Facial Recognition Adversarial Methods and Defenses

Vansh Gandhi
Georgia Institute of Technology
vgandhi8@gatech.edu

Farhad Sedaghati
Georgia Institute of Technology
farhad.sedaghati@gatech.edu

Mohammad Minhaz
Georgia Institute of Technology
mminhaz3@gatech.edu

Abstract

Facial recognition technology has become increasingly important in society, as it is used for a wide range of applications such as security and identity verification. The accuracy of facial recognition models is crucial, as false positives or negatives can have serious consequences, such as denying individuals access to services or falsely accusing them of crimes. In order to ensure that facial recognition systems are accurate and fair, it is important to exploit and patch their weaknesses. We explore this specifically by studying current state of the art models on a similarity task and identify their weaknesses. We then attempt to improve their accuracy by generating adversarial data and retraining the model. We find that this technique is helpful at improving the model for the specific circumstances created by the generated images. However, this comes at the cost of a much larger dataset, a large amount of dataset preprocessing, and ultimately a model which is not robust to further variations that may be made. We find that the technique is helpful albeit brittle.

1. Introduction/Background/Motivation

Our goal was to improve the accuracy of existing facial recognition models. We start by comparing the performance of three state-of-the-art models - ResNet50, DeepFace, and FaceNet - on a task that involves identifying whether two images are of the same person. This task is referred to as Face Similarity. We then use the results to identify situations where the model produces the false positives or false negatives. Finally, to try and fix the errors, we created new data that is designed to confuse the model, and use this data to retrain the model. By doing this, we aim to improve the accuracy of the model.

Today, facial recognition datasets are typically trained using real-world images. However, this approach has some limitations, as it does not account for all possible scenarios. This can leave holes in the model that can be exploited, such as by using fooling images. Additionally, current practice does not account for other potential challenges, such as changes in lighting or angles, which can affect the accuracy of the model. As a result, current face recognition systems may not be able to accurately identify individuals in all situations.

Improvements in the accuracy of facial recognition systems would have a number of important implications. For one, it would make these systems more reliable and effective for use in security and identity verification applications. It would also help to reduce the risk of false identifications, which can have serious consequences for individuals. Additionally, improving the accuracy of face recognition systems would help to increase their trustworthiness and acceptance by the general public, which is crucial for their widespread adoption and use. Finally, it would allow facial recognition to be used for a greater amount of use cases, such as automatically verifying identities. However, it can also lead to some more nefarious uses, such as allowing corporations to track individuals more easily.

1.1. Datasets

In order to evaluate face similarity, we used two base datasets, the LFW and VGGFace2 datasets. These datasets were further processed adversarially and also augmented to create new inputs.

1.1.1 VGGFace2

VGGFace2 includes 3.31 million images from 9131 celebrities (actors, athletes, politicians) (8631 identities for training, 500 identities for testing) with an average of 362 images per person [1]. These images were downloaded from Google Image Search having large variations in pose, age, illumination, ethnicity, and profession. This dataset has

developed by researchers at the Visual Geometry Group (VGG) at the University of Oxford.

1.1.2 LFW

The LFW dataset [3] includes thousands of subjects with tens of thousands of total images. The full dataset was trimmed to only include subjects with at least 2 or more images. This is to ensure we can run evaluations of the face similarity task, which requires 2 input images. After this pre-processing, we are left with 1,690 subjects, with some subjects having more than 500 faces.

1.1.3 Adversarial Data - FGSM

One of routine attack is the Fast Gradient Sign Method (FGSM) in which the sign of the gradient of the neural network loss with respect to the input image is multiplied by a small constant epsilon to produce noise and perturb the input image [4]. It has been shown that even with a small value of epsilon which results in indistinguishable noise to the image causes the model to incorrectly classify it (similar to the human eye but fools the model), as we will demonstrate later.

1.1.4 Augmented Data - Image Manipulation

Another dataset used was the LFW dataset, but with various image manipulations applied. Manipulations include transformations such as rotations, zooming in/out, and flipping the image.

2. Approach

We aimed to improve the accuracy of facial recognition models by identifying and addressing scenarios where these models produce incorrect results. To do this, we first evaluated the performance of three state-of-the-art models - ResNet50, DeepFace, and FaceNet - on the face similarity task, which involves determining whether two images are of the same person. We used the results of this evaluation to identify specific scenarios where the models produce the wrong answer.

Next, we used this information to create adversarial examples, which are images that are specifically designed to fool the models. These examples These adversarial examples were then used to retrain the models, by adding them to the training dataset and continuing to train and fine tune the models with this additional data. This approach was hypothesized to be successful because it allows us to identify and address specific weaknesses in the models, which in turn should improve the overall accuracy.

Finally, we enhanced the dataset with data augmentation, and retrained the models again.

One novel aspect of this approach is the combined use of adversarial and augmented examples to retrain the models. This allowed us to create specific and targeted data focused on improving the models' performance in specific scenarios, rather than relying on general real-world data. By including both forms of dataset enhancement, we hope to create a more robust model.

2.1. Problems

We anticipated a number of potential problems. Some of the problems that were top of mind for us when working with facial recognition technology included the following.

2.1.1 Data bias

One of the biggest challenges with facial recognition systems is that they can be biased towards certain groups of individuals. This can happen if the training data is not representative of the broader population, or if the algorithms are designed in a way that favors certain groups. As a result, it is important to carefully curate the training data to ensure that it is diverse and representative, and to carefully evaluate the algorithms to ensure that they do not exhibit bias.

2.1.2 Variability

Another challenge with facial recognition systems is that they must be able to handle a wide range of scenarios and conditions. This includes different lighting conditions, angles, and facial expressions, which can all affect the accuracy of the model. To address this challenge, it is important to include a variety of images in the training data, and to carefully evaluate the model's performance in different scenarios.

2.1.3 Accuracy

Another potential problem is that facial recognition systems can sometimes produce incorrect results, either by failing to identify individuals or by incorrectly identifying them. This can have serious consequences, such as denying individuals access to services or falsely accusing them of crimes. To address this problem, it is important to carefully evaluate the model's performance and to identify and address specific scenarios where the model produces the wrong answer.

We aimed to address the variability and accuracy aspect of these problems with one solution. We did not aim to improve the diversity of the training set directly.

2.2. Models

3 models were evaluated: ResNet50, DeepFace, and FaceNet.

2.2.1 ResNet

There are different types of ResNet networks (e.g. ResNet18, ResNet34, ResNet50, ResNet101, ResNet152, etc), with similar structures but different number of layers.

Generally the expectation is to see better performance by increasing the number of layers; however in practice, we observe that the accuracy of the model degrade (decrease suddenly) or saturate (which is not related to overfitting). This happens due to the effects of vanishing gradient. During the backpropagation process, the gradients are sent back to hidden layers and the weights are updated and this process continues until the input layer is reached. Vanishing gradient results in having smaller and smaller gradients at the very beginning layers, and therefore, the weights change slightly or not change at all. Thus, the network doesn't learn effectively and because of that the deeper networks may not converge or result in higher accuracy. Deep residual networks are pretty similar to CNNs with addition of identity connection (residual block or skip connection) between layers. By using the identity connection, all the layers in the network generate the best optimal feature maps [6, 5]. The advantage of adding this type of skip connection is that if any layer hurt the performance of the architecture, then it will be skipped by regularization.

In this project, we will use ResNet50 which 50 layers deep.

2.2.2 DeepFace

DeepFace is a facial recognition model designed by Facebook [7]. It is an eight layer model and achieves better than human accuracy when trained on a large dataset. What makes this model unique is that it does not need face alignment and this task is performed by the network itself. This is valuable because it removes the need for extravagant preprocessing of image data. We used a pretrained model with weights based on training on VGG-Face2 dataset.

2.2.3 FaceNet

FaceNet is a facial recognition model that generates embeddings. It is trained on the VGG-Face2 dataset using the triplet method. That is, it tries to reduce the distance between similar images and increase the distance between dissimilar images. [2]

We used a Pytorch implementation of FaceNet (https://github.com/timesler/facenet-pytorch) for our experiment. We also used MTCNN that comes with it to frame the facial images in a rectangle, so that facial features are more centered to their right places. This is because we are trying to analyze FaceNet's ability to find similarity and dissimilarity, and not the framing aspect of it. FaceNet uses InceptionResNetv1 which has among other blocks 153 Conv2d blocks. We also tried to create a simplified version of this using only 3 Conv2d blocks and trained it using distillation mechanism on LFW image set.

2.3. Code Repositories Leveraged

To help us in our endeavours, we used the PyTorch (pytorch.org) and Keras (https://keras.io) libraries. Additionally, we also made use of the Python Deepface library (https://github.com/serengil/deepface), which provides helpful wrappers around existing state of the art face recognition models and APIs for performing face similarity and verification. Augmentor (https://github.com/mdbloice/Augmentor) was used to aid in performing image data augmentation and modification. For the FaceNet implementation, a Pytorch implementation was used (https://github.com/timesler/facenet-pytorch).

3. Experiments and Results

3.1. Evaluation of Models

Our experimentation began with evaluating how current state of the art models performed on the face similarity task. 3 models were evaluated, ResNet50, DeepFace, and FaceNet. The models were evaluated against labeled images within the LFW dataset as well as the VGGFace2 dataset.

After the initial evaluation of the face similarity task, the networks were not performing as expected. After further investigation, we realized that normalization of the data was necessary. So, the LFW input dataset was normalized such that all input images were 250 pixels by 250 pixels and such that all the faces were aligned to the same pose within the frame. Additionally, a MTCNN was used in order to frame the images correctly.

Face similarity was ran among all the images in the datasets. The results are presented in Table 3. The image similarity metric was calculated using a 2 step process. First, each face was mapped to an embedding using the selected model. Then, the distance between two faces was calculated by taking the cosine similarity or euclidean distance between both of the embedding vectors. Finally, a match was constituted if the metric was above a certain threshold. This threshold was different for each model. 0.23 for DeepFace, and 0.4 for ResNet50 and FaceNet.

3.1.1 FaceNet

We found that mean distance between similar images is 0.47 with variance of 0.03. We also found that mean distance between dissimilar images is 1.38 with variance of 0.01. So clearly FaceNet proved to be very accurate in identifying similar and dissimilar images. However, FaceNet takes a long time to run for all images in LFW. So, we tuned the

network to create a distilled version of FaceNet. The distilled version had mean distance of 0.85 with variance of 0.04 for similar images and mean distance of 1.39 with variance of 0.005 for dissimilar images. Although the similar images had higher distance, it was still discernible from the dissimilar images. More importantly, the simpler, distilled version was running five times faster than the undistilled FaceNet model. The simpler version had the following structure: Three layers of CNN, followed by one layer of linear. Although, the similarity score was quite high and in some cases came close to dissimilarity score, out of 1680 individuals we only had 32 similarity miss - that is similar images was deemed as dissimilar considering Euclidean distance of 1.3 or above as being dissimilar. This is amazing considering the fact that simplified model ran almost 50 times faster than original.

Simplified Model The simplified model we designed is presented below.

InceptionResnetSimple(
(conv1): BasicConv2d(
(conv): Conv2d(3, 32, kernel_size=(8, 8), stride=(4, 4), bias=False)
(bn): BatchNorm2d(32, eps=0.001, momentum=0.1, affine=True, track_running_stats=True)
(relu): ReLU()
)
(conv2): BasicConv2d(
(conv): Conv2d(32, 64, kernel_size=(6, 6), stride=(4, 4), bias=False)
(bn): BatchNorm2d(64, eps=0.001, momentum=0.1, affine=True, track_running_stats=True)
(relu): ReLU()
)
(conv3): BasicConv2d(
(conv): Conv2d(64, 128, kernel_size=(8, 8), stride=(1, 1), bias=False)
(bn): BatchNorm2d(128, eps=0.001, momentum=0.1, affine=True, track_running_stats=True)
(relu): ReLU()
)
(lin): Linear(in_features=512, out_features=512, bias=False)
(last_bn): BatchNorm1d(512, eps=0.001, momentum=0.1, affine=True, track_running_stats=True)
)

After this, we trained the model on the augmented data, specifically focusing on the examples in which the model had elevated false positive and false negative rates. We then re-ran the experiment. For adversarial dataset enhancement, using the FSGM method, we obtain the baseline error rates

Figure 1. Adversarial example using FSGM with epsilon of 0.02

indicated in Table 2. Using the image generator function, we also perform data augmentation and add different rotation and flipping case to the dataset. The final dataset includes 1200 images, 200 for each subject.

Based on our analysis, the similarity test passes for each individual even introducing perturbation for adversarial examples. This indicates that the feature extractor part (the residual CNN part) works correctly; however, the classification part after feature extraction fails to classify the correctly. This means that we can use the transfer learning, freeze the first part of the network up to classifier and only re-train the fully connected layers (classifier part) to update the weights of neurons to include the effects of adversarial images.

After running several additional iterations of training on the adversarial and augmented dataset, the models were re-evaluated. While there was no regression in the existing test sets, the newly generated test set was unable to achieve much better results.

There are several reasons why adding adversarial examples to the training set might not lead to an improvement in the model's performance against new adversarial examples.

First, it's possible that the adversarial examples that were used in the experiment were not representative of the full range of possible adversarial examples that the model could encounter. In this case, even if the model is able to correctly classify the examples in the training set, it might still be vulnerable to other, unseen adversarial examples.

Second, it's also possible that the model's architecture or training procedure was not well-suited to learning from adversarial examples. In this case, even if the adversarial examples were added to the training set, the model might not be able to learn from them effectively.

Finally, it's also possible that the adversarial examples were not added to the training set in a way that was effective at improving the model's performance. For example, if the adversarial examples were added to the training set in small numbers, or if they were not balanced with other types of examples, this could limit their effectiveness at improving the model's performance.

Ultimately, because there is always a gradient for a given image, there is always a way to adjust it towards some specified target. And unless every possible input is covered, the approach of adding arbitrary new examples was not a fully

Model	Similarity Metric	Threshold	Accuracy
ResNet50	Cosine	0.40	TODO
DeepFace	Cosine	0.23	68%
FaceNet	Euclidean	0.47	TODO

Table 1. Evaluation of facial recognition models on the face similarity task. No fine tuning

Epsilon	Error Rate
0.01	68%
0.015	89%
0.02	100%

Table 2. Error Rates due to Adversarial Examples

adequate approach.

4. Conclusion

We find that existing state of the art models have some weaknesses. We attempted to address these weaknesses through the lens of face similarity by augmenting the dataset and running additinoal training. Through this process, we were able to improve performance of the model. We find that FSGM is a fast and simple method to create adversarial examples and by using a small value of epsilon we can create adversarial images which are indistinguishable from the original to the human eye. We found that the feature extractor part of the ResNet50 model (CNN part up to the Fully Connected Layers) works very well on creating embeddings. The previously failing adversarial inputs were subsequently classified correctly after training. Additionally, we created a simpler FaceNet model that maintains a similar level of accuracy but can run an order of magnitude faster. And finally, our analysis revelaed that simply adding adversarial examples as part of the training set and fine tuning is not a fully adequate measure to combat adversarial attacks. In the future, we hope to explore other methods for mitigating adversarial attacks by revisiting training procedures and architectures. It is also worth considering adding specific preprocessing steps to reduce the likelihood of these adversarial examples. Ultimately, this is an important issue to be solved, as these networks get used for more and more tasks, their robustness will be of the utmost importance.

5. Work Division

See Table 3.

References

[1] Shen L. Xie W. Parkhi O.M. Cao, Q. and A. Zisserman. Vggface2: A dataset for recognising faces across pose and age, 2018. 1

[2] James Philbin Florian Schroff, Dmitry Kalenichenko. Facenet: A unified embedding for face recognition and clustering. 3

[3] Tamara Berg Gary B. Huang, Manu Ramesh and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. 2

[4] Shlens J. Szegedy C. Goodfellow, I. J. Explaining and harnessing adversarial examples, 2014. 2

[5] Xiangyu Zhang Shaoqing Ren He, Kaiming and Jian Sun. Identity mappings in deep residual networks. In European conference on computer vision. 3

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

[7] Marc'Aurelio Ranzato Lior Wolf Yaniv Taigman, Ming Yang. Deepface: Closing the gap to human-level performance in face verification. 3

Student Name	Contributed Aspects	Details
Vansh Gandhi	Implementation & Analysis. Final paper writing+assembly	DeepFace implementation, evaluation, and analysis. Dataset curation. Dataset augmentation. Assembly of all analysis and final paper writing+editing
Mohammad	Implementation & Analysis.	FaceNet implementation, evaluation, and analysis. Distilled FaceNet implementation
Farhad	Implementation & Analysis.	ResNet50 implementation, evaluation, and analysis. Adversarial dataset creation. Data augmentation

Table 3. Contributions of team members.