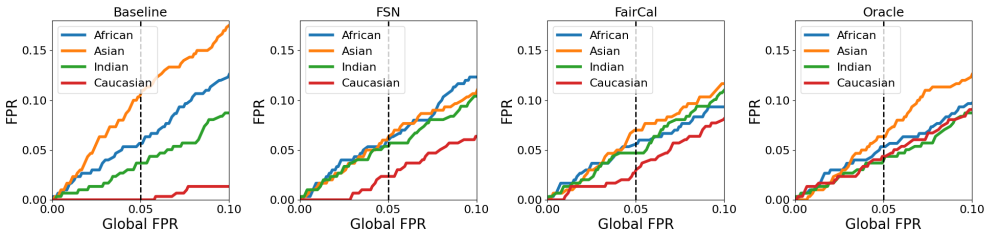

Figure 1. Illustration of how to inspect improved fairness measured by the FPRs evaluated on ethnicity pairs in the RFW dataset. Bias between subgroups is reduced with post-processing methods Fair Template Comparison (FTC) [2], Fair Score Normalization (FSN) [3], Faircal and Oracle. (Lines close together mean ethnicities have similar False Positive rates; they are treated similarly.)

2 Scope of reproducibility

Salvador et al. introduce two fairness calibration methods, FairCal and Oracle, and compare them to state-of-the-art fairness calibration models. The authors test their method with three face-embedding methods: Facenet (VGGFace2), Facenet (Webface) [4] and ArcFace [5]. They evaluate on two datasets BFW [6] and RFW [7, 8, 9, 10] according to three metrics: (i) accuracy using true positive rate (TPR) at fixed false positive rates (FPR), (ii) fairness calibration using the mean and deviation of the Kolmogorov-Smirnov calibration (KS) error [11], and (iii) deviation across subgroups (predictive equality) using FPR deviation across subgroups for fixed global FPR.
This paper aims to prove the following claims, adapted from the original claims by Salvador et al. [1]:

- FairCal obtains state-of-the-art accuracy compared to previous calibration methods on both datasets for all metrics and embeddings. Oracle achieves accuracy slightly lower than FairCal.
- FairCal and Oracle respectively obtain the second lowest and lowest KS mean and deviation.
- FairCal and Oracle obtain low deviation across subgroup FPR for a fixed global threshold for all datasets and models. When compared to Oracle, FairCal obtains a significantly lower deviation. Both methods can be outperformed slightly on 0.1% global FPR.

3 Methodology

To verify the claims that were made in Section 2, we start with the provided open-source implementation on the Github page of the authors [12].
The first step, image preprocessing and embedding, is not included in this repository and not described in detail in the paper, requiring a re-implementation. We start with a Python PyTorch implementation of MTCNN [4, 13] to detect and crop faces from the datasets. Each image is resized to square 400 pixel dimensions to accelerate the detection using batching. Pairs for which at least one of the faces does not get detected, are dropped from the datasets. The filtered images are then embedded using Facenet (trained on VGGFace2 and Webface) and ArcFace, which are described in detail below.
To start the experiments the code needs an undocumented csv file which we were able to recreate with help of the authors. With this, it becomes possible to run the experiments through the original code, however, we re-implement all code from scratch. This re-implementation enables us to significantly decrease the runtime for important functions, like clustering embeddings for FairCal, from a shared total of hours to minutes.

3.1 Embedding model descriptions

The paper uses 4 pre-trained models to create image embeddings. MTCNN for face cropping, and three methods for the embeddings from these cropped images: Facenet (VGGFace2), Facenet (Webface) and ArcFace.

MTCNN — A Multi Task Convolutional Neural Network (MTCNN) uses three convolutional networks to detect faces and facial landmarks in a cascaded fashion [13]. We use a Python implementation of the MTCNN that uses the PyTorch library, obtained from [4]. The MTCNN architecture and parameters for the networks should be identical to the original MATLAB implementation by Zhang et al. All hyperparameters are kept at their default values.

Facenet (VGGFace2) and Facenet (Webface) — The Facenet models are Convolutional Neural Networks (CNNs) that create 512 dimensional embeddings of input images containing aligned faces. These networks are based on an Inception v1 architecture [14] and can be retrieved from the same Pyorch library as the MTCNN [4]. The models were trained on the VGGFace2 dataset [15] and the Webface dataset [16] as indicated by the parenthesis.

ArcFace — The ArcFace model is another CNN that creates 512-dimensional embeddings of aligned faces. The specific ArcFace model used is based on a ResNet100 architecture [17] and can be retrieved from the ONNX model repository[1] [5]. The model has been trained on the MS-Celeb-1M dataset [18].
Due to implementation complications and computational constraints, we were unable to create and use appropriate embeddings with this model. See Table 2 in section 4 for the large discrepancy between our results and those obtained by Salvador et al.

3.2 Fairness calibrator descriptions

To prove the improved fairness of the new FairCal and Oracle methods they are compared to a baseline and two state-of-the-art approaches that need training: FTC and FSN. Since FSN performed best in the original paper and FTC requires long neural network training, we chose to exclude FTC in combination with Arcface in the comparisons.

[1]Specifically, through an amazon cloud storage link.

Baseline — The baseline method for face verification systems is to determine a threshold cosine similarity. To compare different methods on their fairness with KS calibration error, models should output probabilities. Thus, the similarities between faces are mapped to probabilities of a match using post-hoc beta calibration [19].

Fair Template Comparison (FTC) — The Fair Template Comparison (FTC) method uses a small Neural Network to estimate the similarity between embeddings [2]. The layer sizes of the original model have been scaled by a factor 4 to accommodate the 512-dimensional embeddings. Salvador et al. do not mention one of the layers, but do include this layer in their code. The FTC method uses a novel penalty score for individual fairness, which is used as a loss function when training the network. The output scores of this approach are not calibrated probabilities, and beta calibration is used to measure fairness-calibration. In this paper, we train the network for 50 epochs on the full training split with shuffling. From the validation results in section 4 we conclude this causes FTC to overfit.

Fair Score Normalization (FSN) — Terhörst et al. later proposed Fair Score Normalization (FSN) [3]. This method uses unsupervised clusters obtained by K-means clustering and a predefined global FPR to create group-specific shifts to the cosine similarities of image embeddings. Again, the output scores are mapped to probabilities using beta calibration.

FairCal — FairCal proposed by Salvador et al. is a similar post-training calibration method to FSN that instead does the beta calibration on each K-means cluster separately [1]. The calibrated scores are turned into output probabilities by taking the cluster-size-weighted average.

Oracle — Oracle is a supervised variant of FairCal that uses the sensitive attributes to create clusters [1]. In itself, it should represent a gold standard of fairness baselines. Oracle is not applicable in real situations where the sensitive attributes are not known or cannot be used for inference.

3.3 Dataset descriptions

The original paper uses two datasets for the main results. The Biased Faces in the Wild (BFW) [6] and the Racial Faces in the Wild (RFW) [7, 8, 9, 10] datasets. These datasets are designed to asses fairness of visual systems and were based on images from the larger VGGFace2 [15] and MS-Celeb-1M [18] datasets respectively. Models trained on one of the latter datasets are not evaluated on the respective former dataset. Both datasets consist of four different racial subsets: African, Asian, Caucasian and Indian.[2] The BFW dataset also contains splits based on gender annotations.
The preprocessing with MTCNN removes 234 and 77 faces from the BFW and RFW datasets respectively. This causes 21,495 and 94 image pairs to be respectively unusable.[3]
A summary of these statistics is provided in Table 1 on the next page.

3.4 Hyperparameters

Most of the hyperparameters were kept at their default values, or the authors clarified their adjustments appropriately.

[2] The BFW dataset uses the respective class names Black, Asian, White and Indian.

[3] A face may be identified by the MTCNN without being used if it is only paired with images where the MTCNN fails to identify the other face from the pair. The RFW dataset has 40,530 out of 40,607 identified faces, but 63 of these are not used due to their pairing.

Table 1. Overview of the used fairness evaluation datasets.

Dataset	Based on	Used imgs	Used pairs	Subgroups	Inter-group pairings	Folds	URL
BFW	VGGFace2	19,766	902,403	Race, gender	✓	5	link
RFW	MS-Celeb-1M	40,467	23,906	Race	✗	10	link

The most important hyperparameter is the number of clusters, K, in the K-means clustering algorithm. The original paper provides code for a manual search across different cluster sizes K and the results are shown in Figures 4 to 14 of [1]. The chosen value for the results in the paper and this reproducibility is $K = 100$.

3.5 Experimental setup and code

The results are obtained using 5-fold cross-validation. The BFW contains five folds and is split as expected: one fold is left out as validation data while the other four are used for training. The RFW contains ten folds and results are obtained using cross-validation on the first five folds. It is not entirely clear how the remaining (train and validation) folds are used. Either (i) the last five folds were always included in the training split, which provides more training data while not validating on all data, (ii) the last five folds were combined with the first five folds, which expands the folds and includes all data for training and validating, or (iii) the last folds were excluded from all experiments, which discards a significant part of the data. All three approaches were tested and compared, yet none gave any significant differences in validation results. For the results in section 4, approach (i) is used as it includes all data and validates on the same data as presented in the code of the original paper.
When running the K-means algorithm, the embeddings of each image occurring at least once in a fold were included for clustering. It is therefore possible that embeddings are clustered in multiple 'runs' of the K-means algorithm for different folds, just as images are reused across folds. Each 'run' of the K-means algorithm clusters the data ten times and returns the result with the minimal remaining inertia, as per the default parameters of the scikit-learn implementation [20].
Code for this paper is available on GitHub and the Software Heritage Archive.

3.6 Computational requirements

All methods in this paper can run in their entirety on a CPU, including preprocessing. The preprocessing can optionally be accelerated by utilising a GPU. Our timing measurements were obtained using an Intel i5 core CPU with 16GB RAM and a MSI GeForce GTX 1060-3GB GPU.
The face identification and cropping takes around 50 minutes with CPU on the RFW dataset. The smaller BFW dataset completes in approximately 20 minutes. A GPU can complete this in 10 minutes each.
Creating the embeddings takes approximately a minute per 10,000 images with a GPU. Creating the ArcFace embeddings takes approximately 4.5 hours for the BFW images, we only managed to run this on a CPU.
The experiments in their original state took multiple hours, but refactoring the code into only the strictly necessary components and caching temporary results significantly improved the run-time to a few minutes. The most intensive part of the experiments is creating the K-means clusters for FairCal, which takes 100 seconds per fold. Doing the FairCal calibration finishes near instantly and does not form a bottleneck.
Lastly, training the FTC models is computationally expensive and takes 12 seconds and 2 minutes per epoch for RFW and BFW respectively.

Table 2. Global accuracy measured by TPR at several FPR thresholds, comparing the original results (Sal.) with ours. (Higher is better.)

		Approach →	Baseline			FTC			FSN			FairCal			Oracle		
		By →	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.
Dataset	Feature	TPR @ ↓															
RFW	FaceNet (VGGFace2)	0.1% FPR	18.42	21.52	+3.10	6.86	21.52	+14.66	23.01	27.26	+4.25	23.55	28.34	+4.79	21.40	26.84	+5.44
		1.0% FPR	34.88	40.15	+5.27	23.66	40.15	+16.49	40.21	44.93	+4.72	41.88	48.90	+7.02	41.83	47.98	+6.15
	FaceNet (WebFace)	0.1% FPR	11.18	13.00	+1.82	4.65	13.00	+8.35	17.33	15.24	-2.09	20.64	21.05	+0.41	16.71	19.18	+2.47
		1.0% FPR	26.04	26.58	+0.54	18.40	26.58	+8.18	32.80	32.92	+0.12	33.13	35.38	+2.25	31.60	33.08	+1.48
BFW	FaceNet (WebFace)	0.1% FPR	33.61	27.59	-6.02	13.60	0.12	-13.48	47.11	35.02	-12.09	46.74	34.96	-11.78	45.13	32.83	-12.30
		1.0% FPR	58.87	51.61	-7.26	43.09	1.18	-41.91	68.92	57.57	-11.35	69.21	57.56	-11.65	67.56	55.78	-11.78
	ArcFace	0.1% FPR	86.27	17.36	-68.91	82.09	N/A	N/A	86.19	19.23	-66.96	86.28	19.58	-66.70	86.41	17.78	-68.63
		1.0% FPR	90.11	31.65	-58.46	88.24	N/A	N/A	90.06	32.70	-57.36	90.14	32.86	-57.28	90.40	33.25	-57.15

4 Results

The accuracy improvement of FairCal and Oracle on the RFW dataset are proportionally equal compared to the original results. There is less accuracy improvement on the BFW dataset relative to the original results.
We observe that the initial baseline is usually fairer than the original paper suggests and that FairCal and Oracle result in an equal fairness increase.

4.1 Results reproducing original paper

We separately confirm the results across the three claims described in section 2.

Accuracy — The first claim is that the TPR of FairCal is the best when compared to previous methods, with Oracle close behind. Our results in Table 2 highlight that this is the case independent of embeddings and global FPR. However, our results, including the baseline, also differ significantly from the results of Salvador et al. This is consistent across embeddings, which implies that the embeddings are significantly different. The results for ArcFace differ so significantly we infer our implementation to be wrong and refer to subsection 5.2 for possible explanations. Finally, the results of FTC on the BFW dataset are significantly different from the expected values. We suspect that this error comes from overfitting on the relatively small dataset in combination with many image pairs. This may also be the case for the RFW dataset, but it does not show in the results. We were not able to confirm the overfitting.

KS Fairness — The second claim is that FairCal and Oracle obtain the second lowest and lowest KS mean and deviation respectively. Our results in Table 3 confirm FairCal and Oracle are more fairly calibrated, as can be seen by the significantly lower KS mean and standard deviation compared to the baseline and mostly lower mean compared to FSN. Also, the bias within groups, as can be seen by the KS STD, is significantly lower for FairCal and Oracle, again confirming the original claims of the paper. However, it is important to note that in some aspects the reproduced results deviate significantly from the original.

Predictive equality — The third and last claim is that FairCal and Oracle both obtain low FPR deviation across subgroups, but that FairCal obtains significantly better inter-subgroup fairness compared to Oracle. Our results in Table 4 support the claim of low deviation, but do not convincingly show that FairCal is better than Oracle. The biggest difference in deviation is for 1.0% global FPR on the BFW dataset, where all measured deviations, independent of approach, are lower. The deviation for 1.0% global FPR on the RFW dataset with Facenet (VGGFace2) has doubled for FairCal; inspection shows that this is due to folds 1 and 4 being outliers with low Oracle and high FairCal deviation

Table 3. Fairness calibration measured by the mean KS across the sensitive subgroups. Showing the Mean and Standard Deviation (STD). Comparing original results (Sal.) with ours. (Lower is better in all cases.)

		Approach →	Baseline			FTC			FSN			FairCal			Oracle		
		By →	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.
Dataset	Feature	Metric ↓															
RFW	FaceNet (VGGFace2)	Mean	6.37	6.35	-0.02	5.69	6.76	+1.07	1.43	2.33	+0.90	1.37	1.57	+0.20	1.18	1.48	+0.30
		STD	3.77	3.24	-0.53	2.95	3.61	+0.66	0.40	0.92	+0.52	0.34	0.51	+0.17	0.33	0.45	+0.12
	FaceNet (WebFace)	Mean	5.55	5.45	-0.10	4.73	12.46	+7.73	2.49	2.23	-0.26	1.75	1.78	+0.03	1.35	1.82	+0.47
		STD	2.91	2.90	-0.01	2.28	4.82	+2.54	0.91	0.82	-0.09	0.45	0.60	+0.15	0.43	0.39	-0.04
BFW	FaceNet (WebFace)	Mean	6.77	4.11	-2.66	6.64	52.73	+46.09	2.76	2.95	+0.19	3.09	3.29	+0.20	2.23	1.82	-0.41
		STD	4.03	2.90	-1.13	3.27	3.08	-0.19	1.60	2.27	+0.67	1.55	1.55	+0.00	1.40	0.84	-0.56

Table 4. Predictive equality: For two choices of global FPR compare the deviations measured in subgroup FPRs in terms of Standard Deviation (STD). Comparing original results (Sal.) with ours. (Lower is better.)

		Approach →	Baseline			FTC			FSN			FairCal			Oracle		
		By →	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.
Dataset	Feature	STD @ ↓															
RFW	FaceNet (VGGFace2)	0.1% FPR	0.10	0.17	+0.07	0.11	0.17	+0.06	0.11	0.17	+0.06	0.10	0.18	+0.08	0.12	0.21	+0.09
		1.0% FPR	0.74	0.75	+0.01	0.66	0.75	+0.09	0.46	0.56	+0.10	0.32	0.60	+0.28	0.45	0.53	+0.08
	FaceNet (WebFace)	0.1% FPR	0.16	0.19	+0.03	0.14	0.19	+0.05	0.13	0.14	+0.01	0.10	0.18	+0.08	0.13	0.18	+0.05
		1.0% FPR	0.79	0.87	+0.08	0.66	0.87	+0.21	0.40	0.48	+0.08	0.35	0.35	-0.00	0.48	0.40	-0.08
BFW	FaceNet (WebFace)	0.1% FPR	0.40	0.24	-0.16	0.32	N/A	N/A	0.11	0.11	+0.00	0.11	0.13	+0.02	0.15	0.17	+0.02
		1.0% FPR	3.22	1.72	-1.50	2.57	N/A	N/A	1.05	0.72	-0.33	0.95	0.71	-0.24	0.91	0.78	-0.13

respectively. Excluding the outliers results in STD of 0.50 for FairCal and 0.61 for Oracle. This provides some evidence that supports the claim that FairCal is fairer than Oracle.

4.2 Results beyond original paper

To support the result in the previous section we investigate why FairCal could be outperforming Oracle. With access to sensitive attributes, Oracle would be expect to perform better. We hypothesised Faircal is fairer because it can calibrate for subgroups within the ethical groups, whereas Oracle cannot take this into account. By only using the sensitive attributes for clustering Oracle misses out on information contained in the embeddings.

FairCal cluster inspection — To verify our hypothesis we look into the distribution of ethnicities in the K-means clusters. If faces in ethnically diverse clusters have a common feature the hypothesis is supported. We use the cluster assignment for faces on the RFW dataset and plot how often each ethnicity occurs in Figure 2.
We inspect a diverse and a homogeneous cluster in Figure 3. The diverse cluster in Figure 3a has a common denominator of older-looking males and the homogeneous cluster in Figure 3b has a common denominator of Caucasian younger-looking blond-haired females. This illustrates that unsupervised clustering creates subgroups for attributes that are also sensitive like age and sex. Note that only 25 images are shown with a relative high number of outliers compared to the full clusters.

5 Discussion

The accuracy of our reproduced Faircal and Oracle is better than the baseline, FTC and FSN. This supports the claim that FairCal obtains significantly higher accuracy than the baseline and other models on both datasets for all metrics and models. Since Oracle is slightly lower than FairCal, the claim that Oracle is a close second is supported.

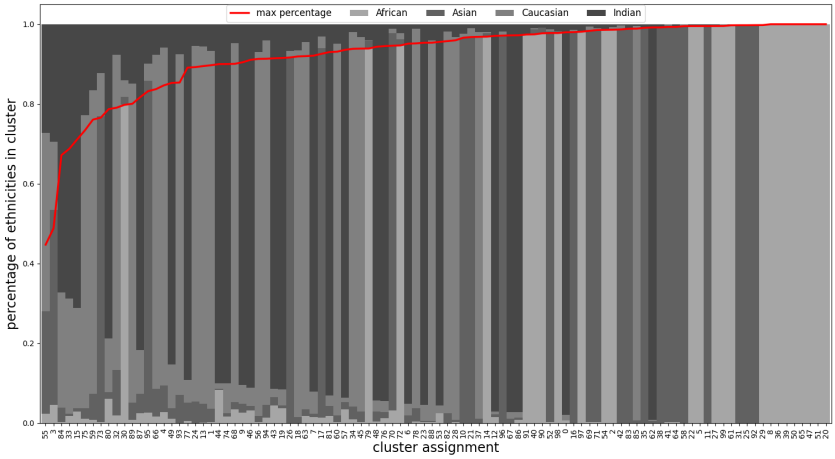

Figure 2. Sorted percentages of different ethnicities in the clusters on the RFW dataset. The clusters to the left are the most diverse clusters and the clusters get more homogeneous the more you move to the right. The red line displays the percentage of the largest ethnicity in the cluster

Although our results support the statement of accuracy, our implementation is significantly off in two cases compared to the original paper, RFW (VGGFace2) and BFW (WebFace). We thoroughly compared the different implementations of the proposed methods looking for a difference that would explain this. We could not locate errors in our code that could justify the differences. Since the differences are stable across embeddings and different between embeddings, we expect the difference to originate from these or a subset of these embeddings.
The second claim is that FairCal and Oracle obtain the lowest KS mean and deviation is supported by the metrics in Table 3. FairCal and Oracle respectively score the lowest and second lowest; they are the best in fairness calibration. The pattern from before, where RFW (WebFace) has the only similar results, is not present in Table 3, which highlights that the methods perform more fair regardless of differences in accuracy.
The third and final claim is that FairCal and Oracle obtain a low deviation across subgroups for all datasets and models and that FairCal is significantly lower than Oracle. When compared to the baseline both of the methods obtain lower deviation and as expected they are outperformed by FSN. While the claim that FairCal obtains lower deviation than Oracle is not convincingly supported by the reported measure, we provide and support a hypothesis that aligns with this claim.

5.1 What was easy

The data and the models were generally accessible, and the execution of the experiments was swift. This allowed for thorough debugging and testing of minor changes. Furthermore, the original paper was explicit about the way the evaluation metrics were used and these were easy to implement. The code for the creation of the tables and figures like the original paper was provided on GitHub and worked straight out of the box after the appropriate information had been provided.

5.2 What was difficult

The exact steps of the original implementation were unclear to us because the provided code had few comments and its structure was not immediately obvious.

(a) Subset of cluster 55, this is the most diverse cluster of all clusters. The cluster mostly consists of older-looking males.

(b) Subset of cluster 85, the most homogeneous Caucasian cluster. This cluster consists of Caucasian females with blonde hair.

Figure 3. Clusters generated on the RFW dataset that does not contain gender-annotations.

Additionally, one of the most challenging parts was when we discovered that the initially used Arcface model on GitHub was incorrect. After some investigation, we discovered that the correct model was provided as an *onnx* file. As authors who had never worked with this specification before, we were required to test multiple options before running the model. Based on our results, we still suspect that we did not manage to implement it correctly.

5.3 Communication with original authors

Indirectly, via fellow students reproducing this paper, we had e-mail contact with the first author, who responded fast and provided two example files for the required metadata structure and clarified that all unmentioned hyperparameters were kept at their default values.

References

1. T. Salvador, S. Cairns, V. Voleti, N. Marshall, and A. M. Oberman. "FairCal: Fairness Calibration for Face Verification." In: **International Conference on Learning Representations**. 2022. URL: https://openreview.net/forum?id=nRj0NcmSuxb.
2. P. Terhörst, M. L. Tran, N. Damer, F. Kirchbuchner, and A. Kuijper. "Comparison-level mitigation of ethnic bias in face recognition." In: **2020 8th international workshop on biometrics and forensics (iwbf)**. IEEE. 2020, pp. 1–6.
3. P. Terhörst, J. N. Kolf, N. Damer, F. Kirchbuchner, and A. Kuijper. "Post-comparison mitigation of demographic bias in face recognition using fair score normalization." In: **Pattern Recognition Letters** 140 (2020), pp. 332–338.
4. T. Esler and Contributors. **Face recognition using pytorch**. Dec. 2021. URL: https://github.com/timesler/facenet-pytorch.
5. O. N. N. E. (ONNX). **ONNX Model Zoo**. Dec. 2022. URL: https://github.com/onnx/models.
6. J. P. Robinson, G. Livitz, Y. Henon, C. Qin, Y. Fu, and S. Timoner. "Face Recognition: Too Bias, or Not Too Bias?" In: **CoRR** abs/2002.06483 (2020). arXiv:2002.06483. URL: https://arxiv.org/abs/2002.06483.
7. M. Wang, W. Deng, J. Hu, X. Tao, and Y. Huang. "Racial Faces in the Wild: Reducing Racial Bias by Information Maximization Adaptation Network." In: **The IEEE International Conference on Computer Vision (ICCV)**. Oct. 2019.

8. M. Wang, Y. Zhang, and W. Deng. "Meta Balanced Network for Fair Face Recognition." In: **IEEE Transactions on Pattern Analysis and Machine Intelligence** (2021).
9. M. Wang and W. Deng. "Mitigate Bias in Face Recognition using Skewness-Aware Reinforcement Learning." In: **arXiv preprint arXiv:1911.10692** (2019).
10. M. Wang and W. Deng. "Deep Face Recognition: A survey." In: **Neurocomputing** 429 (2021), pp. 215–244.
11. K. Gupta, A. Rahimi, T. Ajanthan, T. Mensink, C. Sminchisescu, and R. Hartley. "Calibration of Neural Networks using Splines." In: **International Conference on Learning Representations**. 2021. URL: https://openreview.net/forum?id=eQe8DEWNN2W.
12. T. Salvador. **Code for the paper FairCal: Fair calibration for face verification.** Nov. 2021. URL: https://github.com/tiagosalvador/faircal.
13. K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. "Joint face detection and alignment using multitask cascaded convolutional networks." In: **IEEE signal processing letters** 23.10 (2016), pp. 1499–1503.
14. 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 (CVPR)**. June 2015.
15. Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. "VGGFace2: A Dataset for Recognising Faces across Pose and Age." In: **2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)**. 2018, pp. 67–74. DOI: 10.1109/FG.2018.00020.
16. D. Yi, Z. Lei, S. Liao, and S. Z. Li. "Learning Face Representation from Scratch." In: **CoRR** abs/1411.7923 (2014). arXiv:1411.7923. URL: http://arxiv.org/abs/1411.7923.
17. K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition." In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. June 2016.
18. Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. "MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition." In: **Computer Vision – ECCV 2016**. Ed. by B. Leibe, J. Matas, N. Sebe, and M. Welling. Cham: Springer International Publishing, 2016, pp. 87–102. URL: https://doi.org/10.1007/978-3-319-46487-9_6.
19. K. Patel, W. H. Beluch, B. Yang, M. Pfeiffer, and D. Zhang. "Multi-Class Uncertainty Calibration via Mutual Information Maximization-based Binning." In: **International Conference on Learning Representations**. 2021. URL: https://openreview.net/forum?id=AICNpd8ke-m.
20. F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: **Journal of Machine Learning Research** 12 (2011), pp. 2825–2830.

6 Full comparison Tables

This appendix section is dedicated to Tables 5 and 6 that compare all metrics provided in the original paper. Because the results agree across all metrics, we deemed standard deviation the best metric to highlight the differences and similarities in variation.

Table 5. Fairness calibration measured by the mean KS across the sensitive subgroups. Showing the Mean, Average Absolute Deviation (AAD), Maximum Absolute Deviation (MAD) and Standard Deviation (STD). Comparing original results (Sal.) with ours. (Lower is better in all cases.)

		Approach →	Baseline			FTC			FSN			FairCal			Oracle		
		By →	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.
Dataset	Feature	Metric ↓															
RFW	FaceNet (VGGFace2)	Mean	6.37	6.35	-0.02	5.69	6.76	+1.07	1.43	2.33	+0.90	1.37	1.57	+0.20	1.18	1.48	+0.30
		AAD	2.89	2.55	-0.34	2.32	3.01	+0.69	0.35	0.81	+0.46	0.28	0.43	+0.15	0.28	0.38	+0.10
		MAD	5.73	4.95	-0.78	4.51	6.01	+1.50	0.57	1.37	+0.80	0.50	0.77	+0.27	0.53	0.66	+0.13
		STD	3.77	3.24	-0.53	2.95	3.61	+0.66	0.40	0.92	+0.52	0.34	0.51	+0.17	0.33	0.45	+0.12
	FaceNet (WebFace)	Mean	5.55	5.45	-0.10	4.73	12.46	+7.73	2.49	2.23	-0.26	1.75	1.78	+0.03	1.35	1.82	+0.47
		AAD	2.48	2.32	-0.16	1.93	4.11	+2.18	0.84	0.71	-0.13	0.41	0.52	+0.11	0.38	0.33	-0.05
		MAD	4.97	4.64	-0.33	3.86	8.21	+4.35	1.19	1.20	+0.01	0.64	0.97	+0.33	0.66	0.55	-0.11
		STD	2.91	2.90	-0.01	2.28	4.82	+2.54	0.91	0.82	-0.09	0.45	0.60	+0.15	0.43	0.39	-0.04
BFW	FaceNet (WebFace)	Mean	6.77	4.11	-2.66	6.64	52.73	+46.09	2.76	2.95	+0.19	3.09	3.29	+0.20	2.23	1.82	-0.41
		AAD	3.63	2.36	-1.27	2.80	2.54	-0.26	1.38	1.88	+0.50	1.34	1.38	+0.04	1.15	0.68	-0.47
		MAD	5.96	6.58	+0.62	5.61	5.57	-0.04	2.67	5.19	+2.52	2.48	2.74	+0.26	2.63	1.83	-0.80
		STD	4.03	2.90	-1.13	3.27	3.08	-0.19	1.60	2.27	+0.67	1.55	1.55	+0.00	1.40	0.84	-0.56
	ArcFace	Mean	2.57	10.55	+7.98	2.95	N/A	N/A	2.65	10.82	+8.17	2.49	10.33	+7.84	1.41	2.61	+1.20
		AAD	1.39	2.75	+1.36	1.48	N/A	N/A	1.45	2.78	+1.33	1.30	2.51	+1.21	0.59	1.34	+0.75
		MAD	2.94	6.86	+3.92	3.03	N/A	N/A	3.23	6.78	+3.55	2.68	5.89	+3.21	1.30	3.73	+2.43
		STD	1.63	3.36	+1.73	1.74	N/A	N/A	1.71	3.42	+1.71	1.52	3.11	+1.59	0.69	1.62	+0.93

Table 6. Predictive equality: For two choices of global FPR compare the deviations in subgroup FPRs in terms of Average Absolute Deviation (AAD), Maximum Absolute Deviation (MAD), and Standard Deviation (STD). Comparing original results (Sal.) with ours. (Lower is better in all cases.)

			Approach →	Baseline			FTC			FSN			FairCal			Oracle		
			By →	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.	Sal.	Our	diff.
Thr.	Dataset	Feature	Metric ↓															
0.1% FPR	RFW	FaceNet (VGGFace2)	AAD	0.10	0.15	+0.05	0.10	0.15	+0.05	0.10	0.16	+0.06	0.09	0.16	+0.07	0.11	0.18	+0.07
			MAD	0.15	0.29	+0.14	0.15	0.29	+0.14	0.18	0.25	+0.07	0.14	0.27	+0.13	0.19	0.34	+0.15
			STD	0.10	0.17	+0.07	0.11	0.17	+0.06	0.11	0.17	+0.06	0.10	0.18	+0.08	0.12	0.21	+0.09
		FaceNet (WebFace)	AAD	0.14	0.17	+0.03	0.12	0.17	+0.05	0.11	0.13	+0.02	0.09	0.17	+0.08	0.11	0.17	+0.06
			MAD	0.26	0.28	+0.02	0.23	0.28	+0.05	0.23	0.21	-0.02	0.16	0.26	+0.10	0.20	0.27	+0.07
			STD	0.16	0.19	+0.03	0.14	0.19	+0.05	0.13	0.14	+0.01	0.10	0.18	+0.08	0.13	0.18	+0.05
	BFW	FaceNet (WebFace)	AAD	0.29	0.17	-0.12	0.24	0.00	-0.24	0.09	0.08	-0.01	0.09	0.09	+0.00	0.12	0.13	+0.01
			MAD	1.00	0.63	-0.37	0.74	0.00	-0.74	0.20	0.31	+0.11	0.20	0.37	+0.17	0.25	0.45	+0.20
			STD	0.40	0.24	-0.16	0.32	0.00	-0.32	0.11	0.11	+0.00	0.11	0.13	+0.02	0.15	0.17	+0.02
		ArcFace	AAD	0.12	0.07	-0.05	0.09	N/A	N/A	0.11	0.06	-0.05	0.11	0.08	-0.03	0.12	0.09	-0.03
			MAD	0.30	0.25	-0.05	0.20	N/A	N/A	0.28	0.16	-0.12	0.31	0.25	-0.06	0.27	0.32	+0.05
			STD	0.15	0.09	-0.06	0.11	N/A	N/A	0.14	0.07	-0.07	0.15	0.10	-0.05	0.14	0.12	-0.02
1.0% FPR	RFW	FaceNet (VGGFace2)	AAD	0.68	0.67	-0.01	0.60	0.67	+0.07	0.37	0.48	+0.11	0.28	0.54	+0.26	0.40	0.46	+0.06
			MAD	1.02	0.94	-0.08	0.91	0.94	+0.03	0.68	0.82	+0.14	0.46	0.87	+0.41	0.69	0.80	+0.11
			STD	0.74	0.75	+0.01	0.66	0.75	+0.09	0.46	0.56	+0.10	0.32	0.60	+0.28	0.45	0.53	+0.08
		FaceNet (WebFace)	AAD	0.67	0.74	+0.07	0.54	0.74	+0.20	0.35	0.42	+0.07	0.29	0.30	+0.01	0.41	0.36	-0.05
			MAD	1.23	1.27	+0.04	1.05	1.27	+0.22	0.61	0.74	+0.13	0.57	0.47	-0.10	0.74	0.53	-0.21
			STD	0.79	0.87	+0.08	0.66	0.87	+0.21	0.40	0.48	+0.08	0.35	0.35	-0.00	0.48	0.40	-0.08
	BFW	FaceNet (WebFace)	AAD	2.42	1.32	-1.10	1.94	0.00	-1.94	0.87	0.57	-0.30	0.80	0.57	-0.23	0.77	0.62	-0.15
			MAD	7.48	4.08	-3.40	5.74	0.00	-5.74	2.19	1.87	-0.32	1.79	1.78	-0.01	1.71	1.84	+0.13
			STD	3.22	1.72	-1.50	2.57	0.00	-2.57	1.05	0.72	-0.33	0.95	0.71	-0.24	0.91	0.78	-0.13
		ArcFace	AAD	0.72	0.45	-0.27	0.54	N/A	N/A	0.55	0.31	-0.24	0.63	0.40	-0.23	0.83	0.44	-0.39
			MAD	1.51	1.30	-0.21	1.04	N/A	N/A	1.27	0.91	-0.36	1.46	1.30	-0.16	2.08	1.59	-0.49
			STD	0.85	0.58	-0.27	0.61	N/A	N/A	0.68	0.41	-0.27	0.78	0.52	-0.26	1.07	0.59	-0.48