Generating Synthetic Disguised Faces with Cycle-Consistency Loss and Automated Filtering Algorithm

Mobeen Ahmad, Usman Cheema, Muhammad Abdullah, Seungbin Moon, and Dongil Han

DOI: https://doi.org/10.3390/math10010004
Journal Impact Factor: 2.258
Volume: 10
Issue: 1
Rank by Journal Impact Factor: 24/330 (Q1) Top 7.2%
Rank by Journal Citation Indicator: 18/471 (Q1) Top 3.8%

VIP Lab Sejong University

Introduction

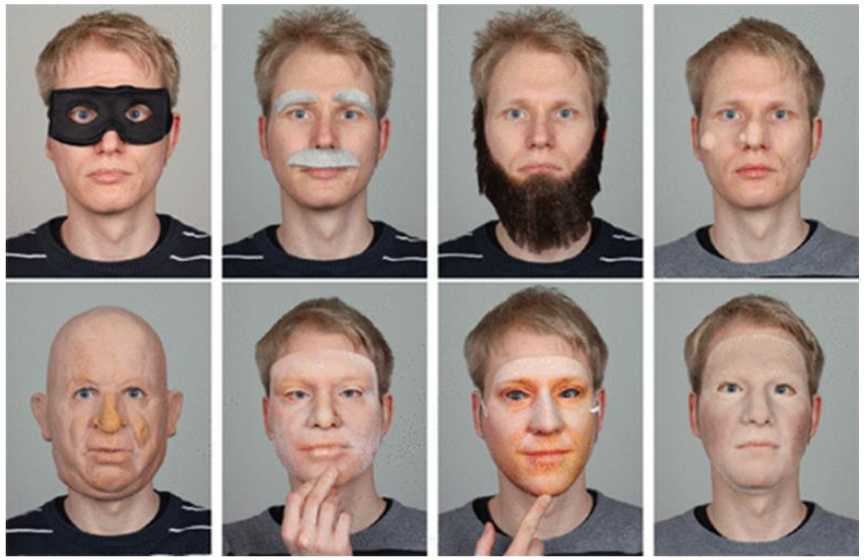
- Face recognition systems are prone to
  - Presentation-attacks
  - Spoofing
  - **Disguise**
- Face recognition is being used in many public domains for security
- It is seen that typical face recognition system is not capable of correctly recognizing disguised faces.

Training data	Testing data	Accuracy (%)
Non-disguised	Non-disguised	99.6
	Disguised	26

Existing Disguised Face Databases

- I^2BVSD
- BRSU Spoof Database
- Spectral Disguise Face Database
- CASIA SURF Database

(a) I^2BVSD



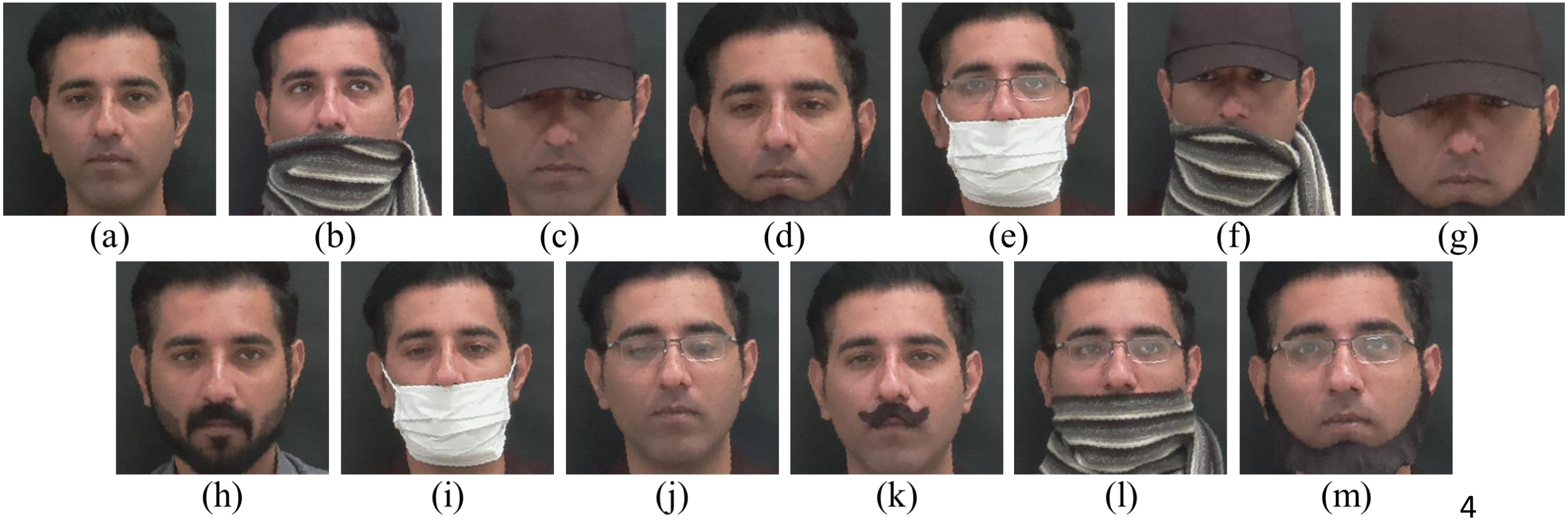
(b) BRSU Spoof Database

(c) Spectral Disguise Face Database

(d) CASIA SURF Database

Existing Disguised Face Databases

- 13 disguise add-ons
- Combination add-ons
- Two subsets: Subset-A, and Subset-B

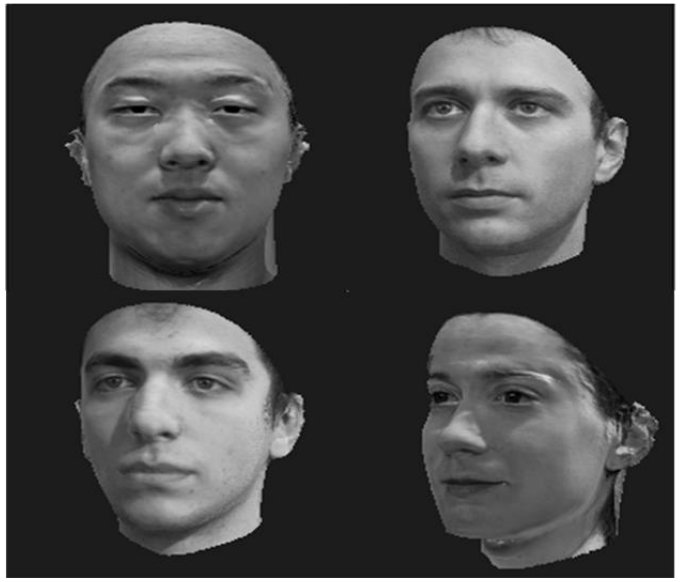


Synthetic Face Databases

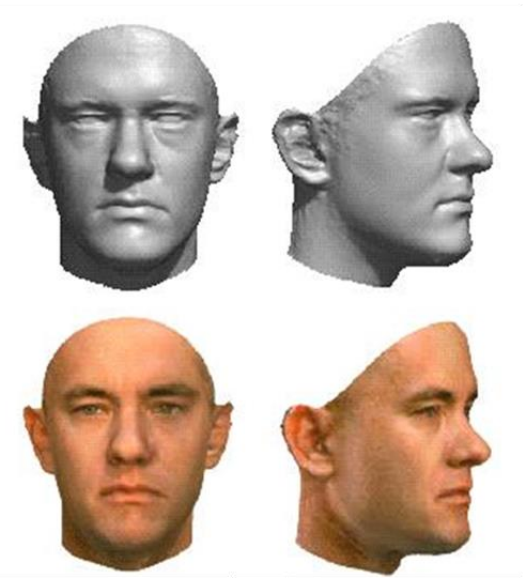
- Specs on Faces
- Virtual Makeup Database
- MIT CBCL Face Recognition Database
- Basel Face Model

(a) Specs on Faces

(b) Virtual Makeup database



(c) MIT CBCL Face Recognition Database



(d) Basel Face Model

Problems

- **Disguised face training data** is crucial for robustness against disguise face attack

- **Lack of enough data** for disguised face recognition

- Collection of disguise face database is challenging

Keypoints of study

- A new **Synthetic Disguised Face Database** is presented
- A **method** to generate synthetic disguised faces is presented
- Synthetic Data Generation may lead to subpar images
- **Automatic filtering algorithm** is presented that can filter out the low-quality generations

Contributions

1. A synthetic disguise face database namely, **"Synthetic Disguised Face Database"** is presented that features synthetically generated disguised faces with **13 add-ons and 7 combinations of add-ons** that provide a challenging task of face recognition under disguise.

2. **A technique to generate synthetic disguised faces is proposed that can enable researchers to extend several existing face databases**. The methodology can be applied to generate disguise add-ons not covered in this study thanks to the robustness of Generative Adversarial Networks and Cycle-consistency loss.

3. This technique can also be used for **runtime data augmentation for face recognition algorithm training**. The presented study has shown robustness towards face recognition while testing on faces with real disguise add-ons.

4. A comprehensive analysis is done by **benchmarking the proposed "Synthetic Faces with Disguise"** database on various face recognition algorithms in **different experimental configurations**. Results are promising on real disguise add-ons and improvement is observed in the performance on non-disguised real faces.

5. An **automated filtering scheme to identify and remove the failed generations** of the proposed method is presented that can filter-out the low-quality image samples from the generated pool of synthetic images.

Face Recognition Experiments

Terminology

Table 1. This table represents the data splits that were used in the facial recognition experiments.

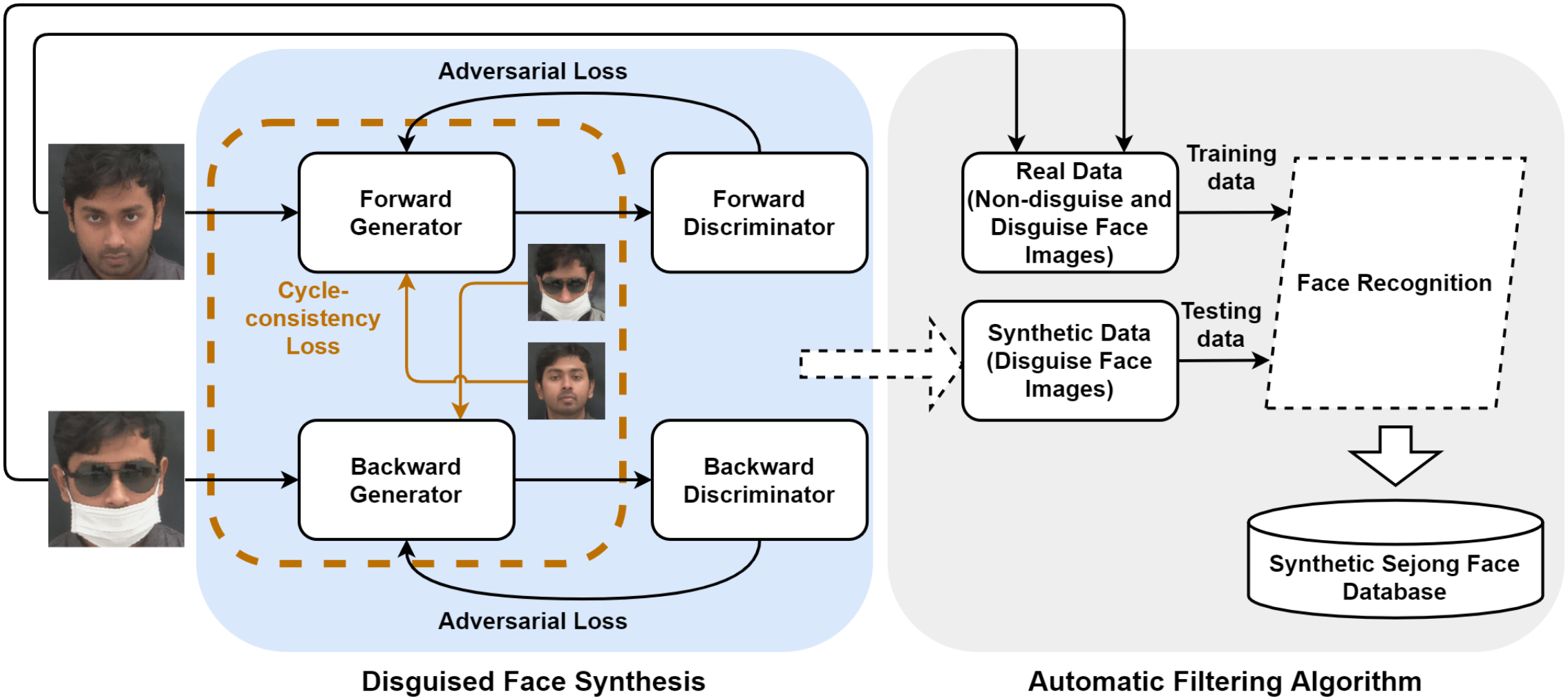
Source	Add-On	Abbreviation
Photographed	No	Real normal
Photographed	Yes	Real Disguise
Synthetically generated	Yes	Gen. Disguise

Summary of Existing Databases with Disguised Faces

Table 2. A summary of the currently available disguised face databases and the proposed Synthetic Disguised Face Database (Syn-DFD).

Database	Total No. of Subjects	Total No. of Images (Visible)	No. of Disguise Images/Subjects (Visible)	Disguise Labels	Gender Male:Female	No. of Add-Ons	Combination Add-Ons
I^2BVSD [15]	75	681	5–9	✘	60:15	5	✔
BRSU [24]	5	35	4–12	✔	4:1	4–12	✔
SDFD [26]	54	285	10	✔	54:0	3	✔
SFD-A [13]	30	390	13	✔	16:14	13	✔
SFD-B [13]	70	5250	75	✔	44:26	13	✔
Proposed Database (Syn-DFD)	**70**	**12,600**	**180**	✔	**44:26**	**13**	✔

Overview of the Presented Study



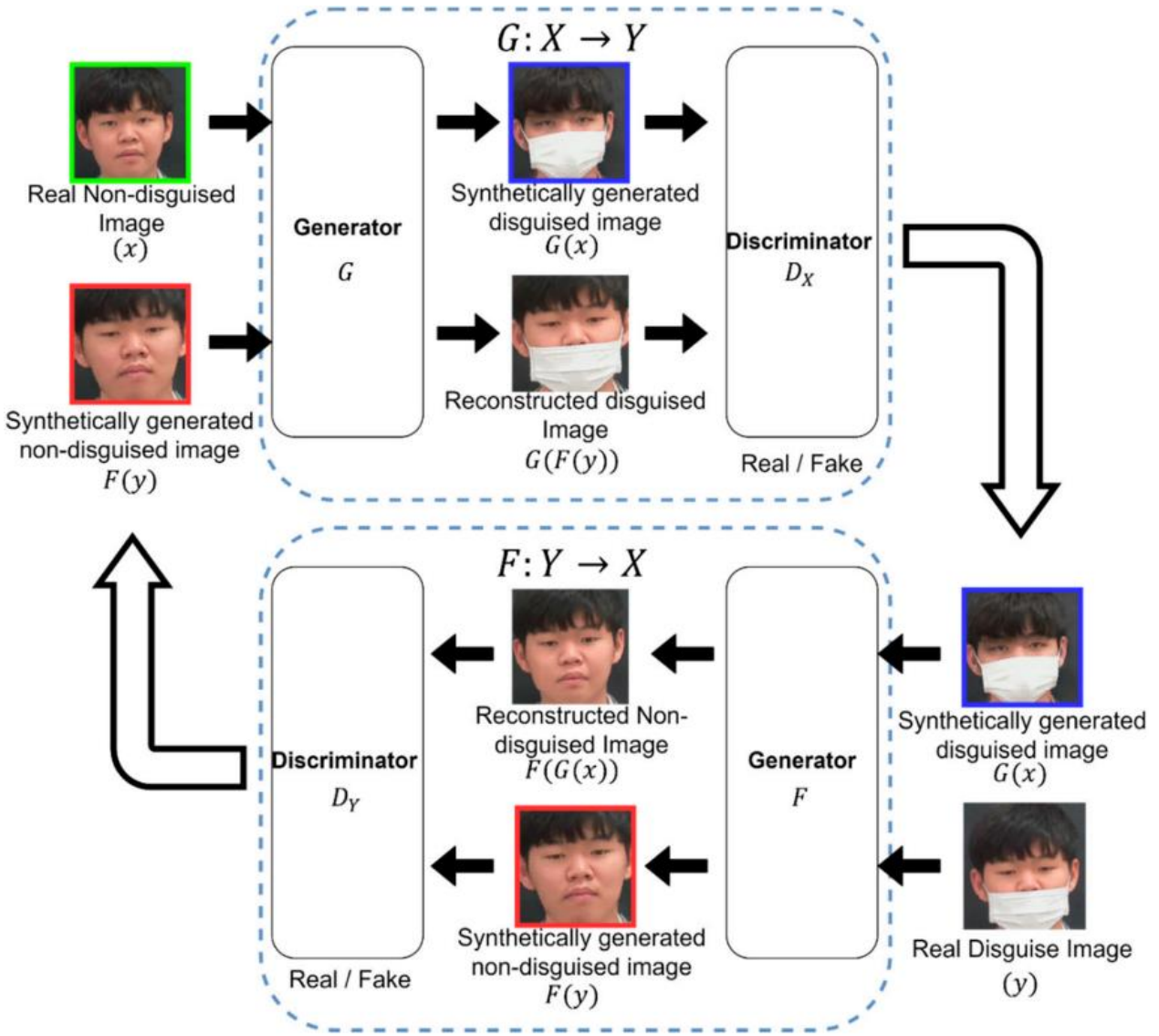
Proposed Method

$$p_{data}(x) = [p_{id}(x),\ p_{disguise}(x)] \quad (1)$$

$$p_{data}(y) = [p_{id}(y),\ p_{disguise}(y)] \quad (2)$$

$$\text{Find } G : X \rightarrow Y \quad (3)$$

$$\text{such that } p_{data}(G(x)) = [p_{id}(x),\ p_{disguise}(y)] \quad (4)$$



Proposed Method

$$\mathcal{L}_{GAN}(G,\ D_Y, X, Y) = \mathrm{E}_y[\log(D_Y(y))] + \mathrm{E}_x[\log(1 - D_Y(G(x)))] \tag{5}$$

$$\mathcal{L}_{GAN}(F,\ D_X, X, Y) = \mathrm{E}_x[\log(D_X(x))] + \mathrm{E}_y[\log(1 - D_X(F(y)))] \tag{6}$$

$$G^* = argmin_G max_{D_Y}(\mathcal{L}_{GAN}(G,\ D_Y, X, Y)) \tag{7}$$

$$F^* = argmin_F max_{D_X}(\mathcal{L}_{GAN}(F,\ D_X, X, Y)) \tag{8}$$

$$x \to G(x) \to F(G(x)) \approx x \tag{9}$$

$$y \to F(y) \to G(F(y)) \approx y \tag{10}$$

$$\mathcal{L}_{Cycle}(G, F) = \mathrm{E}_x[||F(G(x)) - x||_1] + \mathrm{E}_y[||G(F(y)) - y||_1] \tag{11}$$

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{GAN}(G,\ D_Y, X, Y) + \mathcal{L}_{GAN}(F,\ D_X, X, Y) + \mu\mathcal{L}_{Cycle}(G, F) \tag{12}$$

$$G^*, F^* = argmin_{G,F} max_{D_X, D_Y}(\mathcal{L}(G, F, D_X, D_Y)) \tag{13}$$

Convergence Analysis

The global optimality can be defined as:

$$p_G = p_{data}(y) \tag{14}$$

where p_G is the distribution of Generator G, and $p_{data}(y)$ is the data distribution of Domain Y.

Convergence Analysis

Proposition 1. *For a given generator, G, the optimal discriminator, D_Y^*, can be defined as follows:*

$$D_Y^*(y) = \frac{p_{data}(y)}{p_{data}(y) + p_G(y)} \tag{15}$$

Proof. The training criterion for the discriminator, D_X, given Generator G, is to maximize the quantity, $\mathcal{L}(G, D_Y, X, Y)$:

$$\begin{aligned} \mathcal{L}(G, D_Y, X, Y) &= \int_y p_{data}(y) \log(D_Y(y)) dy + \int_x p_{data}(x) \log(1 - D_Y(G(x)) dx \\ &= \int_y p_{data}(y) \log(D_Y(y)) + p_F(x) \log(1 - D_Y(y)) dy \end{aligned} \tag{16}$$

Here, we can use the proof from [19] for $G : X \to Y$. For any $(a, b) \in \mathbb{R}^2 \{0, 0\}$, the function $y \to a \log y + b \log(1 - y)$ achieves its maximum in $[0, 1]$ at $\frac{a}{a+b}$. The discriminator, D_Y, does not need to be defined outside of $Supp(p_{data}(y) \cup Supp(p_G)$, and the same holds for the discriminator, D_X, i.e., $Supp(p_{data}(x) \cup Supp(p_F)$, thus concluding the proof. □

Convergence Analysis

The training objective for D_Y can be interpreted as maximizing the log-likelihood for estimating the conditional probability, $P(S = s|y)$, where S indicates whether y comes from $p_{data}(y)$, i.e., $y = 1$, or from p_G, i.e., $(y = 0)$. The minimax problem in Equation (6) can be reformulated as:

$$\begin{aligned} C(G) &= max_{D_Y} \mathcal{L}(G, D_Y, X, Y) \\ &= \mathrm{E}_{y \sim p_{data}(y)} \left[\log(D_Y^*(y))\right] + \mathrm{E}_{x \sim p_{data}(x)} \left[\log(1 - D_Y^*(G(x)))\right] \\ &= \mathrm{E}_{y \sim p_{data}(y)} \left[\log(D_Y^*(y))\right] + \mathrm{E}_{y \sim p_G} \left[\log(1 - D_Y^*(y))\right] \\ &= \mathrm{E}_{y \sim p_{data}(y)} \left[\log \frac{p_{data}(y)}{p_{data}(y) + p_G}\right] + \mathrm{E}_{y \sim p_G} \left[\log \frac{p_G(y)}{p_{data}(y) + p_G}\right] \end{aligned} \tag{17}$$

Theorem 1. *The virtual training criterion defined as $C(G)$ in Equation (17) achieves the value of $-\log 4$at global minimum.*

Proof. For the global optimality proven in Proposition 1, i.e., $p_G = p_{data}(y)$, the optimal discriminator is $D_Y^*(y) = \frac{1}{2}$ (Equation (15)). Solving Equation (17) for $D_Y^*(y) = \frac{1}{2}$, we get $C(G) = \log \frac{1}{2} + \log \frac{1}{2} = -\log 4$. It can be seen that this is the best possible value of $C(G)$, reached only for $p_G = p_{data}(y)$. Observe that:

$$\mathrm{E}_{y \sim p_{data}(y)}[-\log 2] + \mathrm{E}_{y \sim p_G}[-\log 2] = -\log 4$$

Moreover, by subtracting this expression from $C(G) = \mathcal{L}(G, D_Y, X, Y)$, we obtain:

$$C(G) = -\log 4 + KL\left(p_{data}(y) \parallel \frac{p_{data}(y) + p_G}{2}\right) + KL\left(p_G \parallel \frac{p_{data}(y) + p_G}{2}\right) \tag{18}$$

Convergence Analysis

where KL is the Kullback–Leibler divergence. In the previous expression, the Jensen–Shannon divergence can be observed between the generator's distribution and the data-generating process:

$$C(G) = -\log 4 + 2 \cdot JSD(p_{data}(y) \parallel p_G) \tag{19}$$

Since the Jensen–Shannon divergence between two distributions is zero only when they are equal and always non-negative, it is shown that $C^* = -\log 4$ is the global minimum of $C(G)$, and that the only solution is $p_G = p_{data}(y)$. That is, the generator model perfectly replicates the generating process. □

Convergence Analysis

Proposition 2. *If Generator G and Discriminator D have enough capacity, and the discriminator is allowed to reach its optimum at every training iteration given G, and p_G is updated according to the criterion:*

$$\mathrm{E}_x[\log(D_X(x))] + \mathrm{E}_y[\log(1 - D_X(F(y)))]$$

then p_G converges to $p_{data}(y)$.

Proof. Consider the function, $\mathcal{L}(G, D_Y, X, Y) = Q(p_G, D_Y, X, Y)$, as a function of p_G, as performed in the aforementioned criterion. It is to be noted that $Q(p_G, D_Y, X, Y)$ is convex in p_G. The derivative of the function is included in the subderivatives of a supremum of convex functions at the point where the maximum is attained [19]. In other words, if $f(x) = sup_{\alpha \epsilon A}\, f_\alpha(x)$ and $f_\alpha(x)$ is convex in x for every α, then $\partial f_\beta\ (x) \epsilon\ \partial f$ if $\beta = \mathrm{arg}sup_{\alpha \epsilon A}\, f_\alpha(x)$. This is equivalent to computing a gradient descent update for p_G at the optimal D_Y, given the corresponding generator, G. $sup_{D_Y} U(p_G, D_Y)$ is convex in p_G, with a unique global optimum, as given in Theorem 1. Hence, with sufficiently small gradient updates of p_G, it can converge to $p_{data}(y)$. Hence, the proof is concluded. □

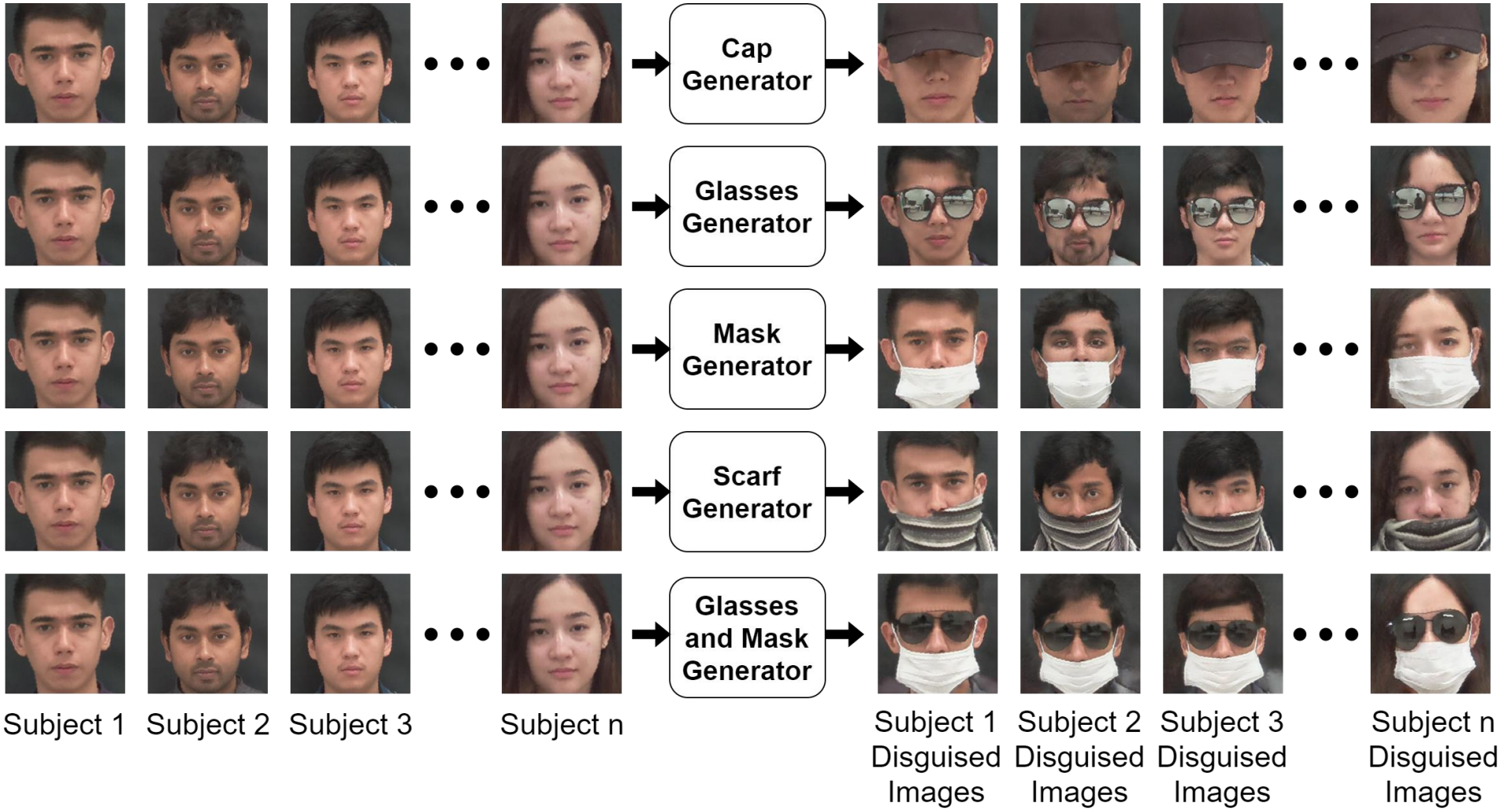
Convergence Analysis

The convergence analysis is provided for $G : X \rightarrow Y$. Similarly, the proof of convergence can be derived for the mapping, $F : Y \rightarrow X$. Furthermore, the cycle-consistency loss is calculated for the mappings, $G : F(Y) \rightarrow Y$ and $F : G(X) \rightarrow X$, by alternating the generator inputs between real samples, $[X, Y]$, and generations, $[F(Y), G(X)]$. Equation (16) can be written for the complete loss function mentioned in Equation (12) as follows:

$$
\begin{aligned}
\mathcal{L}(G, F, D_X, D_Y) &= \int_y p_{data}(y) \log(D_Y(y)) dy + \int_x p_{data}(x) \log(1 - D_Y(G(x)) dx \\
&+ \int_x p_{data}(x) \log(D_X(x)) dx + \int_y p_{data}(y) \log(1 - D_X(F(y)) dy \\
&+ \int_y p_{data}(y) \log(D_Y(y)) dy + \int_{F(y)} p_{data}(F(y)) \log(1 - D_Y(G(F(y)))) d(F(y)) \\
&+ \int_x p_{data}(x) \log(D_X(x)) dx + \int_{G(x)} p_{data}(G(x) \log(1 - D_X(F(G(x)))) d(G(y)) \\
&= \int_y p_{data}(y) \log(D_Y(y)) + p_F(y) \log(1 - (D_Y(y)) dy \\
&+ \int_x p_{data}(x) \log(D_X(x)) + p_G(x) \log(1 - (D_X(x)) dx \\
&+ \int_{F(y)} p_{data}(F(y)) \log(D_X(F(y))) + p_G(G(x)) \log(1 - (D_X(F(y))) d(F(y)) \\
&+ \int_{G(x)} p_{data}(G(x)) \log(D_Y(G(x))) + p_F(F(y)) \log(1 - (D_Y(G(x))) d(G(x))
\end{aligned}
\tag{20}
$$

Propositions 1 and 2 also hold for Equation (20), as the additional terms for cycle-consistency loss simply use the generated output for the same mapping. Given the infinite capacity for all the generators and discriminators, it can theoretically converge to the optimum.

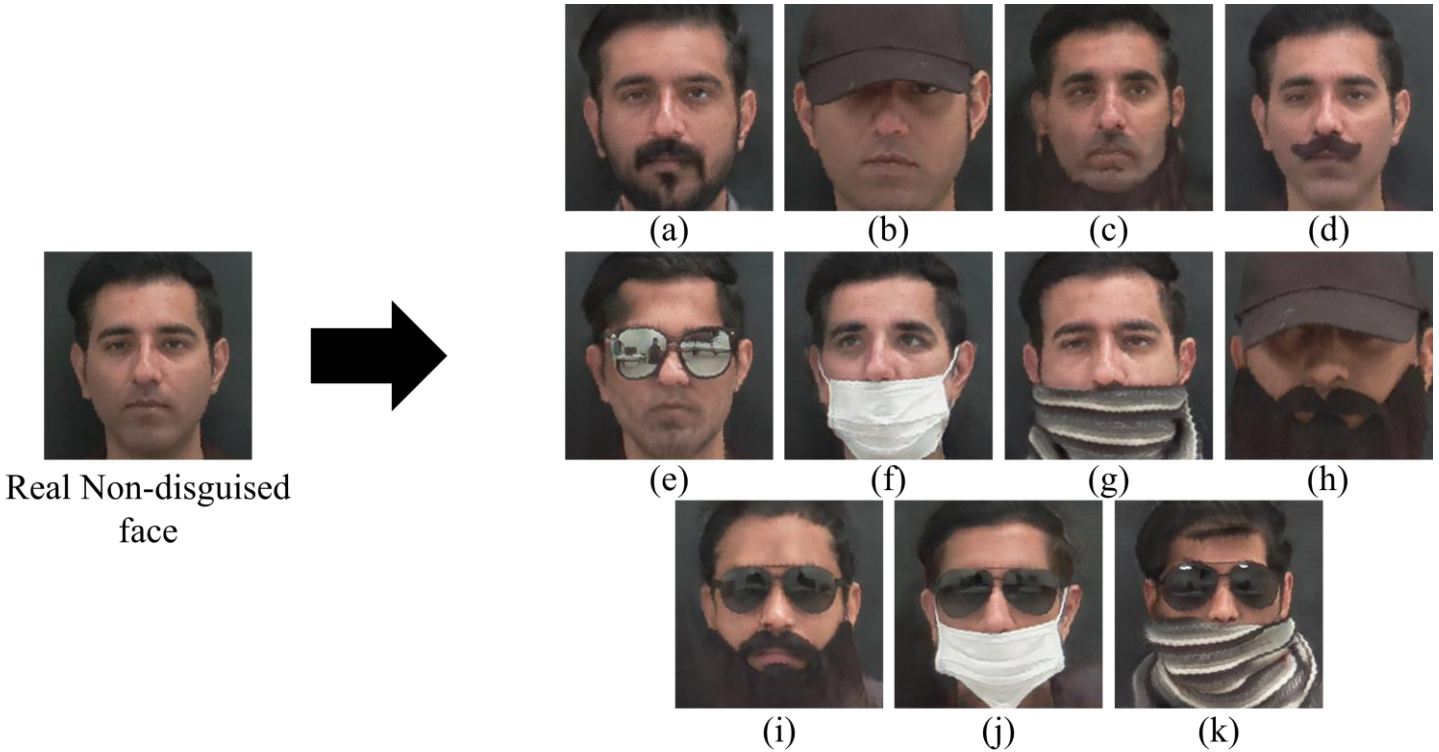
Generating Disguised Face Images using Generator

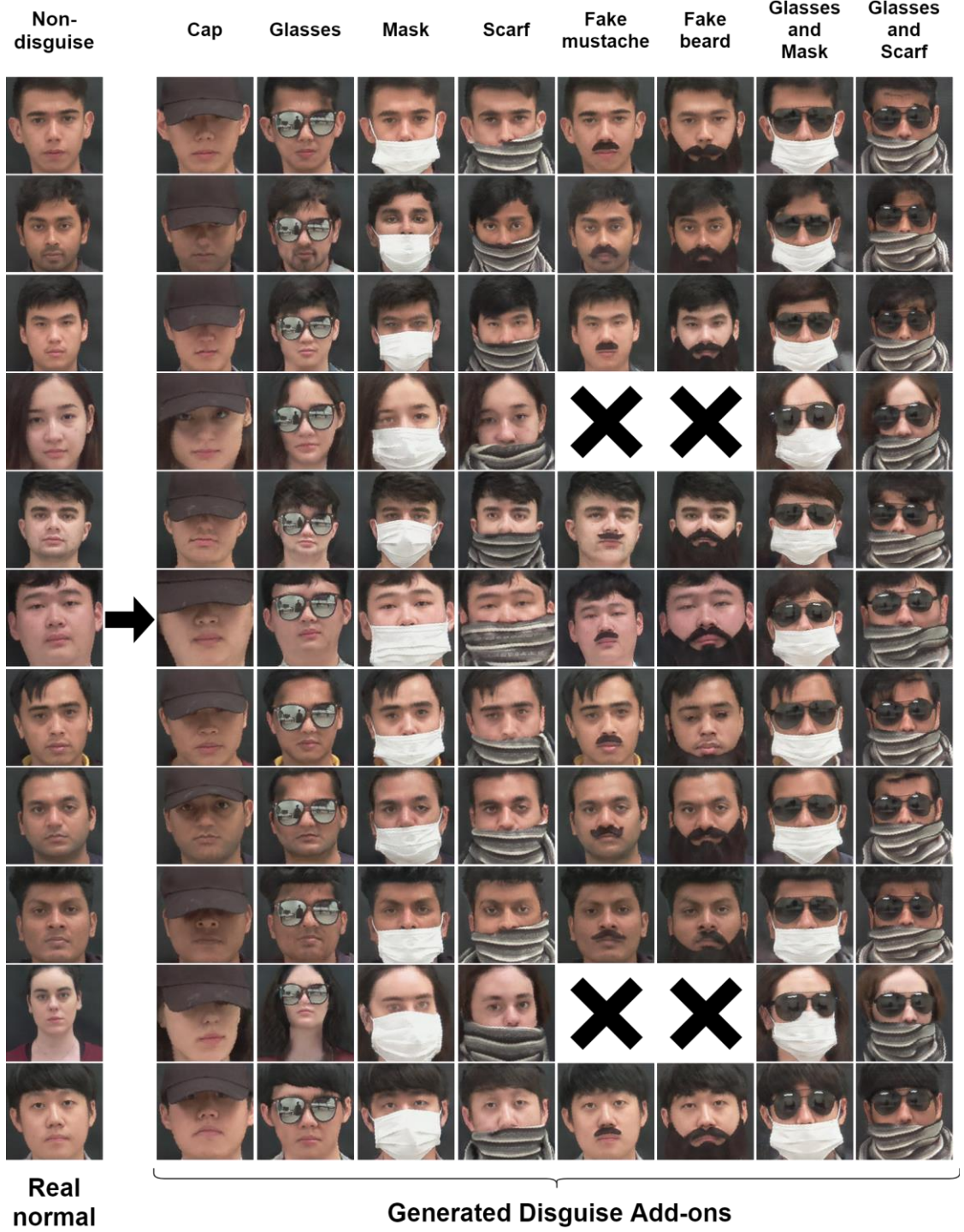


Qualitative Results

Table 3. Quantitative results of the presented study on normal-to-disguised-face generation. The KID means and standard deviations are provided.

Data	FID	KID (Mean ± Standard Deviation)
Real data splits (baseline)	30.967	0.01128 ± 0.00059
Synthetic Images	38.177	0.01684 ± 0.00037




Non-disguise
Cap
Glasses
Mask
Scarf
Fake mustache
Fake beard
Glasses and Mask
Glasses and Scarf
Real normal
Generated Disguise Add-ons

Comparison of data in Seed and Proposed Database

Table 4. The types of disguised add-ons available in the SFD [13] and Syn-DFD, along with the total number of images, and information about the gender of subjects. It is to be noted that some add-ons are gender-specific.

Add-On	Add-On Name	Number of Images		Gender	
		Sejong Face Database	Proposed Database	Male	Female
No Add-on	Natural Face	15	-	✔	✔
	Real Beard	10	15	✔	✘
Accessory Add-on	Cap	5	15	✔	✔
	Scarf	5	15	✔	✔
	Glasses	5	15	✔	✔
	Mask	5	15	✔	✔
	Makeup	5	15	✘	✔
Fake Add-on	Wig	10	15	✘	✔
	Fake Beard	5	15	✔	✘
	Fake Mustache	5	15	✔	✘
Combination Add-on	Wig + Glasses	5	15	✘	✔
	Wig + Scarf	5	15	✘	✔
	Cap + Scarf	5	15	✔	✔
	Glasses + Scarf	5	15	✔	✔
	Glasses + Mask	5	15	✔	✔
	Fake Beard + Cap	5	15	✔	✘
	Fake Beard + Glasses	5	15	✔	✘

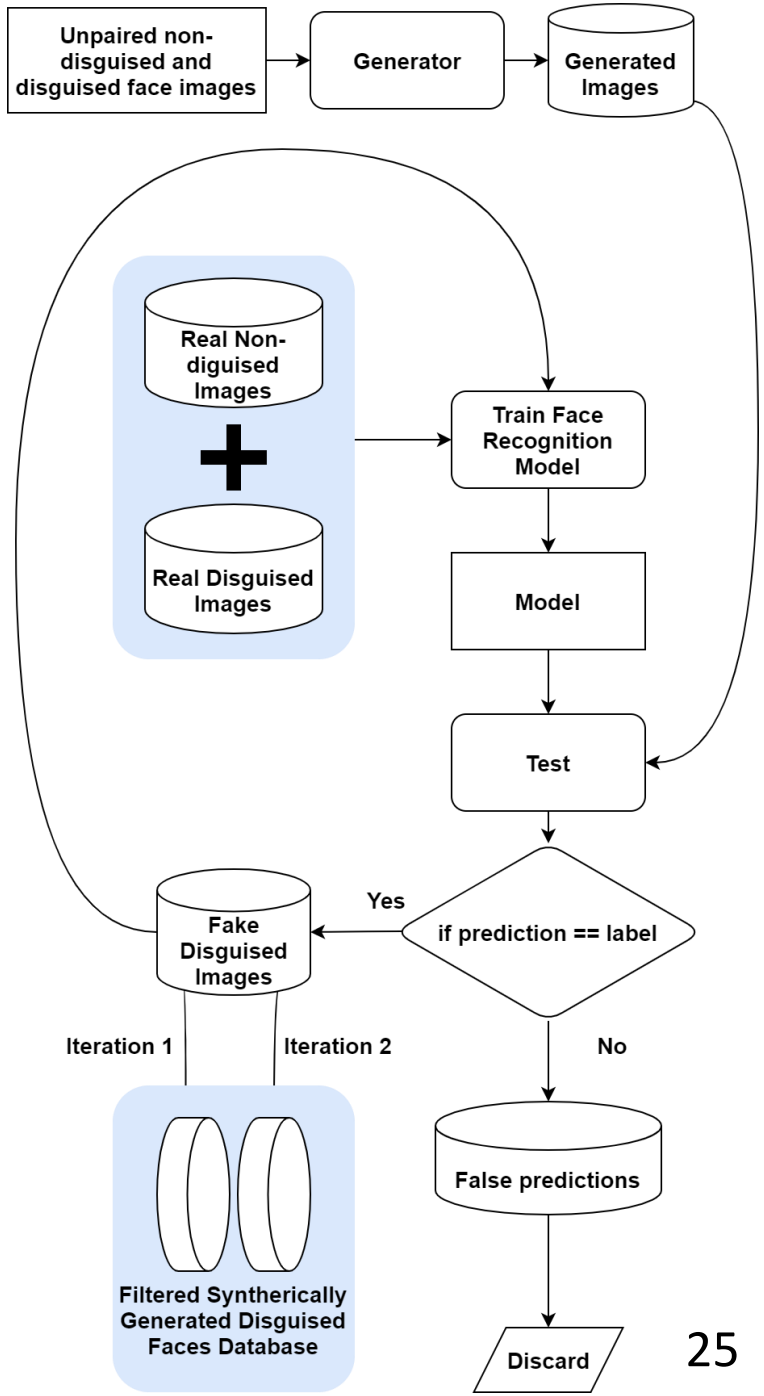
Bad Results

- GANs are prone to output poor-quality results
- poor generations can lead to degradation of the FR system
- when the subject identity is 442 hidden to a greater extent
- common in combination add-ons such as “scarf and glasses”, 444 “scarf and wig”, “mask and glasses”, and “fake beard and cap”

Automatic Filtering

Algorithm 1: Pseudocode of the proposed automated filtering algorithm.

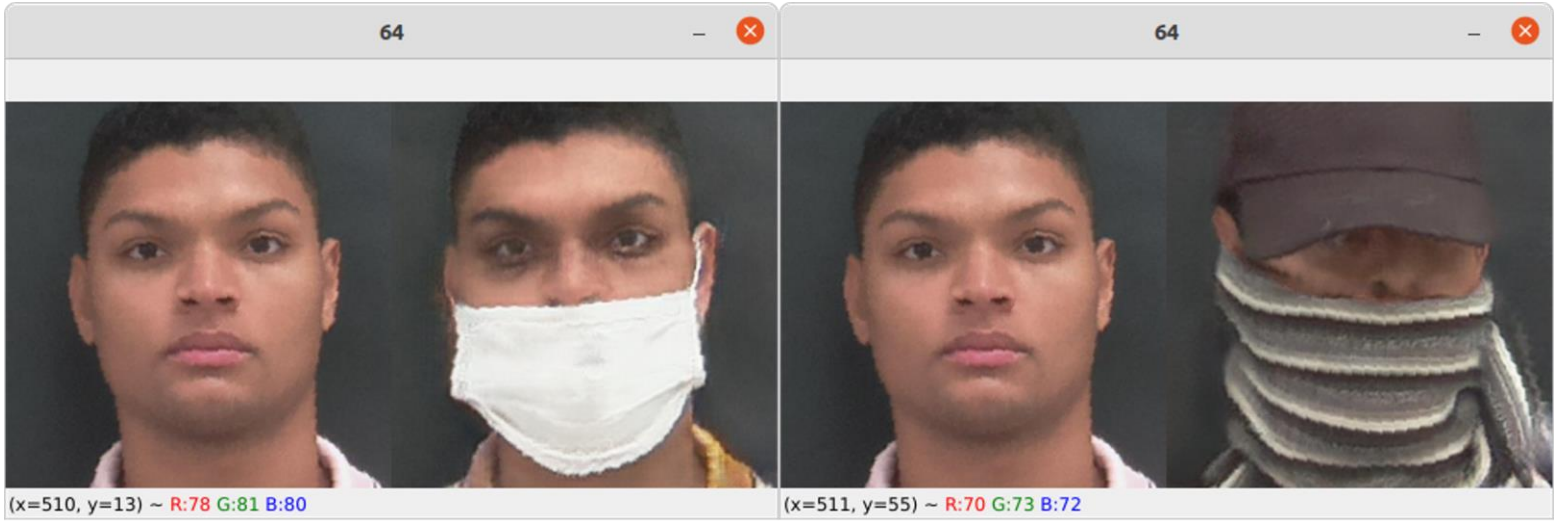
```
1. Initialization:
2.       Real_Normal[]
3.       Real_Addons[]
4.       Normal_to_Addon_Generator()
5.       filtered_Gen_Addons[]
6.       false_predictions[]
7.    Gen_Addons = Normal_to_Addon_Generator(Real_Normal, Real_Addons)
8.    FR_model = SqueezeNet.train(Real_Normal, Real_Addons)
9.    predictions[] = FR_model.predict(Gen_Addons)
10. if predictions[x] == ground_truth[x]:
11.    filtered_Gen_Addons.append(predictions[x])
12. else:
13.    false_predictions.append(predictions[x])
14.    FR_model_2 = SqueezeNet.train(Real_Normal, Real_Addons+fi ltered_Gen_Addons)
15.    predictions[] = FR_model.predict(false_predictions)
16. if predictions[x] == ground_truth[x]:
17.    filtered_Gen_Addons.append(predictions[x])
18.    Gen_DB.save(filtered_Gen_Addons)
19. else:
20.    false_predictions.append(predictions[x])
21.    false_predictions = Null
```



Manual Filtering

- The observer is asked to accept or reject the sample based on the following criteria:

  - Both images must be recognizable as the same person.
  - The image quality must be lifelike.
  - There should be no discrepancy between the original and generated samples such as normalcy of facial features.
  - Ensuring the face is not completely hidden by the generated disguise add-on.



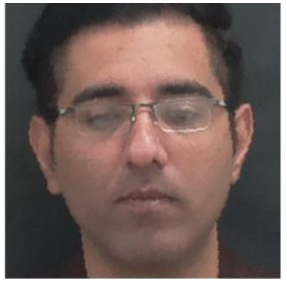
(a) Acceptable (b) Not acceptable

Automatic and Manual Filtering

Table 5. The number of synthetically generated images, before and after applying the automatic filtering algorithm.

Method	Total Number Images	Images/Subjects
No Filtering	12,600	180
Manual Filtering	6780	88
Automatic filtering	4158	60

Within add-on variations



Real image with glasses

Generated image with glasses and mask

Generated image with glasses

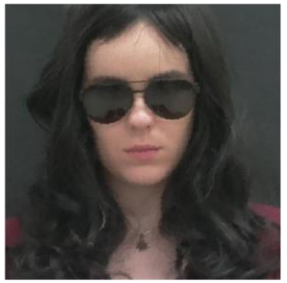
(a)

Real image with fake beard

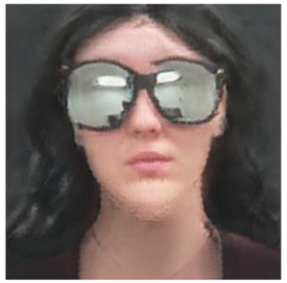
Generated image with fake beard

Generated image with fake beard

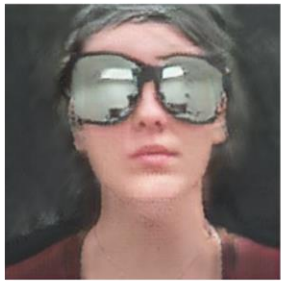
(b)



Real image with wig and glasses



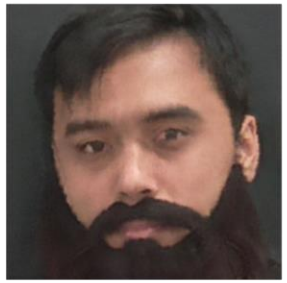
Generated image with wig and glasses



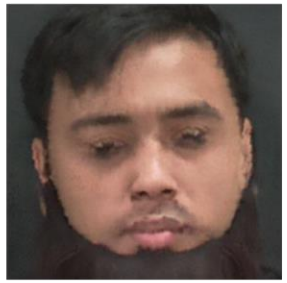
Generated image with wig and glasses

(c)

Real image with fake beard



Generated image with fake beard



Generated image with fake beard

(d)

Face Recognition Experiments

Table 6. This table presents the configurations of the facial recognition experiments conducted to demonstrate the efficacy of the synthetically generated disguised faces. Different data configurations were used in experiments for the sake of comparison and were tested on the set of real add-ons.

Training Configuration	Training Data Type	Total Images	Images Per Subject	Subjects	Disguise Add-Ons
Configuration 0	Real normal	685	10	All	All
Configuration 1	Real normal + Real Add-on	986	15 (4 + 11)		
Configuration 2	Real normal + Gen. Add-on	1240	19 (4 + 15)		
Configuration 3	Real normal + Real Add-on + Gen. Add-on	1276	20 (4 + 8 + 8)		
Configuration 4	Gen. Add-on	1486	23		
Configuration 5	Real normal + Gen. Add-on (Manual Filtering)	1240	19 (4 + 15)		
Configuration 6	Real normal + Gen. Add-on (Automatic Filtering)	1240	19 (4 + 15)		

Face Recognition Experiments

Table 7. The test data used for evaluation of facial recognition models trained with the presented training configurations.

Test Set	Data Description	Number of Total Images	Number of Images Per Subject
Real Normal *	Photographed images of nondisguised faces (reduced set used for Configuration 0)	251	4
Real Normal	Photographed images of nondisguised faces	564	9
Real Add-on	Photographed images of disguised faces	2272	36

* Real Normal used for testing Configuration 0 has less test images.

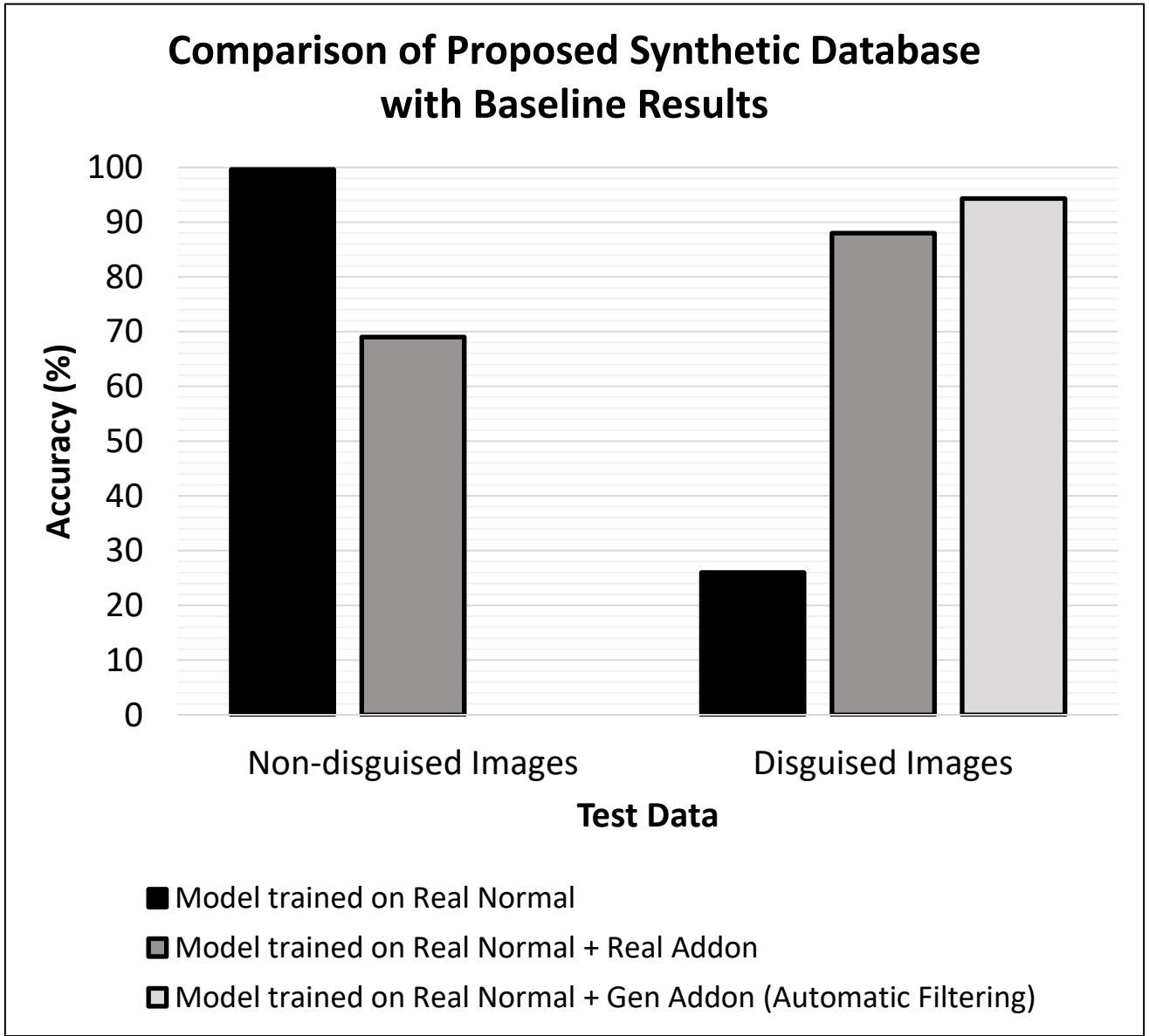
Face Recognition Experiments

Table 8. This table presents the facial recognition results achieved on the real nondisguised (real normal) and the disguised (real add-on) faces by the models trained using the training configurations mentioned in Table 6.

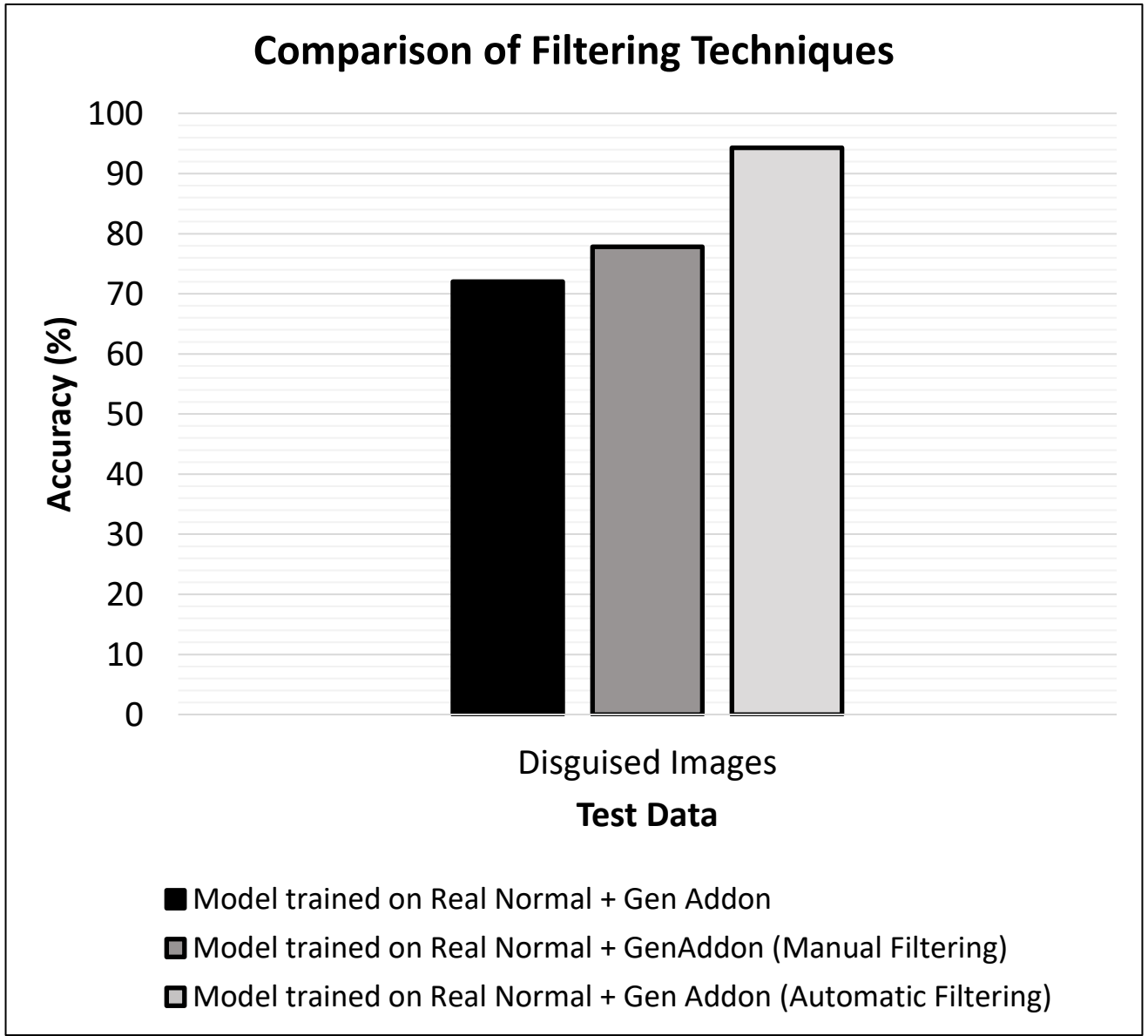
Training Configuration	Training Data Type	Accuracy (%)	
		Real Normal	Real Add-On
Configuration 0	Real normal	99.6 *	26
Configuration 1	Real normal + Real Add-ons	69	88
Configuration 2	Real normal + Gen. Add-ons	86	72
Configuration 3	Real normal + Real Add-ons + Gen. Add-ons	62	89
Configuration 4	Gen. Add-ons	74	71
Configuration 5	Real normal + Gen. Add-ons w/Manual Filtering	-	77.8
Configuration 6	Real normal + Gen. Add-ons w/Automatic Filtering	-	94.3

* Real Normal used for testing Configuration 0 has less test images, as shown in Table 7.

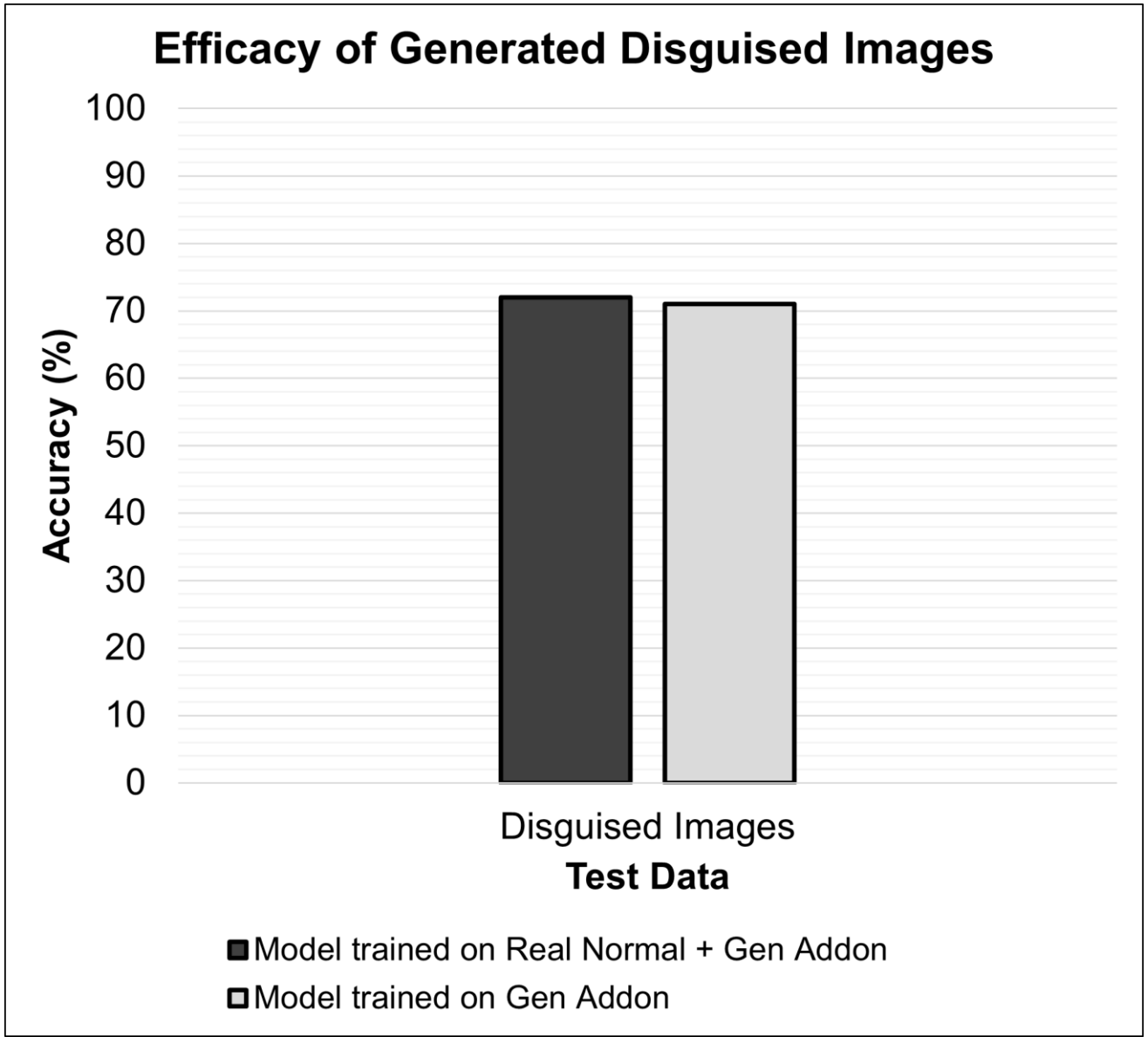
Face Recognition Results



Face Recognition Results



Face Recognition Results



Computational Complexity

Table 9. Computational complexity and model sizes of the proposed methodology are presented in the form of the number of multiple-add operations and the number of trainable parameters. It is to be noted that, during the inference phase, only one generator is used; therefore, the computational complexity during inference time is only based on Generator G.

Models	Computational Complexity (GMac)	Number of Parameters (Million)	Training	Inference
Generator G	56.89	11.38	✔	✔
Generator F	56.89	11.38	✔	✘
Discriminator D_X	3.15	2.76	✔	✘
Discriminator D_Y	3.15	2.76	✔	✘
Total (Training)	120.08	28.28	-	-
Total (Inference)	56.89	11.38	-	-

Extension to other domains

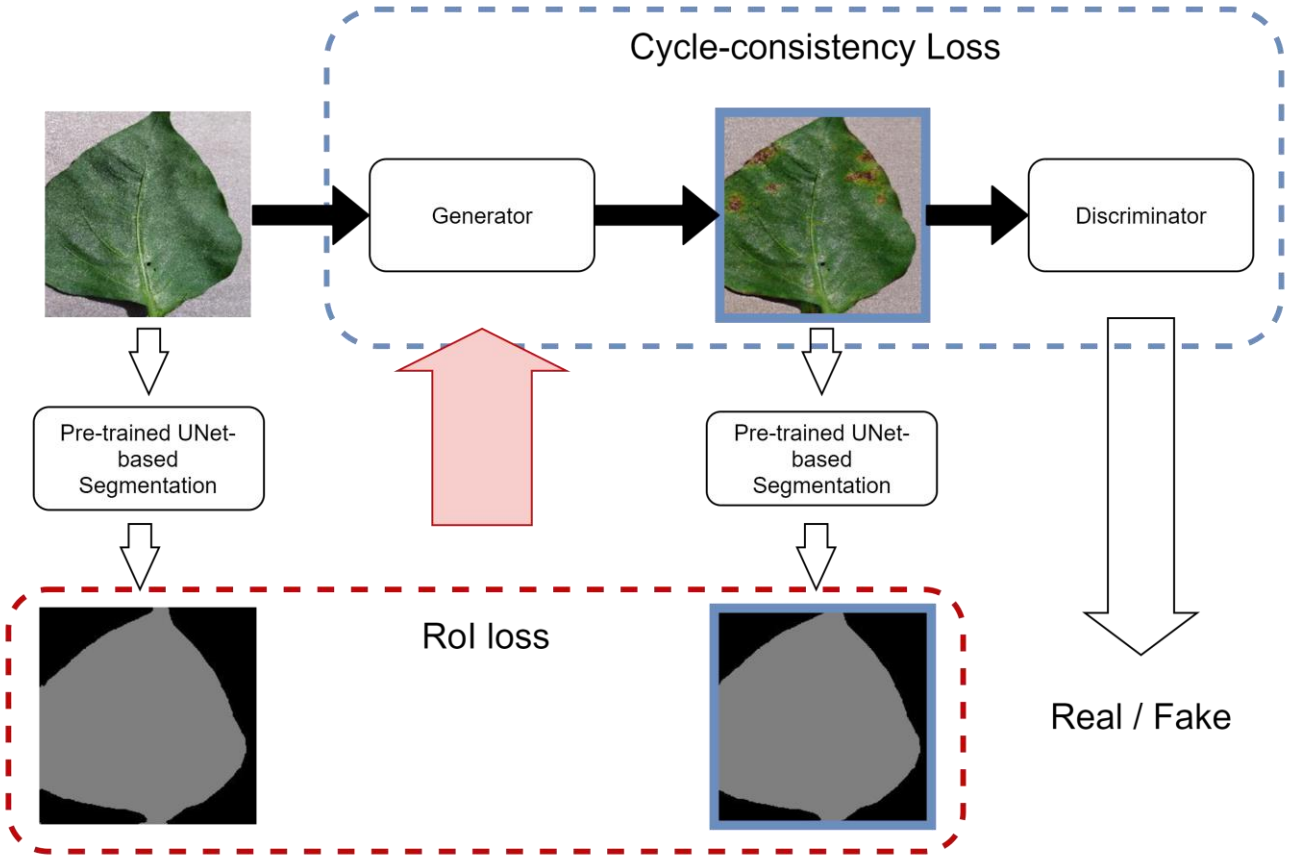
LR → HR

Plant Disease Synthesis (Problem)

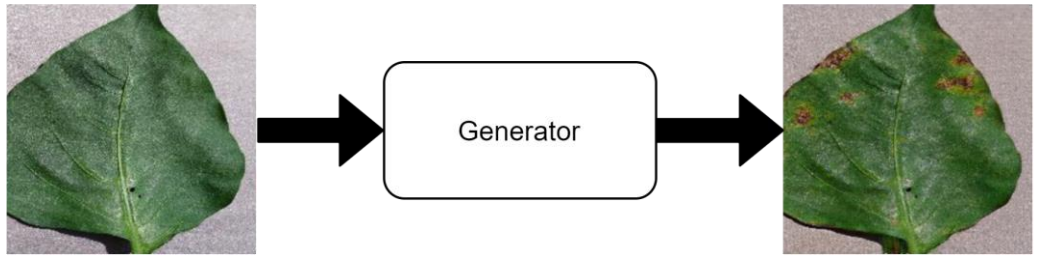
Healthy

Generated
Disease Image

Localized disease transfer using cycle-consistency loss and RoI Loss

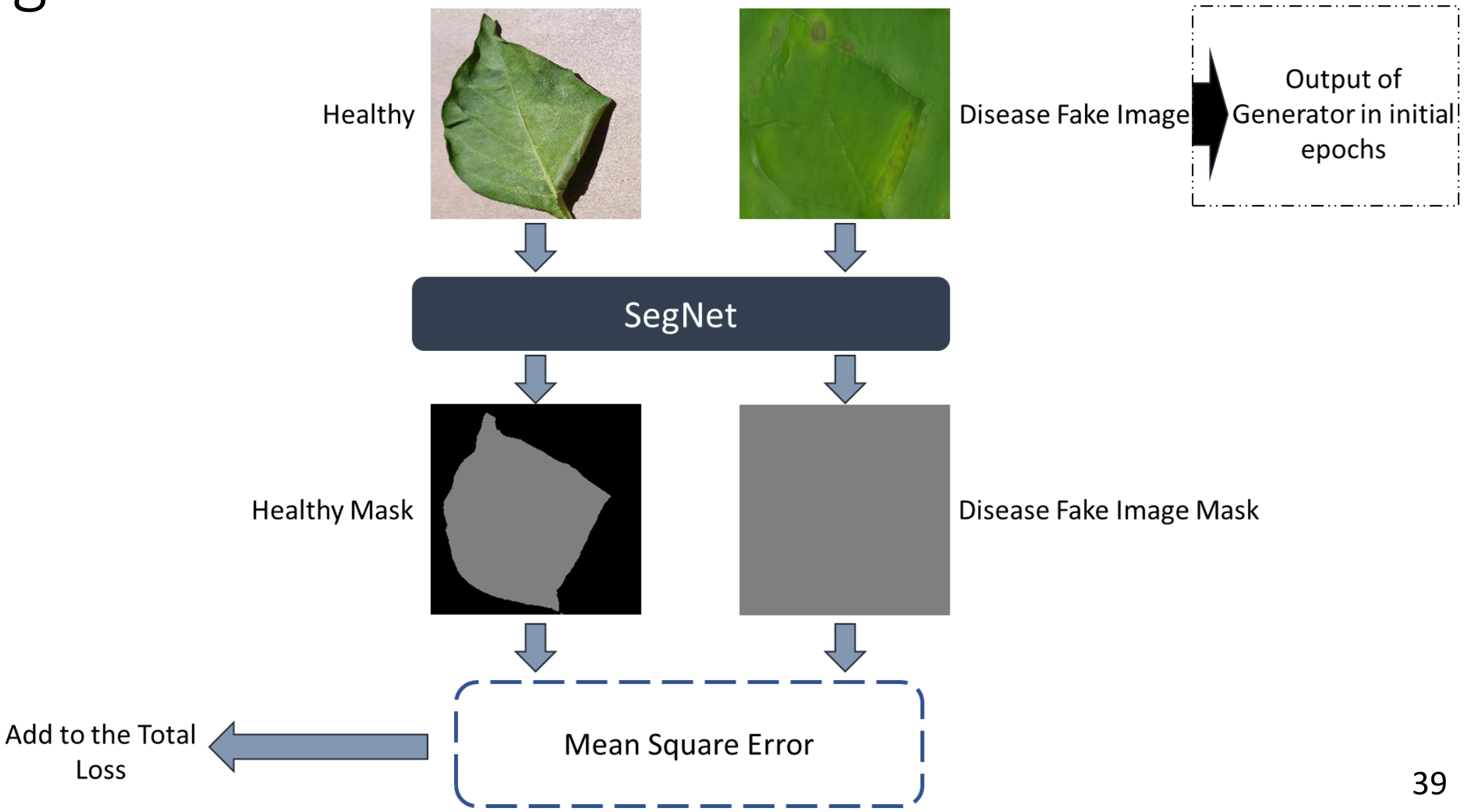


Training pipeline



Testing pipeline

Segmentation Module



Leaf Disease Synthesis

- Works well for similar datasets
- Subpar results when two domains are from different datasets.

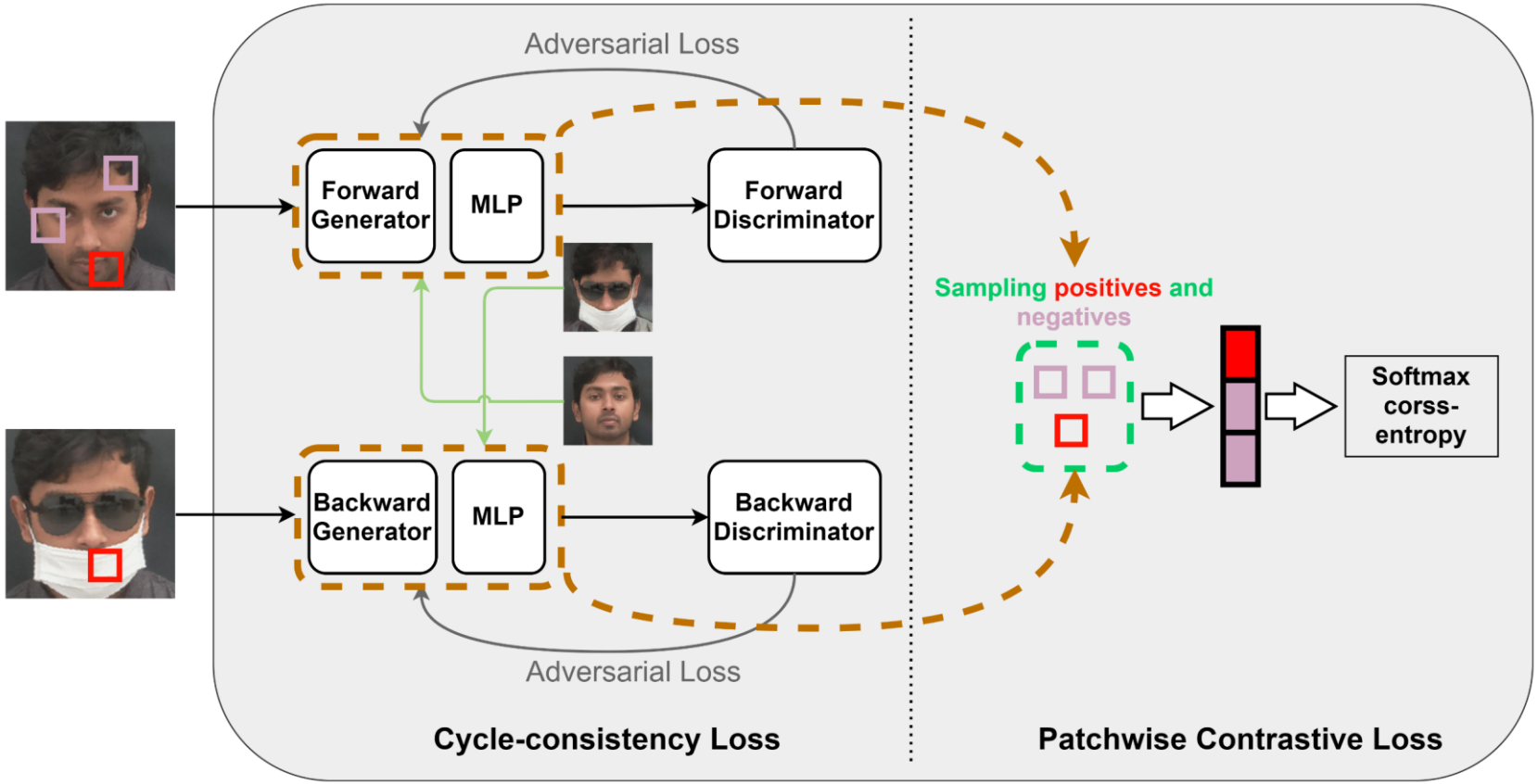
Real healthy Leaf

Generated using cycle-consistency loss

Generated using the proposed method

Future Work

- Working to achieve same results using **one training sample per domain**.



감사합니다