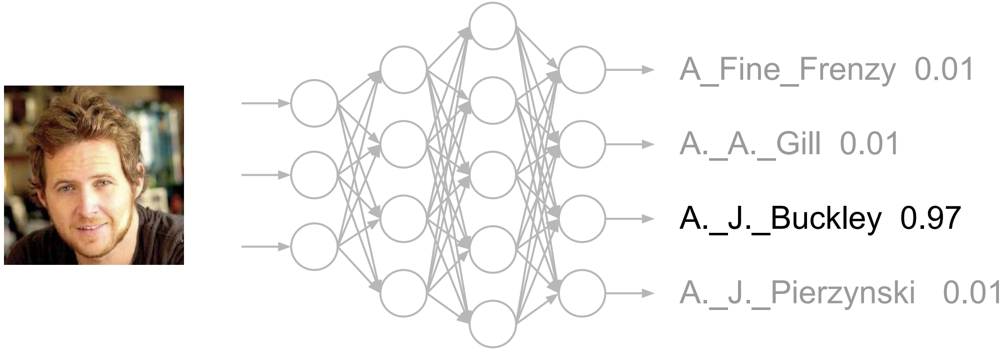
MIRROR: Model Inversion for Deep Learning Network with High Fidelity

Shengwei An, Guanhong Tao, Qiuling Xu, Yingqi Liu, Guangyu Shen, Yuan Yao, Jingwei Xu, Xiangyu Zhang

PURDUE UNIVERSITY

南京大學
NANJING UNIVERSITY

Deep Learning Classifiers



Online Commercial Services

clarifai

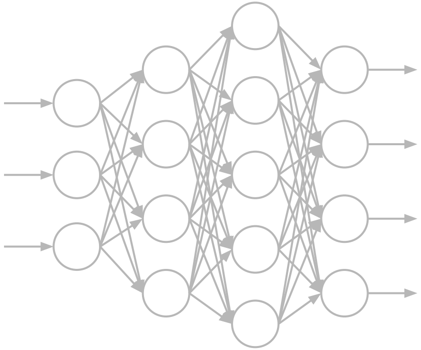
Azure

Google Cloud

aws

Model Inversion

???

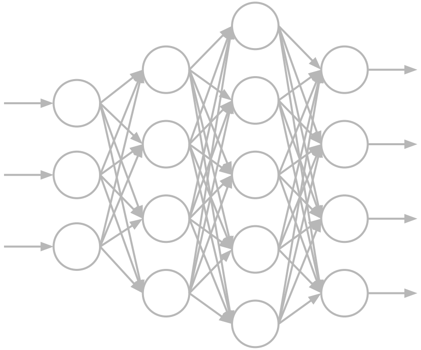


target

Goal:

Generate a representative image
Cause privacy leakage

Model Inversion

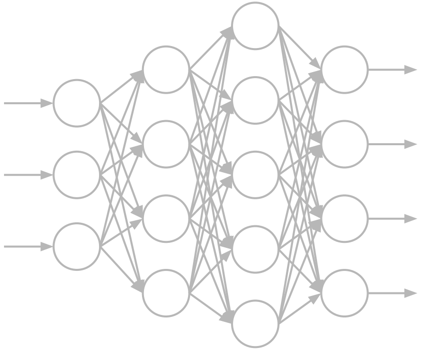


target

Goal:

Generate a representative image
Cause privacy leakage

Model Inversion



target

Goal:

Generate a representative image
Cause privacy leakage

E.g.,

Disguise themselves
Pass the classification
Cause security breach

White-box and Black-box Model Inversion

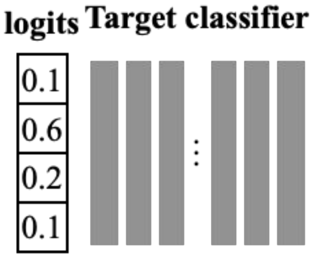
Don't know the labels or the training data.

White-box:

Have the model architecture and weights

Can access the internals

Can compute the gradients

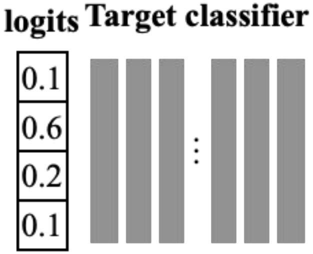


White-box and Black-box Model Inversion

Don’t know the labels or the training data.

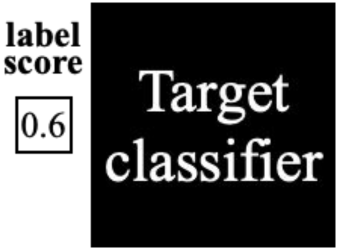
White-box:

- Have the model architecture and weights
- Can access the internals
- Can compute the gradients



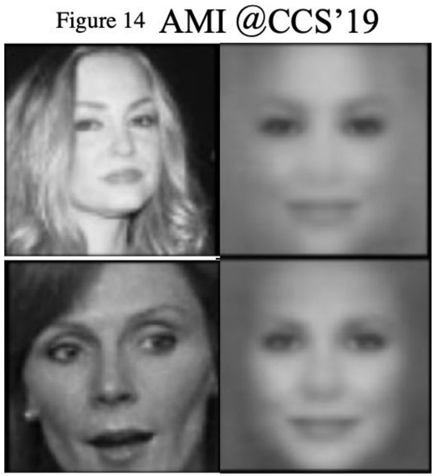
Black-box:

- Can only get the output confidence

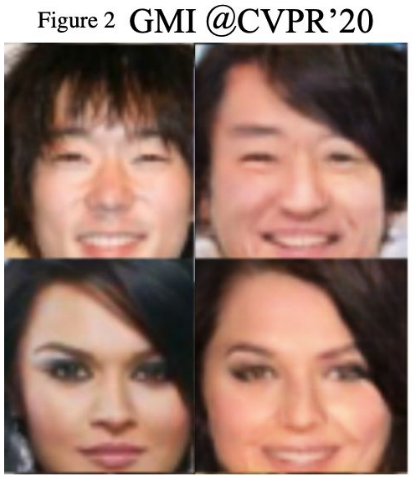


Existing Methods

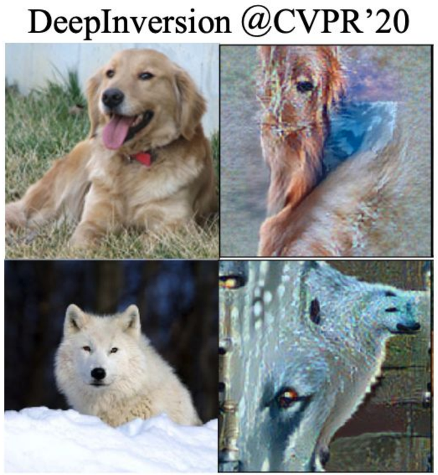
Target Inversion
White-box



Target Inversion
Black-box



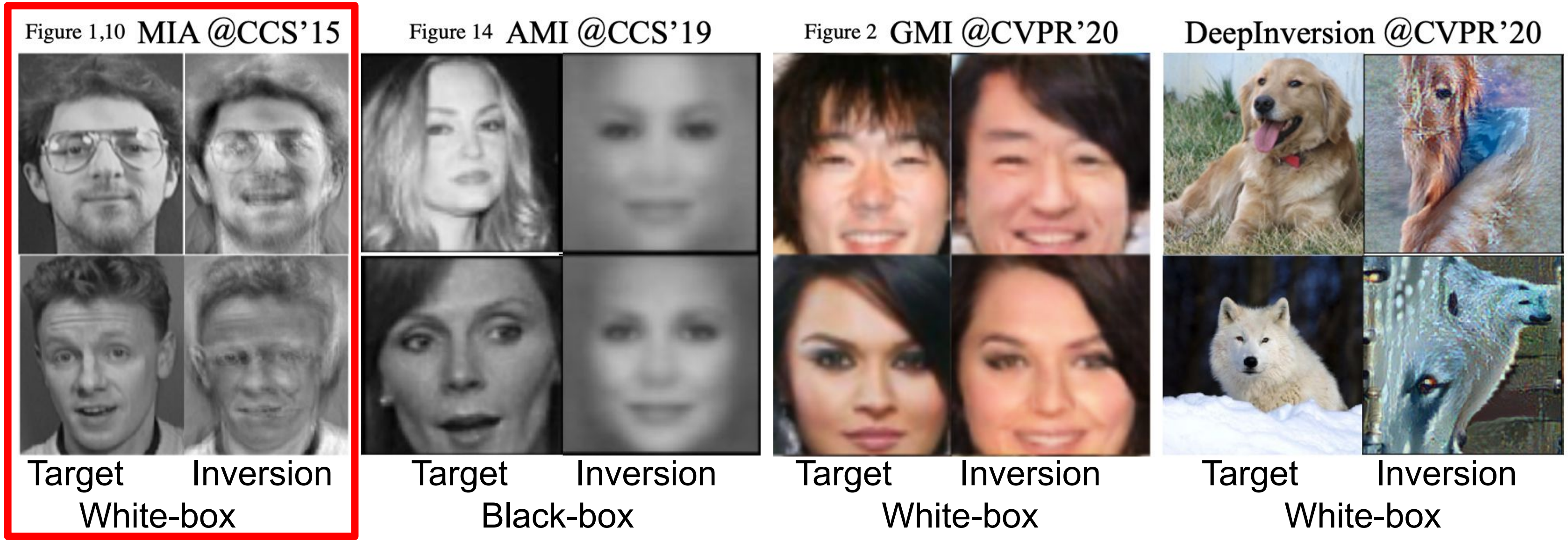
Target Inversion
White-box



Target Inversion
White-box

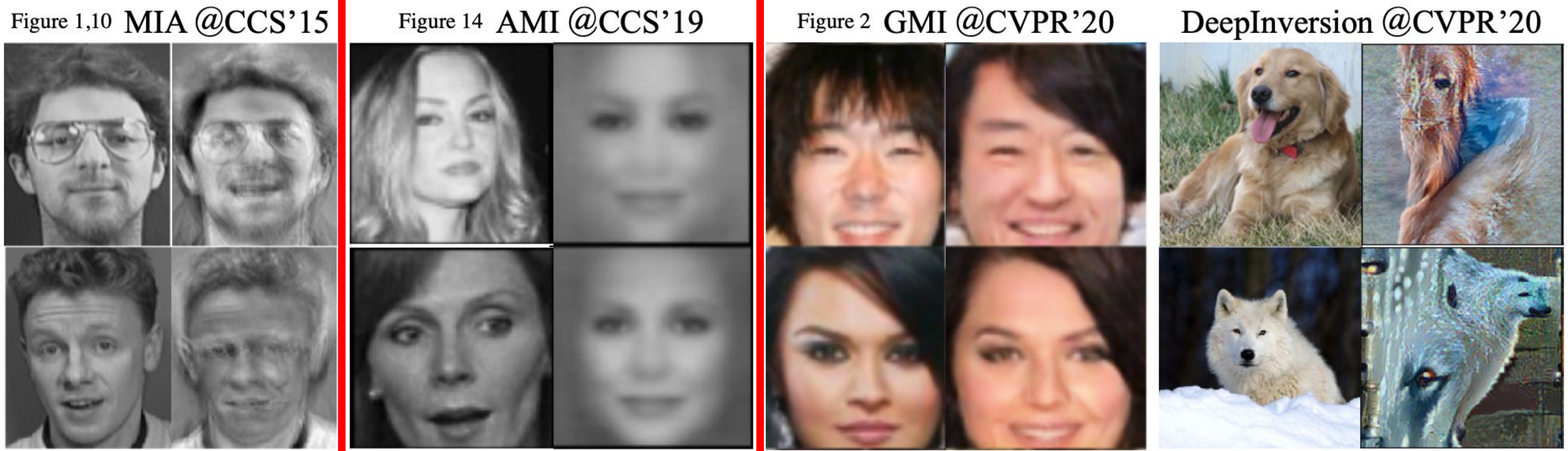
Inversion results in the top are from their original papers or the official GitHub repository.

Existing Methods



Inversion results in the top are from their original papers or the official GitHub repository.

Existing Methods



Target Inversion
White-box

Target Inversion
Black-box

Target Inversion
White-box

Target Inversion
White-box

Inversion results in the top are from their original papers or the official GitHub repository.

Existing Methods

Inversion results in the top are from their original papers or the official GitHub repository.

Existing Methods

Figure 1,10 MIA @CCS'15

Target Inversion
White-box

Figure 14 AMI @CCS'19

Target Inversion
Black-box

Figure 2 GMI @CVPR'20

Target Inversion
White-box

DeepInversion @CVPR'20

Target Inversion
White-box

Inversion results in the top are from their original papers or the official GitHub repository.

Existing Methods

Figure 1,10 MIA @CCS'15 — Figure 14 AMI @CCS'19 — Figure 2 GMI @CVPR'20 — DeepInversion @CVPR'20

Target Inversion
White-box

Target Inversion
Black-box

Target Inversion
White-box

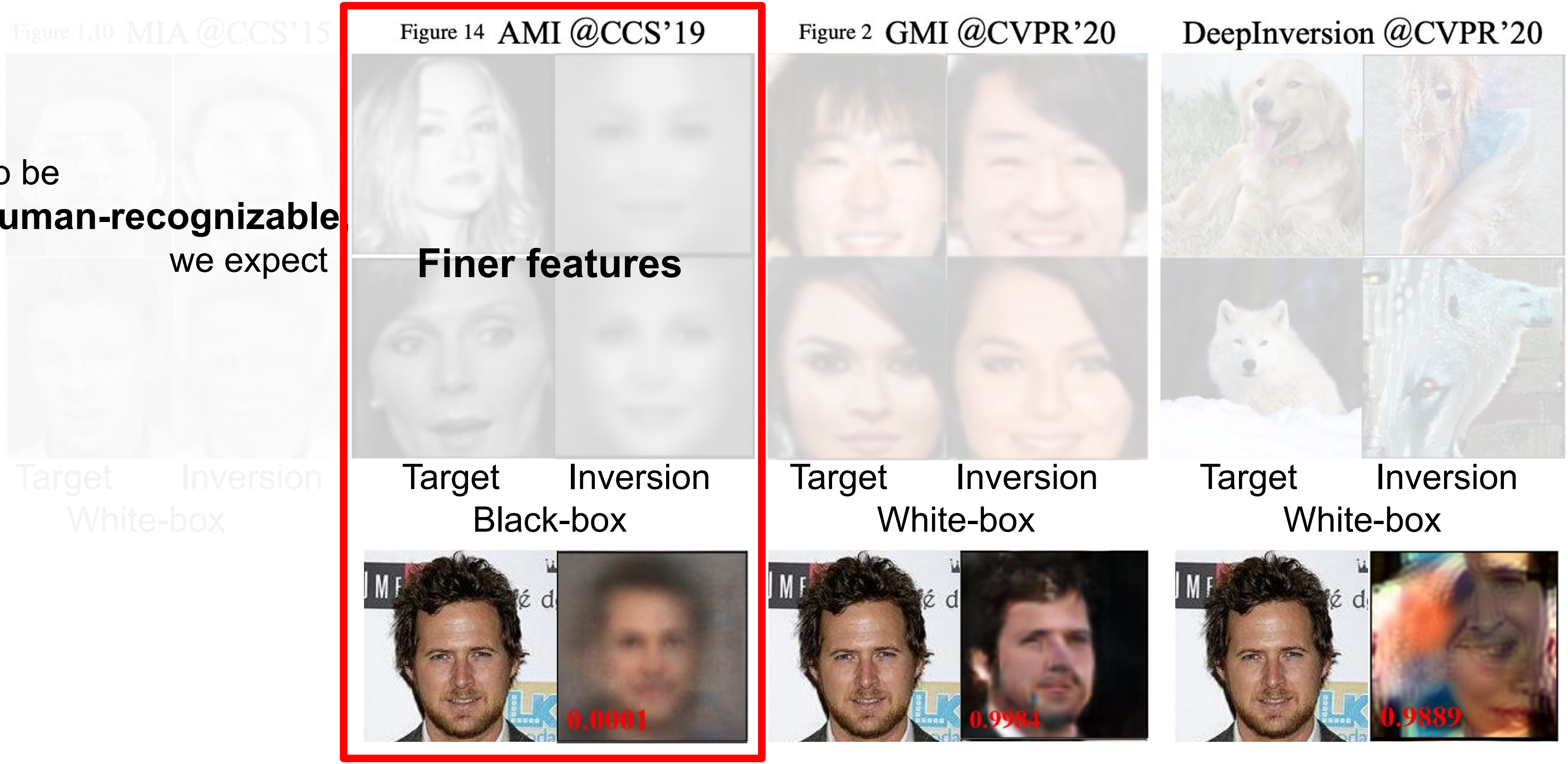
Target Inversion
White-box

Not very
human-recognizable

Inversion results in the top are from their original papers or the official GitHub repository.

Existing Methods

To be **human-recognizable**, we expect



Inversion results in the top are from their original papers or the official GitHub repository.

Existing Methods

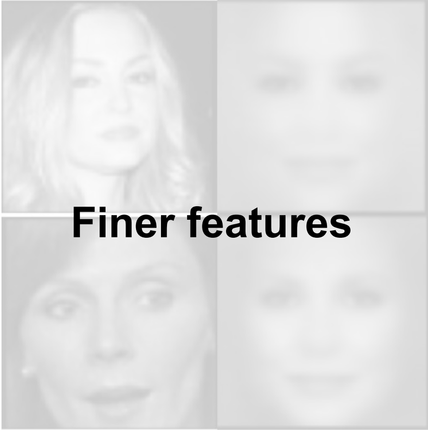
To be **human-recognizable,** we expect

Figure 1,10 MIA @CCS'15

Target Inversion

White-box

Figure 14 AMI @CCS'19



Finer features

Target Inversion

Black-box

Figure 2 GMI @CVPR'20

Higher resolutions

Target Inversion

White-box

DeepInversion @CVPR'20

Target Inversion

White-box

Inversion results in the top are from their original papers or the official GitHub repository.

Existing Methods

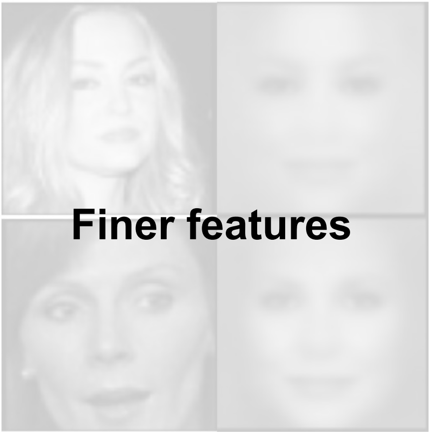
To be **human-recognizable,** we expect

Figure 1,10 MIA @CCS'15

Target Inversion

White-box

Figure 14 AMI @CCS'19



Finer features

Target Inversion

Black-box

Figure 2 GMI @CVPR'20

Higher resolutions

Target Inversion

White-box

DeepInversion @CVPR'20

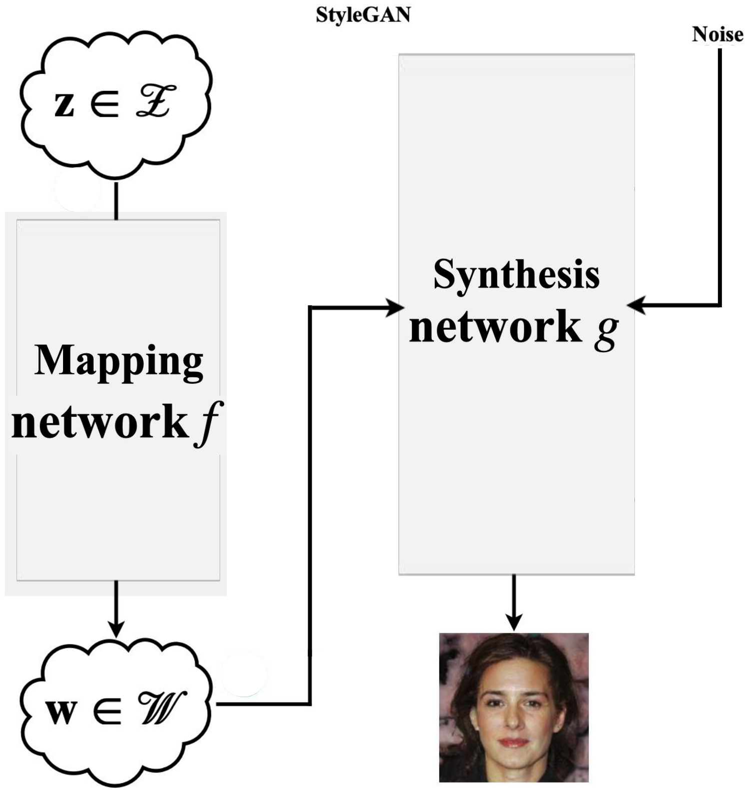
Features in the correct positions

Target Inversion

White-box

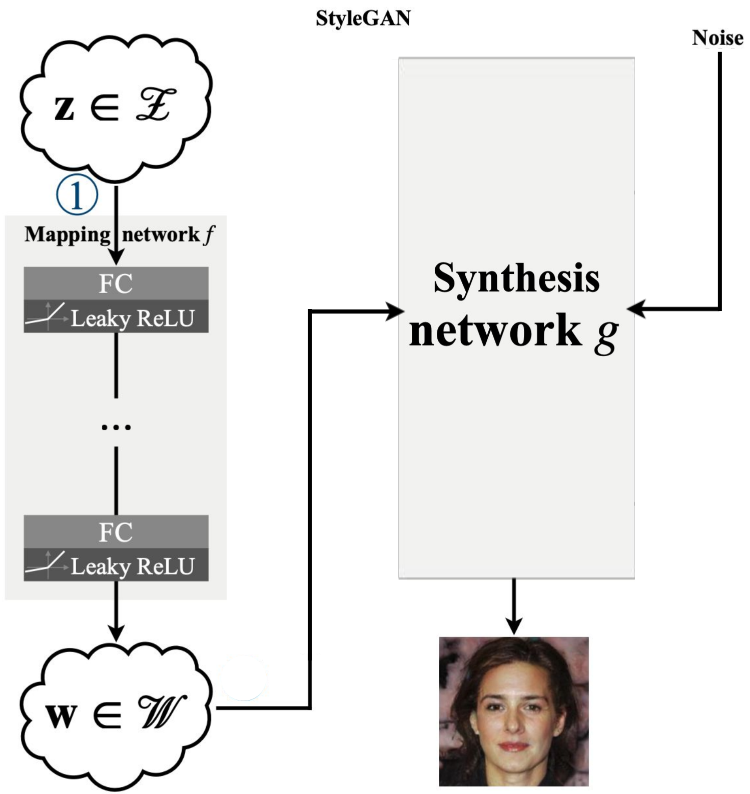
Inversion results in the top are from their original papers or the official GitHub repository.

Background: StyleGAN



Two main components: mapping and synthesis networks.

Background: StyleGAN



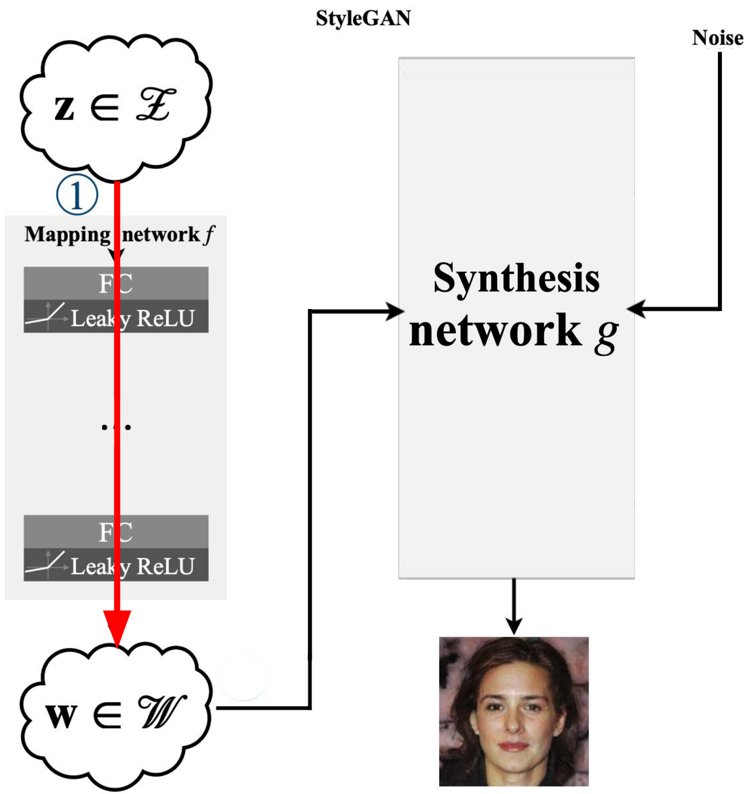
Two main components: mapping and synthesis networks.

Step1:

sample z from Gaussian distribution

generate w by f(z)

Background: StyleGAN



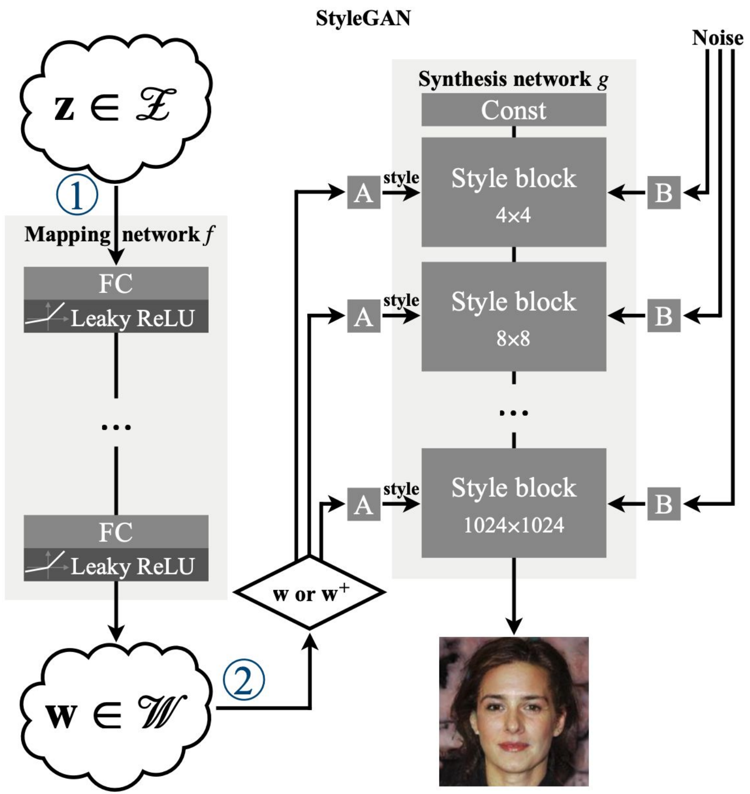
Two main components: mapping and synthesis networks.

Step1:

sample z from Gaussian distribution

generate w by f(z)

Background: StyleGAN



Two main components: mapping and synthesis networks.

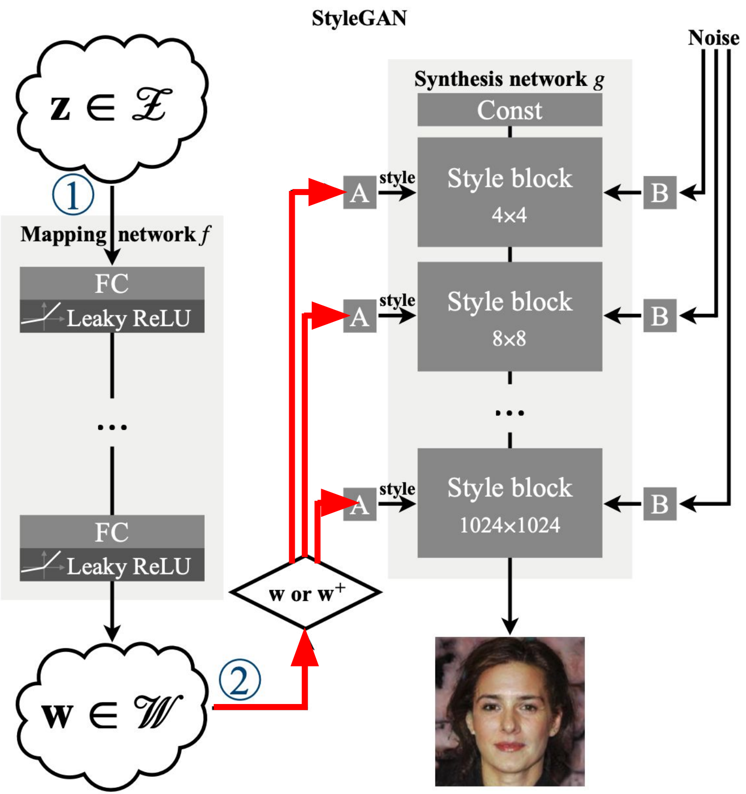
Step1:

sample z from Gaussian distribution
generate w by f(z)

Step 2:

w is duplicated and fed to each style block
w is transformed into styles (means and stds)
generate image g(f(z))

Background: StyleGAN



Two main components: mapping and synthesis networks.

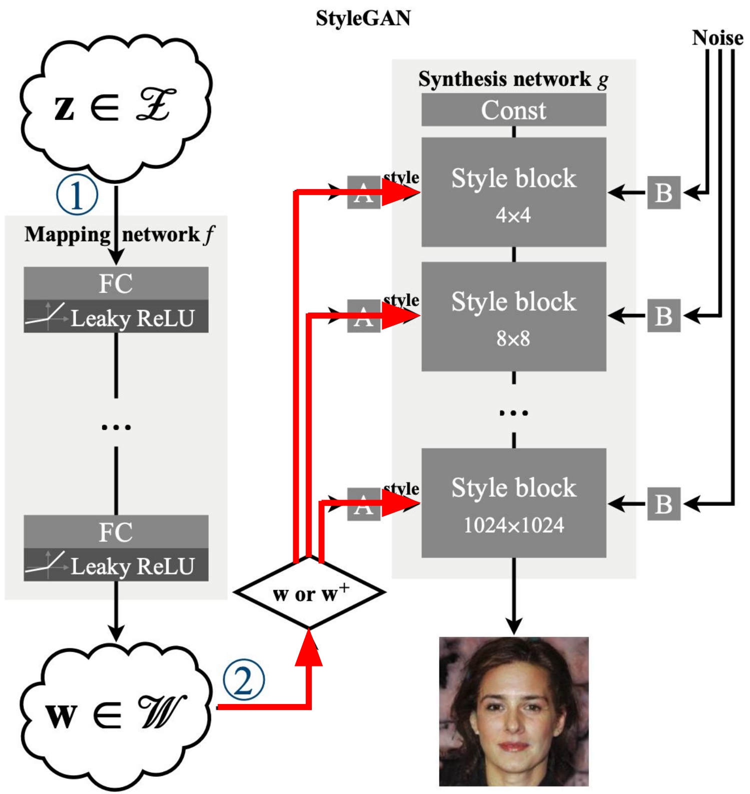
Step1:

sample z from Gaussian distribution
generate w by f(z)

Step 2:

w is duplicated and fed to each style block
w is transformed into styles (means and stds)
generate image g(f(z))

Background: StyleGAN



Two main components: mapping and synthesis networks.

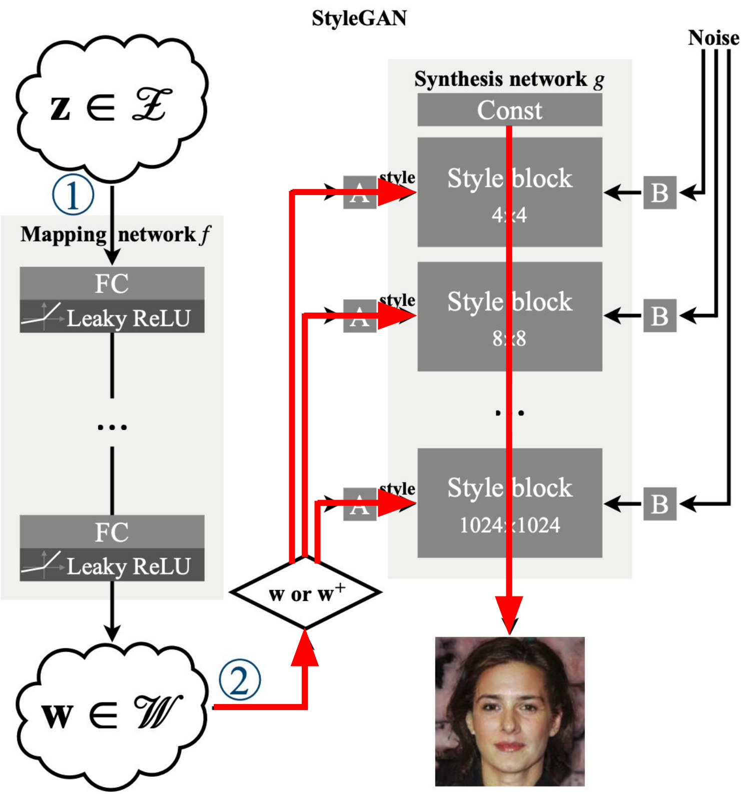
Step1:

- sample z from Gaussian distribution
- generate w by f(z)

Step 2:

- w is duplicated and fed to each style block
- w is transformed into styles (means and stds)
- generate image g(f(z))

Background: StyleGAN



Two main components: mapping and synthesis networks.

Step1:

sample z from Gaussian distribution

generate w by f(z)

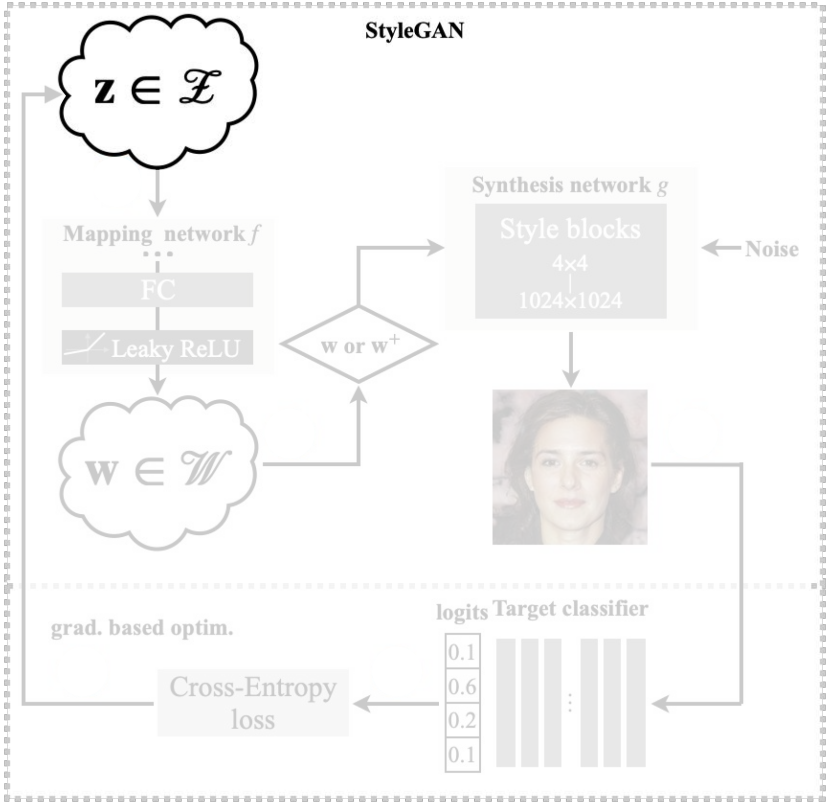
Step 2:

w is duplicated and fed to each style block

w is transformed into styles (means and stds)

generate image g(f(z))

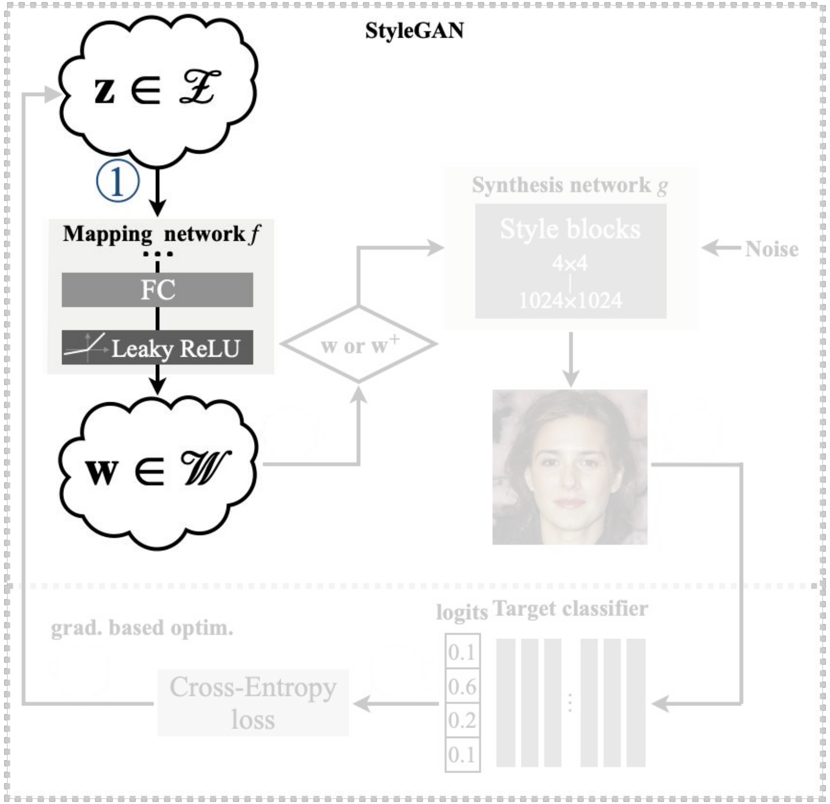
Design MIRROR (White-box) in Z space



Initialization:

Sample an initial z from Gaussian distribution

Design MIRROR (White-box) in Z space



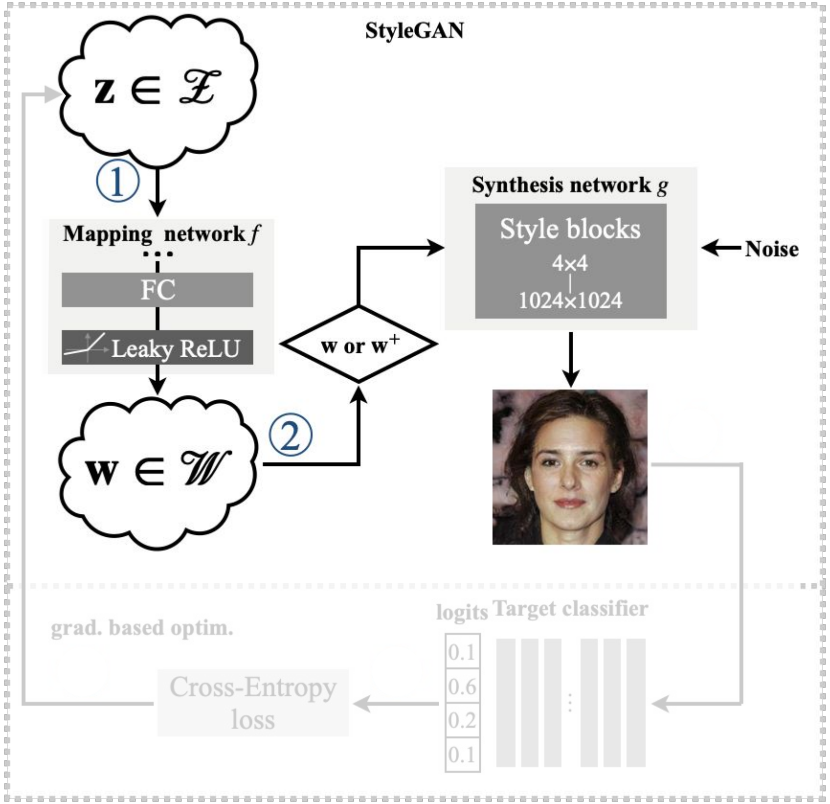
Initialization:

Sample an initial z from Gaussian distribution

Step 1:

Generate w by f(z)

Design MIRROR (White-box) in Z space



Initialization:

Sample an initial z from Gaussian distribution

Step 1:

Generate w by f(z)

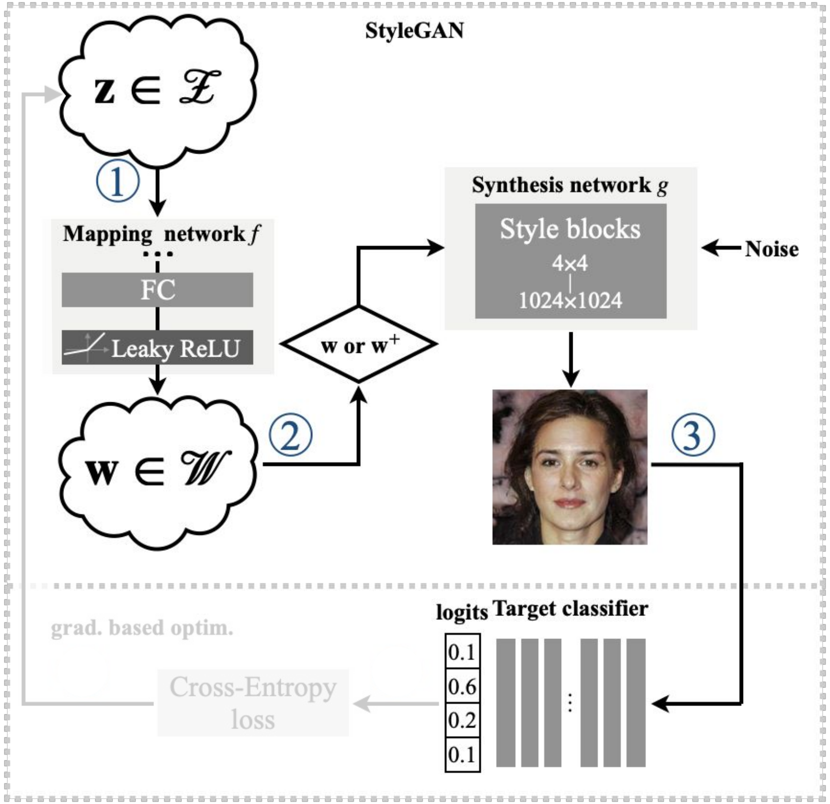
Step 2:

w is duplicated and fed to each style block

w is transformed into styles (means and stds)

Generate image g(f(z))

Design MIRROR (White-box) in Z space



Initialization:

Sample an initial z from Gaussian distribution

Step 1:

Generate w by f(z)

Step 2:

w is duplicated and fed to each style block

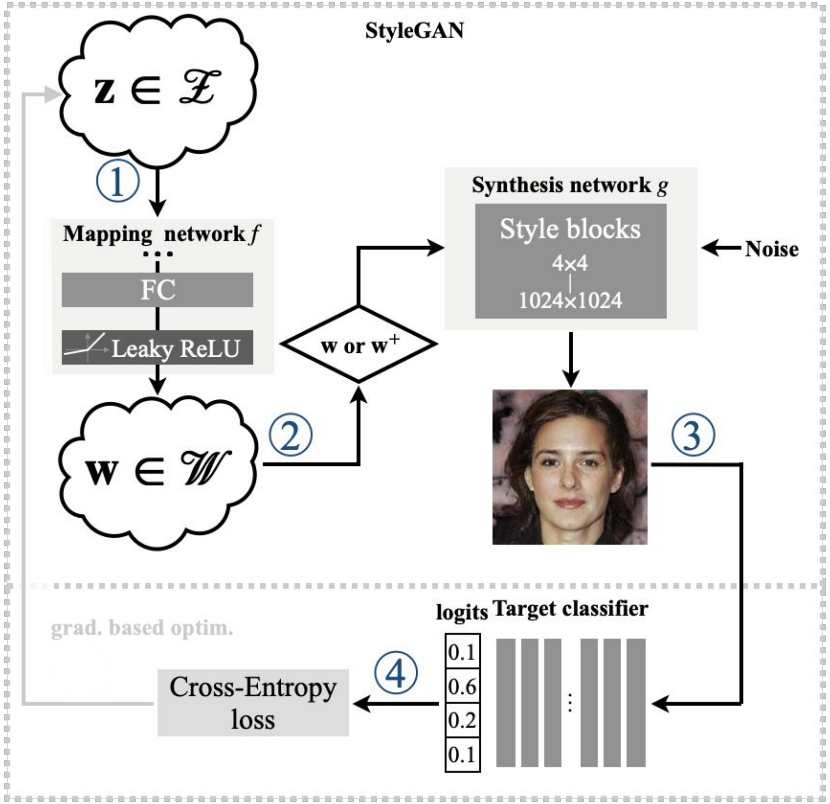
w is transformed into styles (means and stds)

Generate image g(f(z))

Step 3:

Feed g(f(z)) to the subject model M

Design MIRROR (White-box) in Z space



Initialization:

Sample an initial z from Gaussian distribution

Step 1:

Generate w by f(z)

Step 2:

w is duplicated and fed to each style block

w is transformed into styles (means and stds)

Generate image g(f(z))

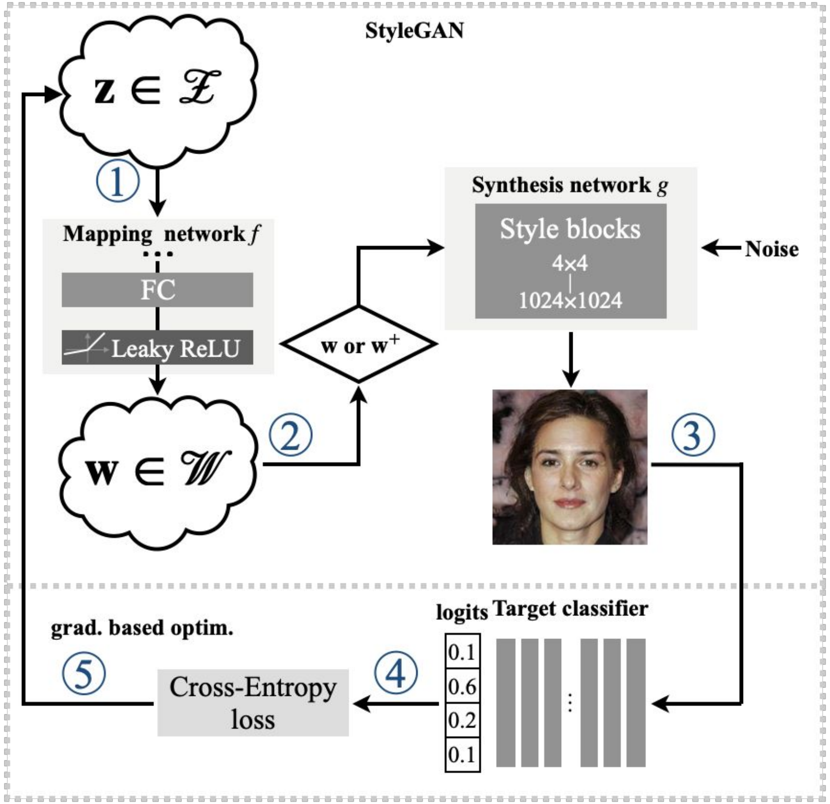
Step 3:

Feed g(f(z)) to the subject model M

Step 4:

Compute the classification loss

Design MIRROR (White-box) in Z space



Initialization:

Sample an initial z from Gaussian distribution

Step 1:

Generate w by f(z)

Step 2:

w is duplicated and fed to each style block

w is transformed into styles (means and stds)

Generate image g(f(z))

Step 3:

Feed g(f(z)) to the subject model M

Step 4:

Compute the classification loss

Step 5:

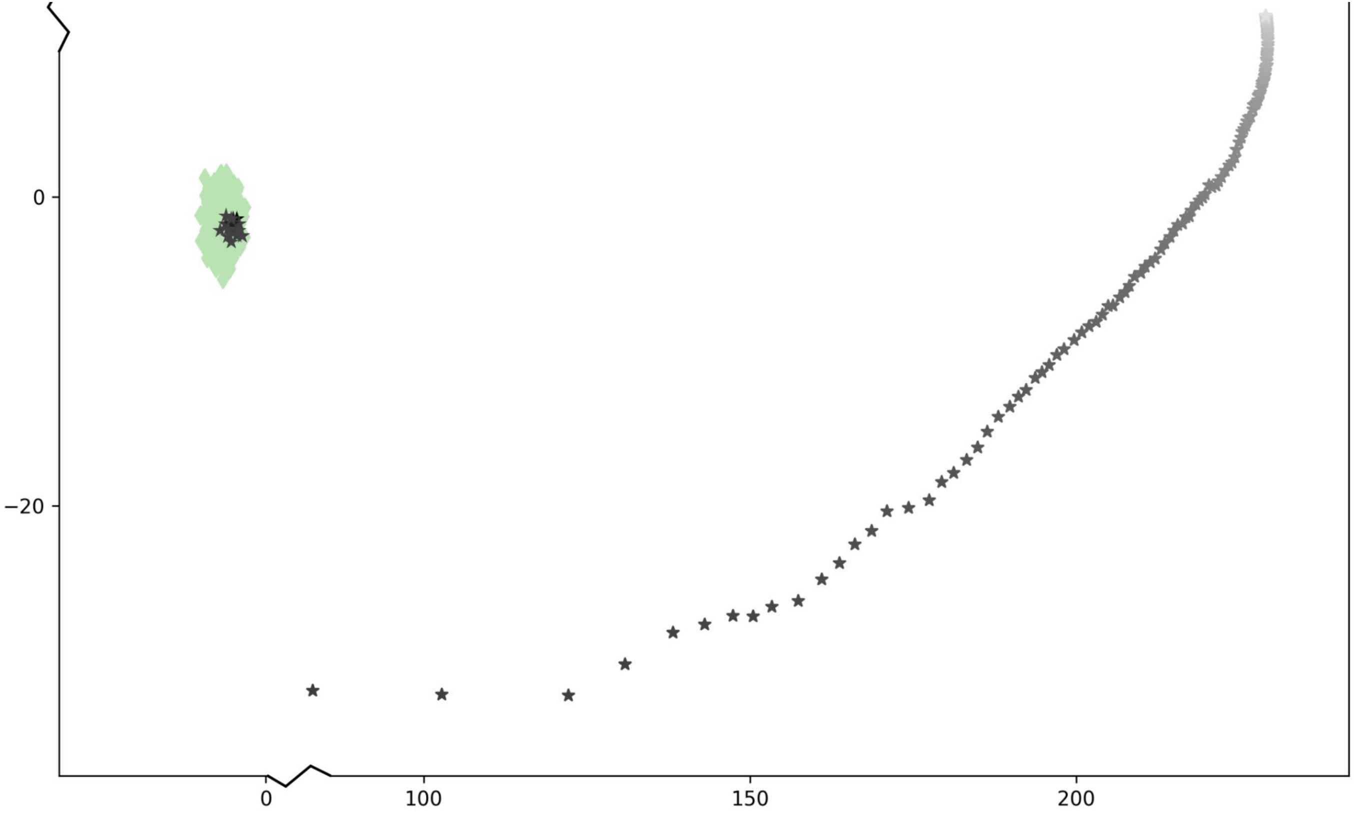
Use the gradient-descent method to update z

Repeat Step 1-5

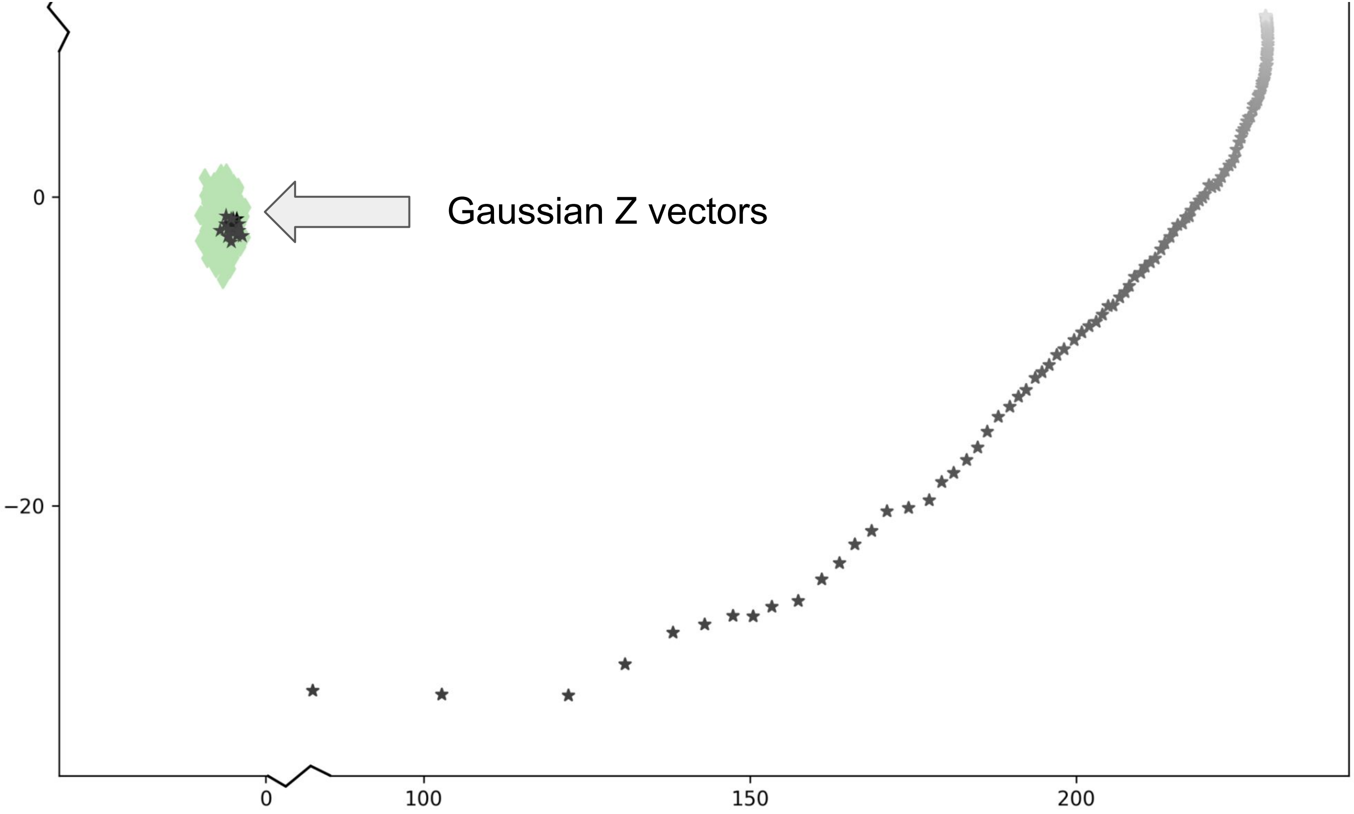
Optimization in the Z space is ineffective

Target person

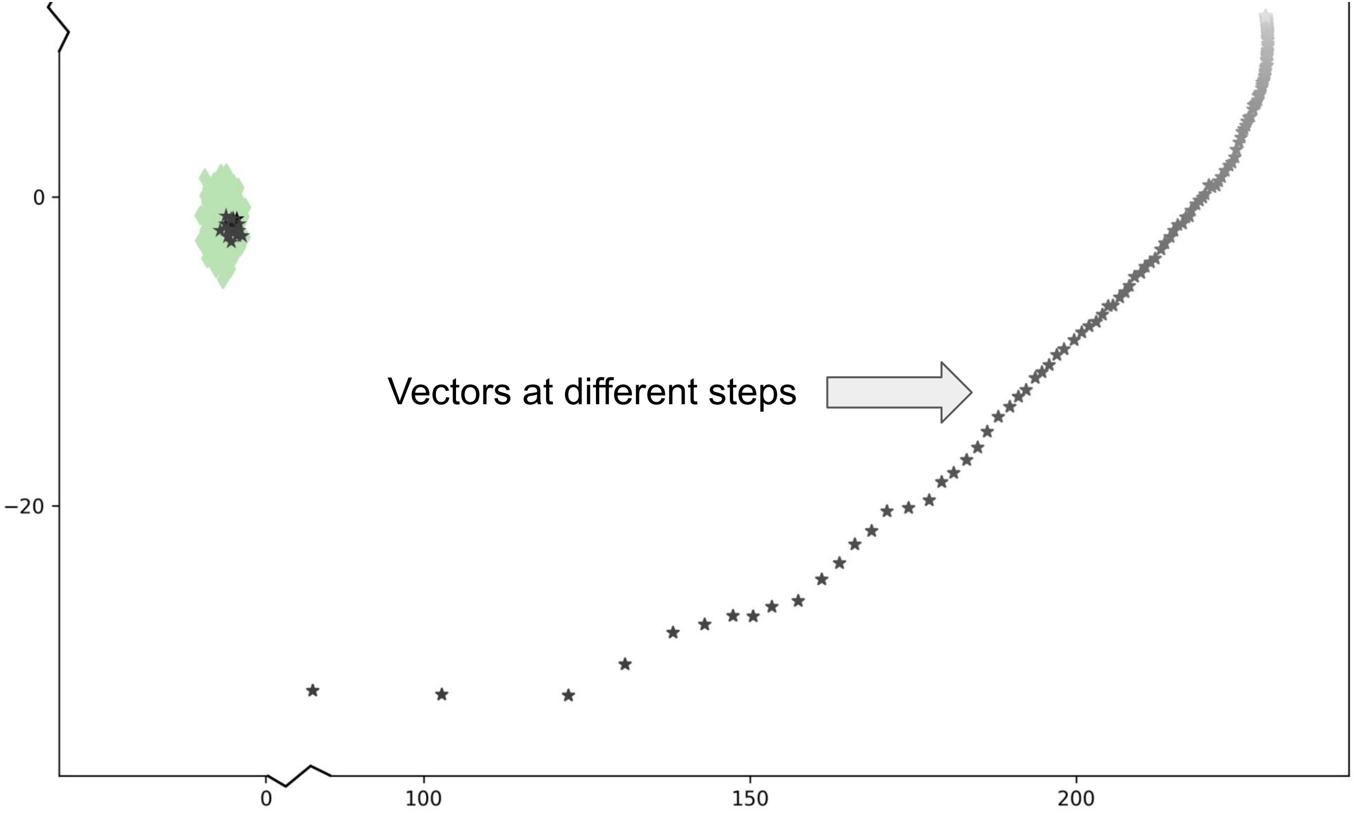

Optimization in the Z space is ineffective



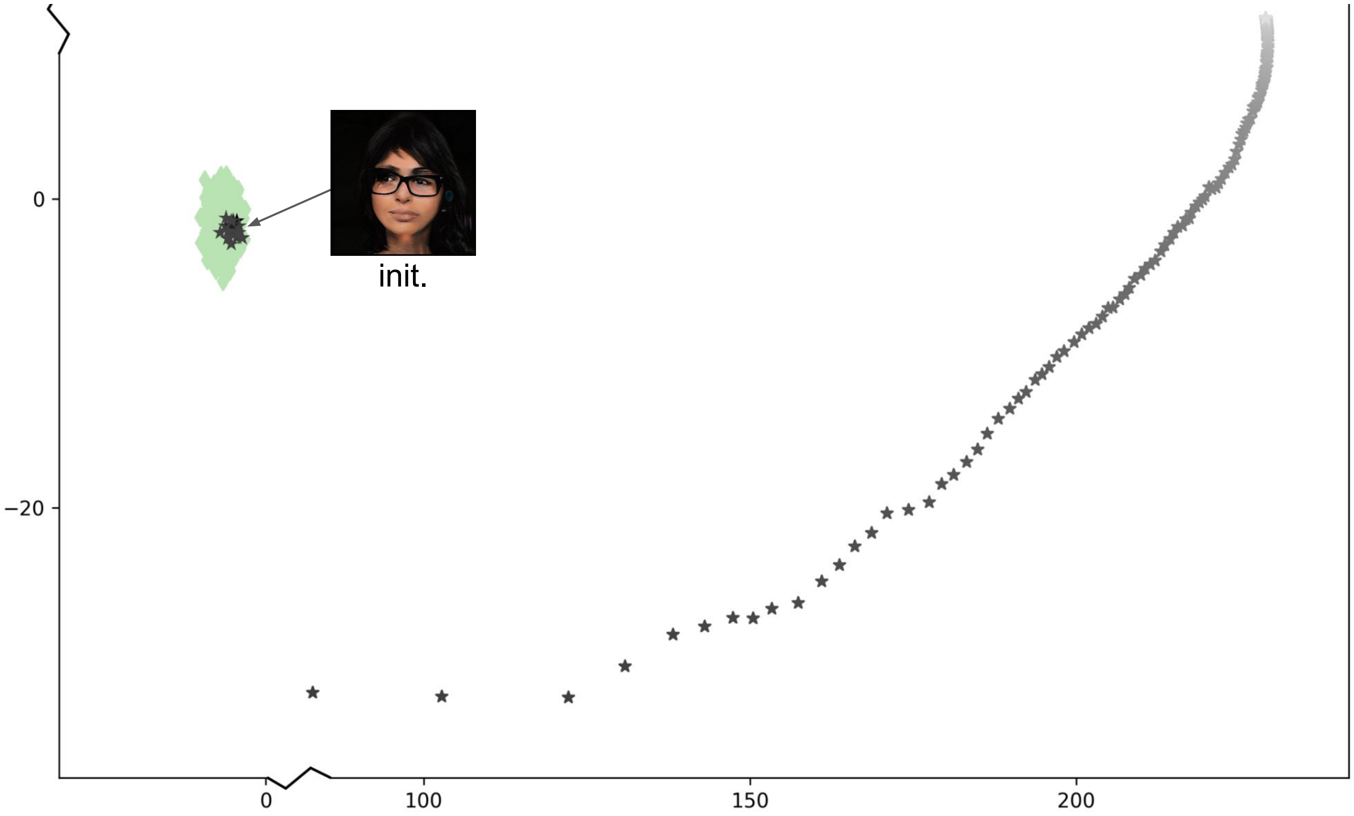
Optimization in the Z space is ineffective



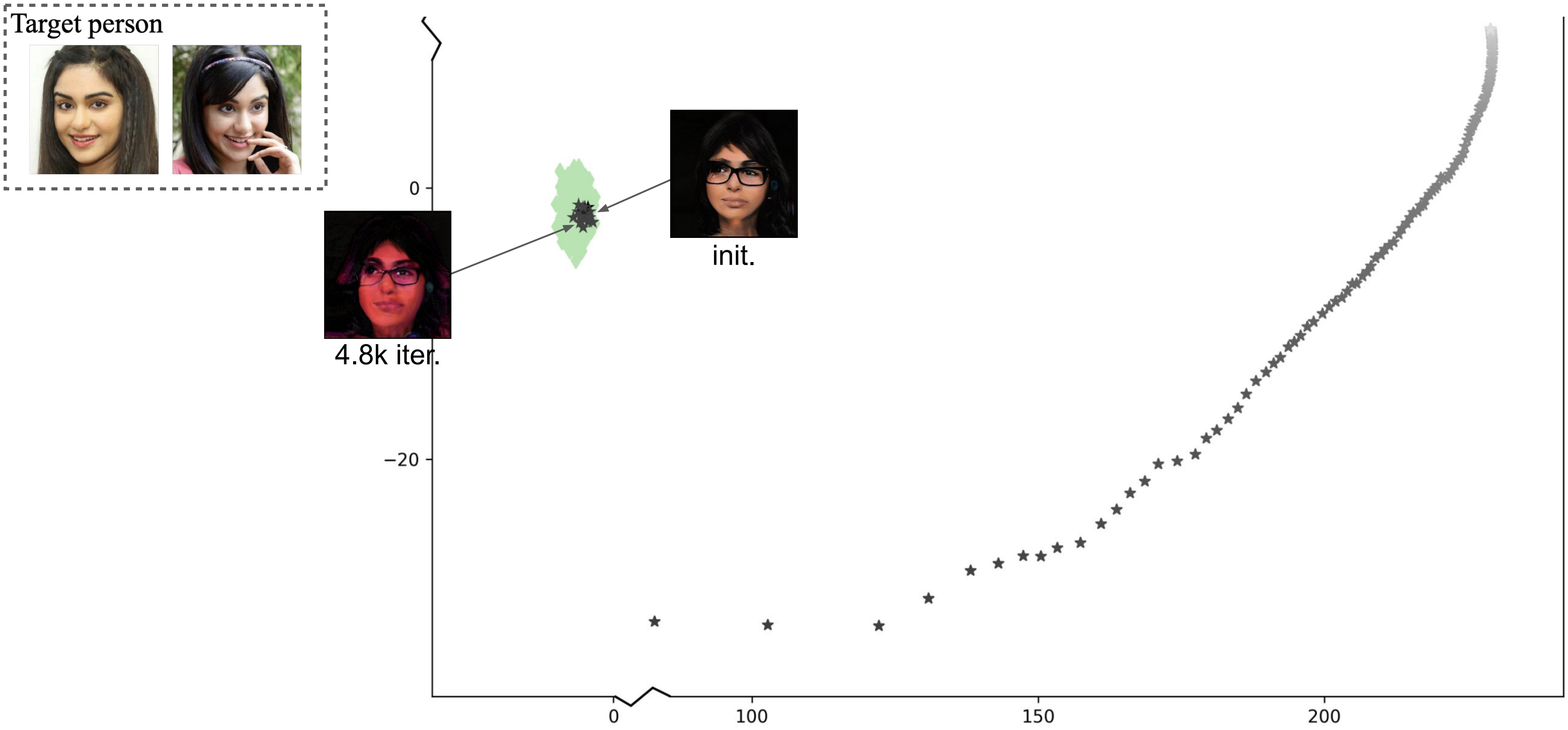
Optimization in the Z space is ineffective



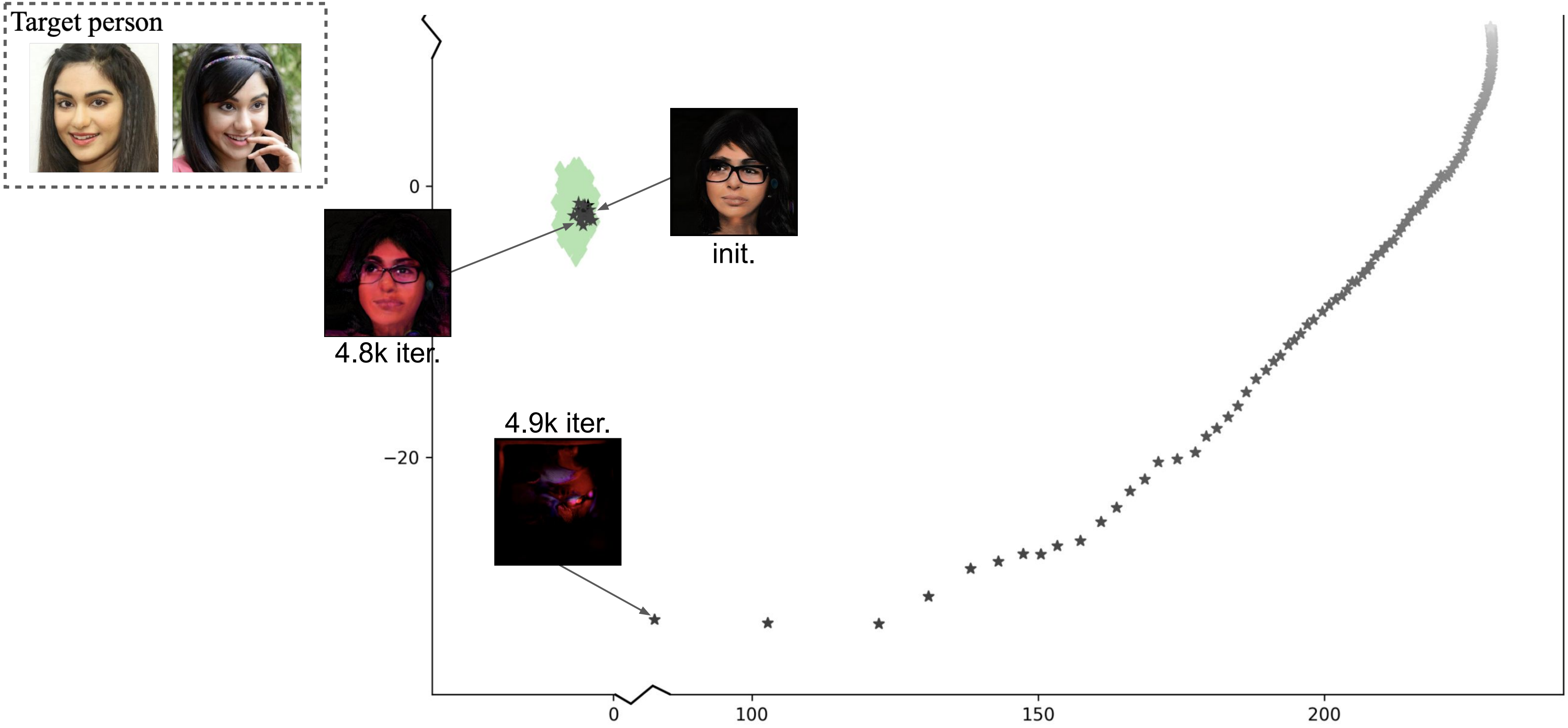
Optimization in the Z space is ineffective



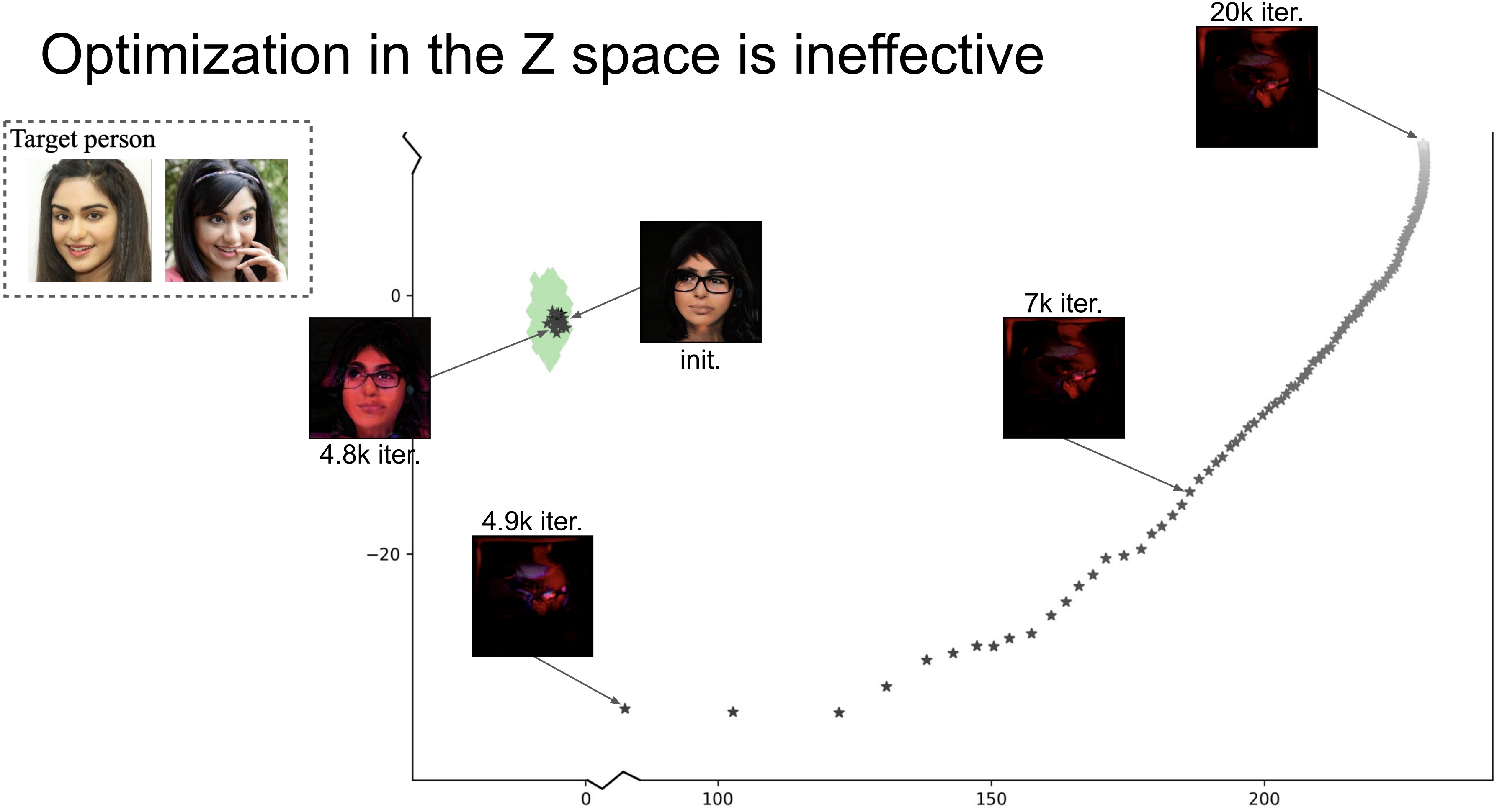
Optimization in the Z space is ineffective



Optimization in the Z space is ineffective

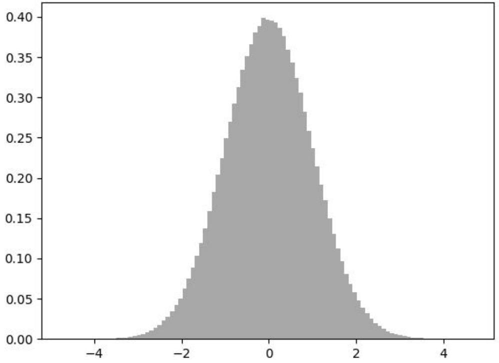


Optimization in the Z space is ineffective



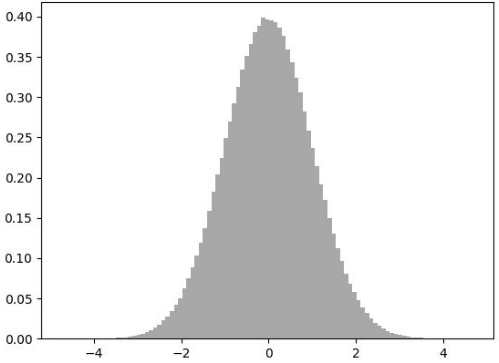
Optimization in the Z space is ineffective (even with clipping)

Target person

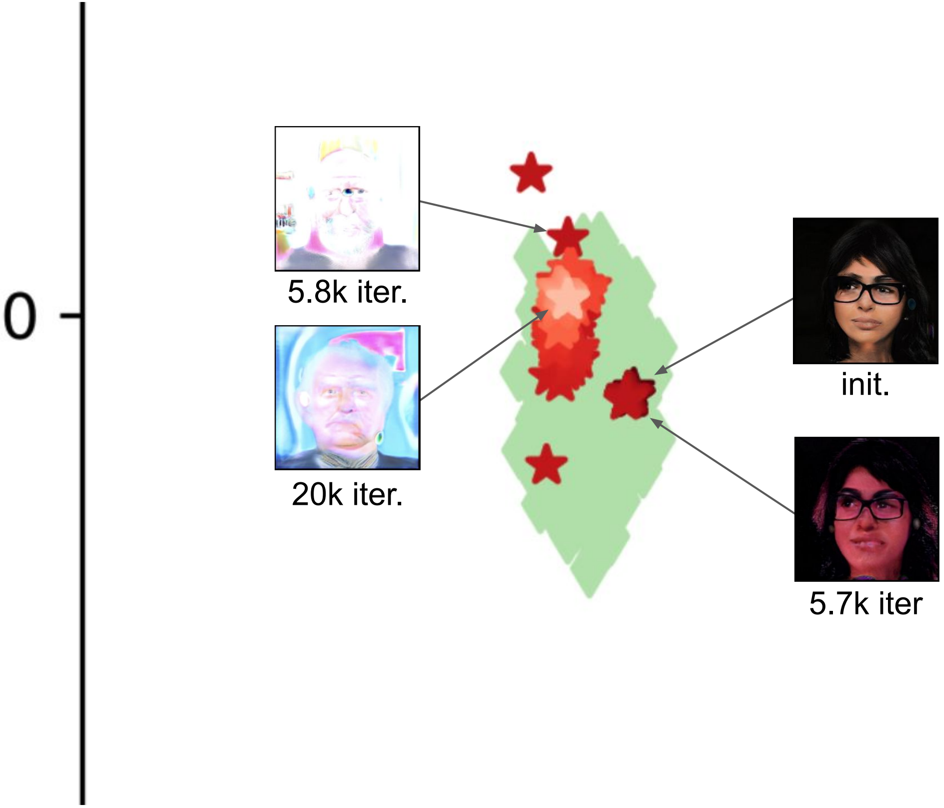


Z clipping in to [mean-std, mean+std]

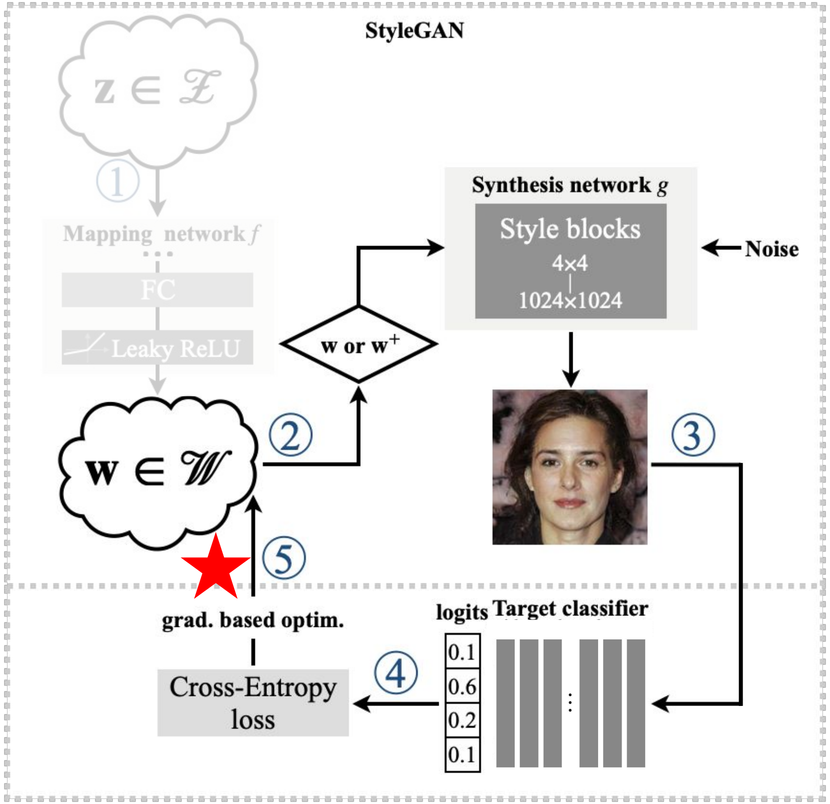
Optimization in the Z space is ineffective (even with clipping)



Z clipping in to [mean-std, mean+std]



Design MIRROR (White-box) in W space



Initialization:

Sample an initial z from Gaussian distribution

(Step 1) Generate the initial w by f(z)

Step 2:

w is fed to each style block

w is transformed into styles (means and stds)

Generate image g(w)

Step 3:

Feed g(w) to the subject model M

Step 4:

Compute the classification loss

Step 5:

Use the gradient-descent method to update w

Repeat Step 2-5

Simple clipping in W space doesn't work

Target person

init.

20k iter.

Without clipping

With simple w clipping

Simple clipping in W space doesn't work

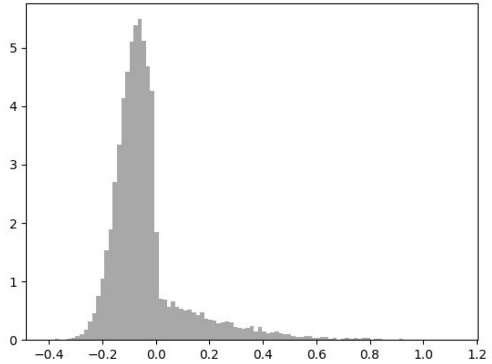
Target person

init.

20k iter.

Without clipping

With simple w clipping



Different from Z space, W space is not normal.

Simple clipping in W space doesn't work

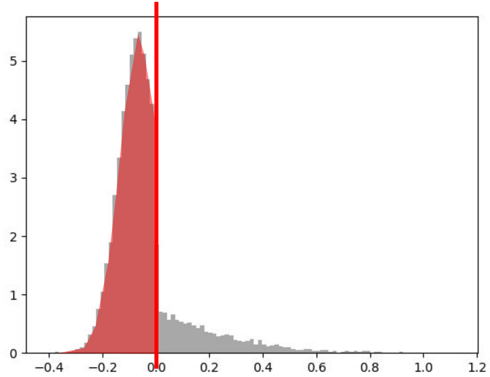
Target person

init.

20k iter.

Without clipping

With simple w clipping



Lots of *negative* values are close to 0.

Simple clipping in W space doesn't work

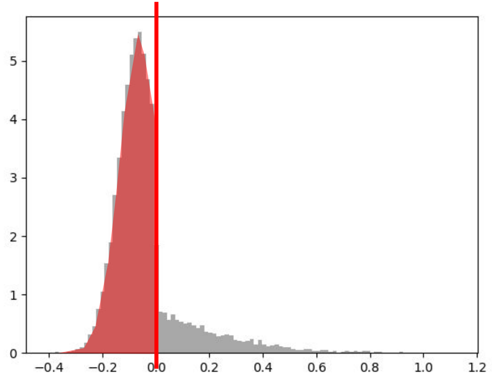
Target person

init.

20k iter.

Without clipping

With simple w clipping



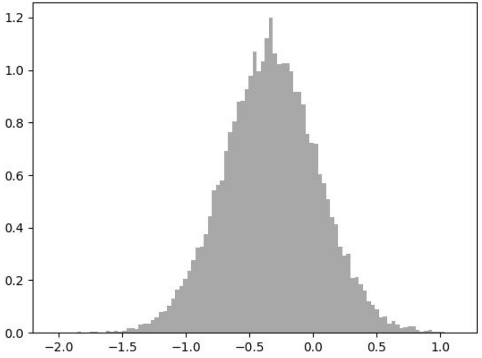
Lots of *negative* values are close to 0.

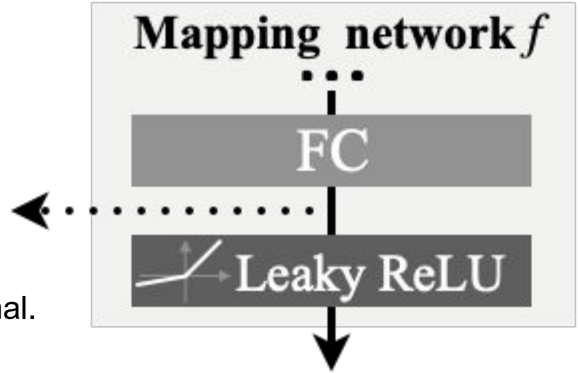
x 0.2

Simple clipping in W space doesn't work

Without clipping

With simple w clipping





The P space before W space is normal.

W space

Use clipping in P space

Target person

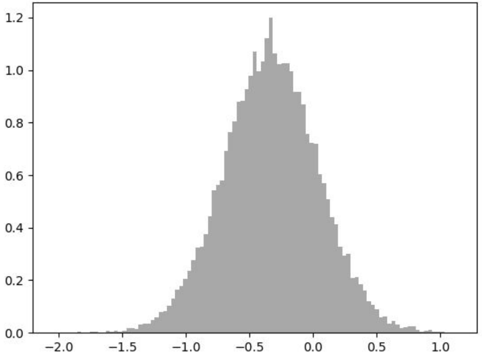
init.

20k iter.

Without clipping

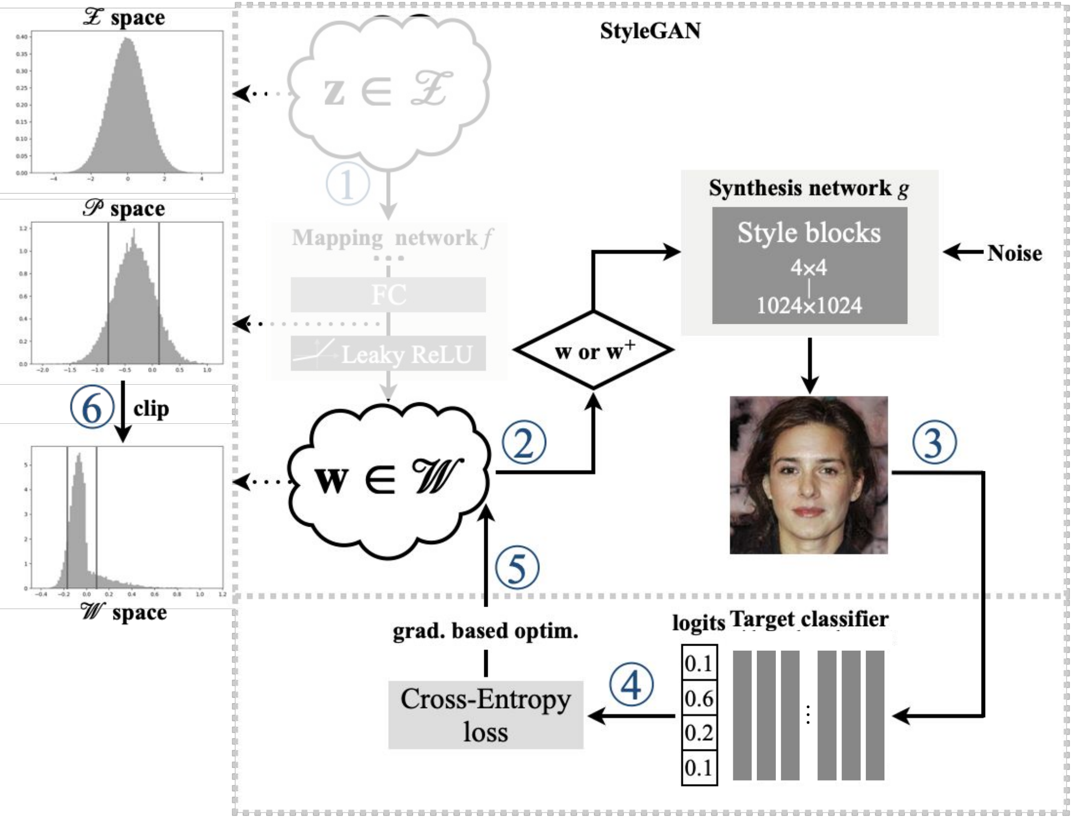
With simple w clipping

With p clipping



P clipping in to [mean-std, mean+std]

Design MIRROR (White-box) W Space & P Clipping



Initialization:

Sample an initial z from Gaussian distribution

(Step 1) Generate the initial w by f(z)

Step 2:

w is fed to each style block

w is transformed into styles (means and stds)

Generate image g(w)

Step 3:

Feed g(w) to the subject model M

Step 4:

Compute the classification loss

Step 5:

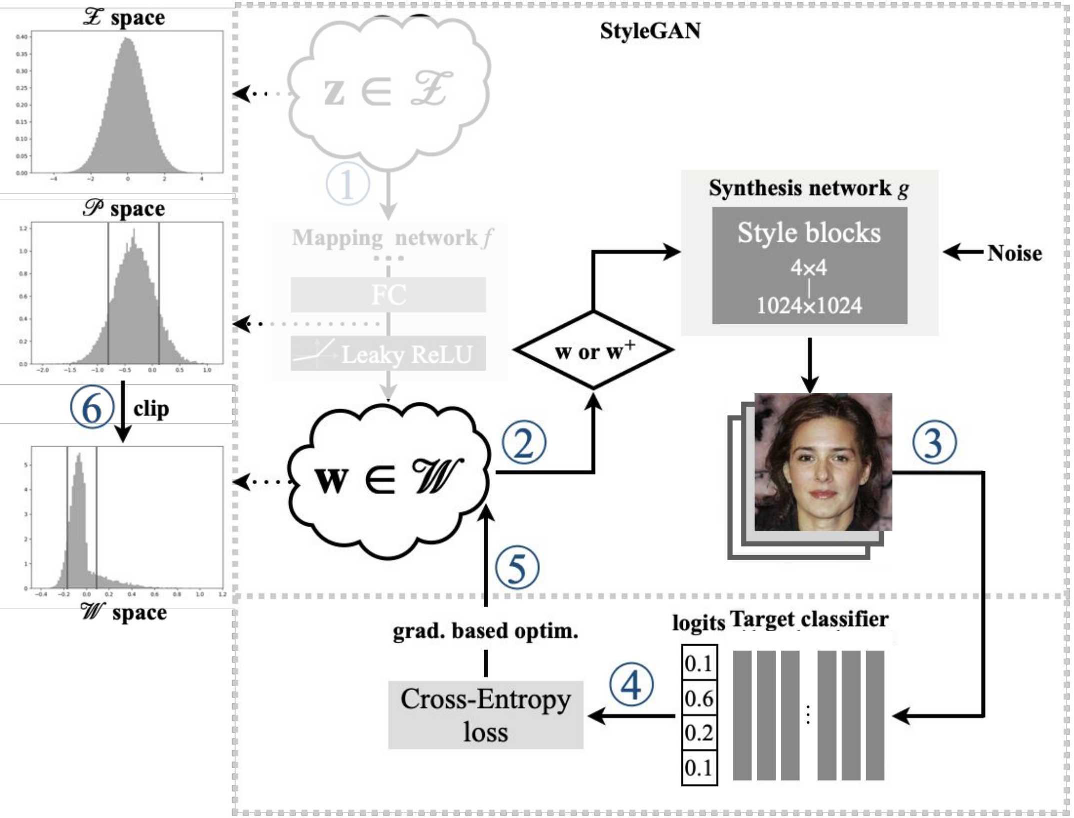
Use the gradient-descent method to update w

Step 6:

Clip w in P space

Repeat Step 2-6

Design MIRROR (White-box) W Space & P Clipping



Initialization:

Sample **a batch of zs**

(Step 1) Generate **a batch of ws** by f(zs)

Step 2:

ws is fed to each style block

ws is transformed into styles (means and stds)

Generate **a batch of image** g(ws)

Step 3:

Feed g(ws) to the subject model M

Step 4:

Compute the classification loss

Step 5:

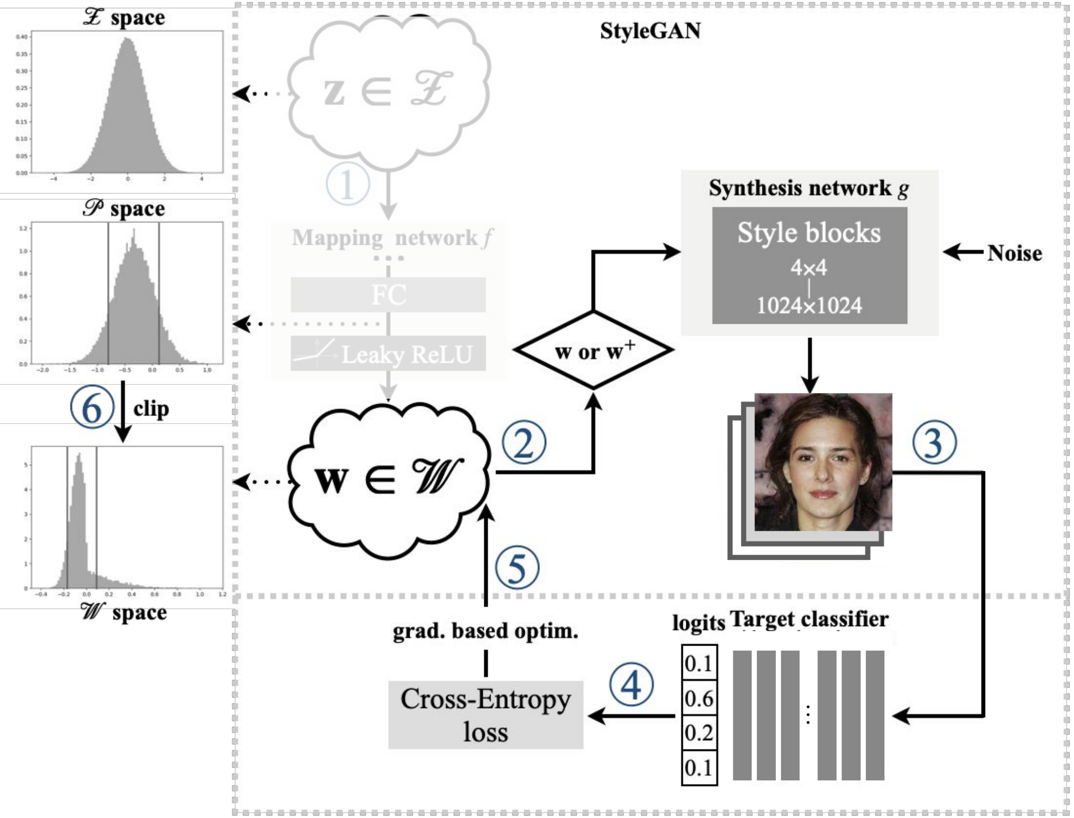
Use the gradient-descent method to update ws

Step 6:

Clip ws in P space

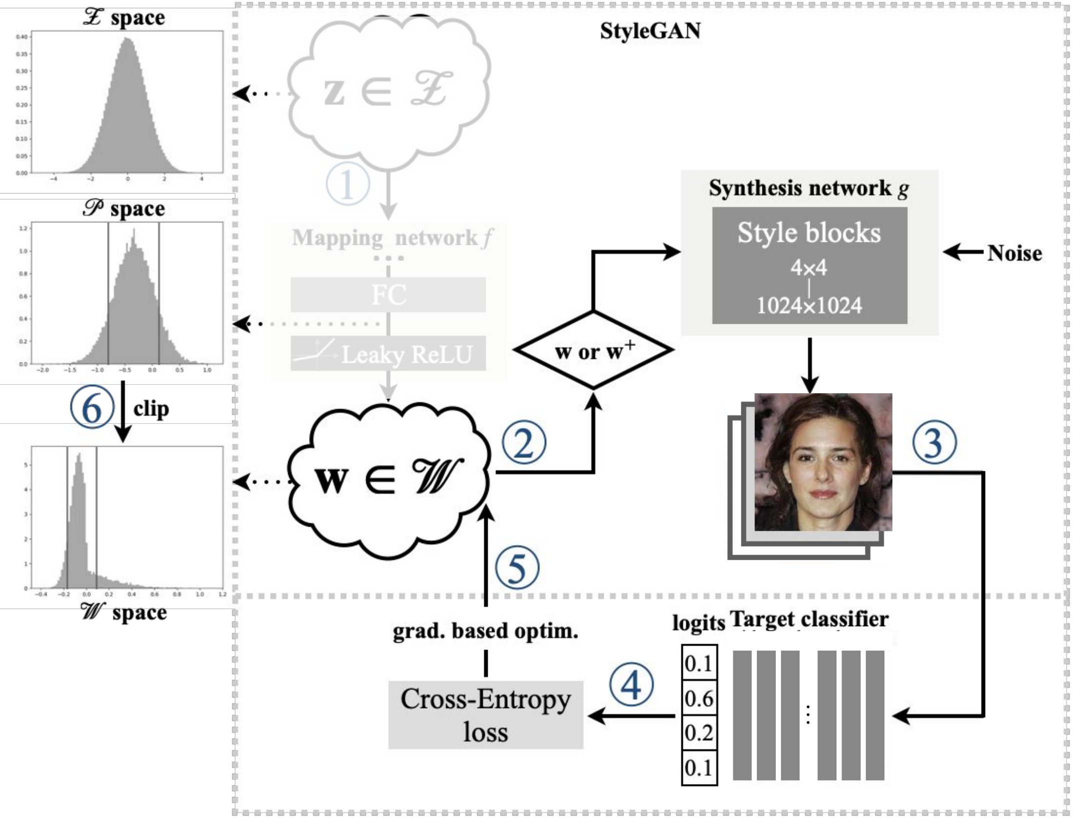
Repeat Step 2-6

Design MIRROR (White-box) - Overfitting



Target

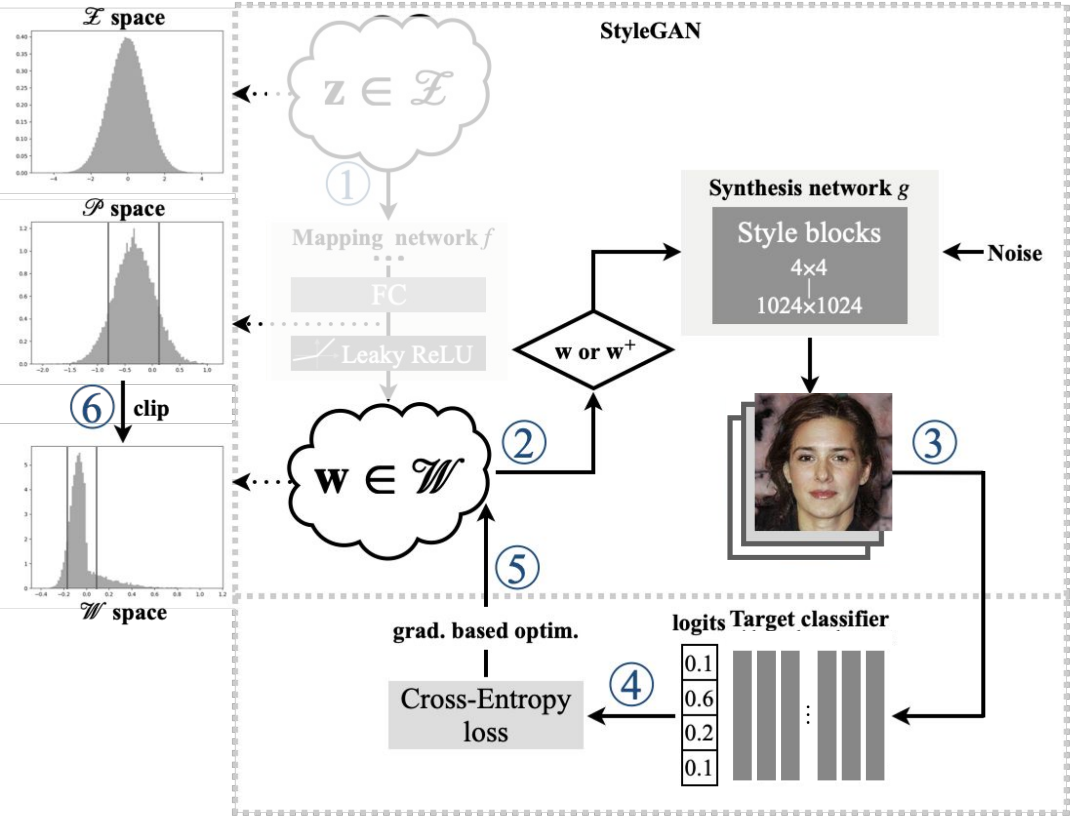
Design MIRROR (White-box) - Overfitting



Target

Inversion

Design MIRROR (White-box) - Overfitting

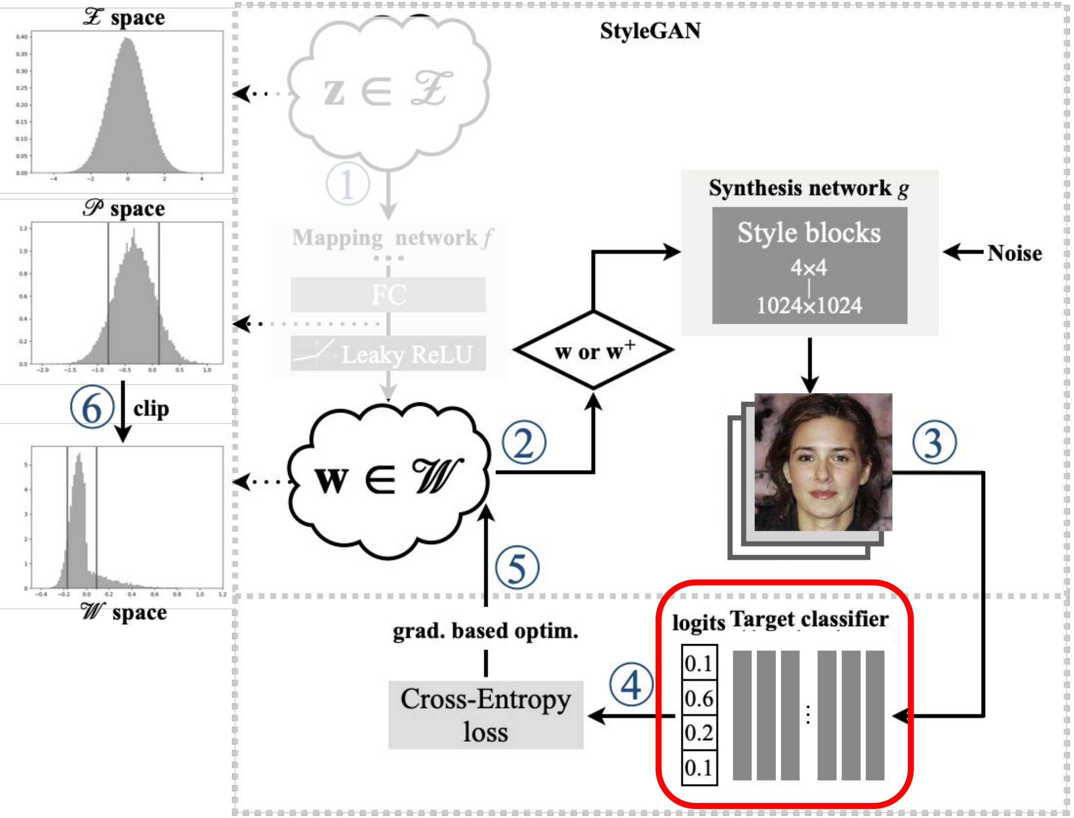


Target

Inversion

Issue: natural images with high confidences are not target person.

Design MIRROR (White-box) - Overfitting



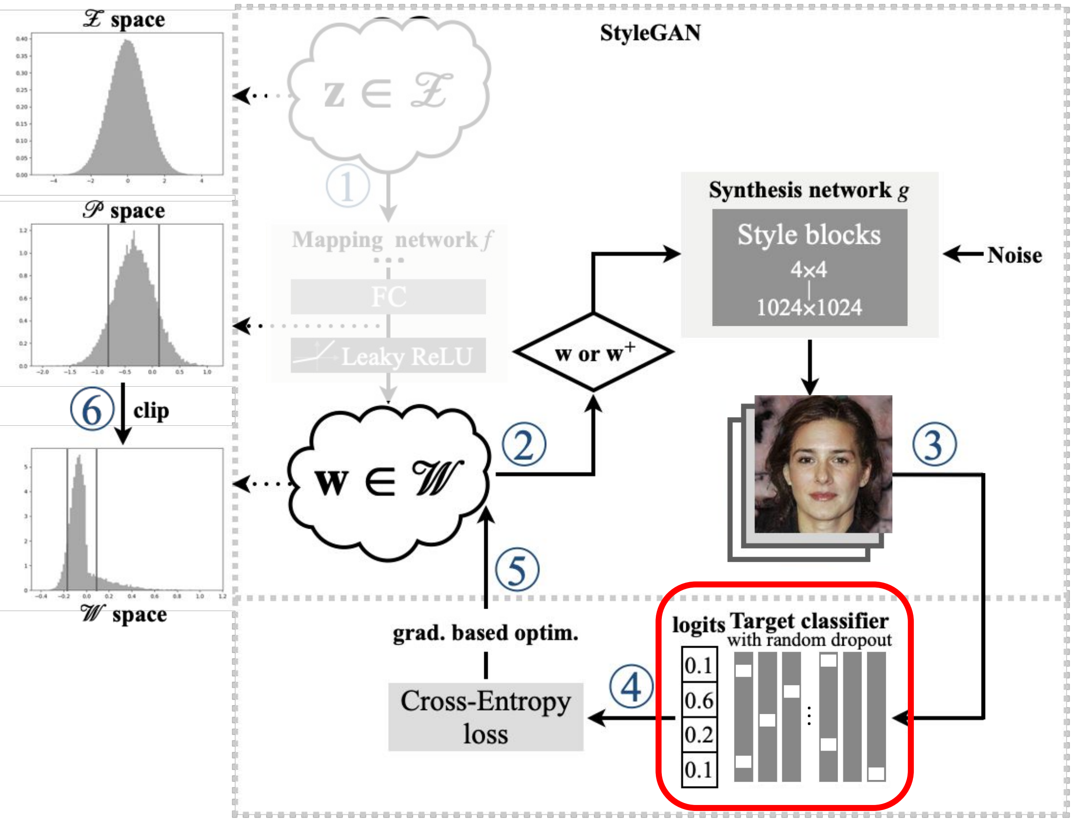
Target

Inversion

Issue: natural images with high confidences are not target person.

Cause: overfitting on low-level features leads to local optima.

Design MIRROR (White-box) - Random Dropout

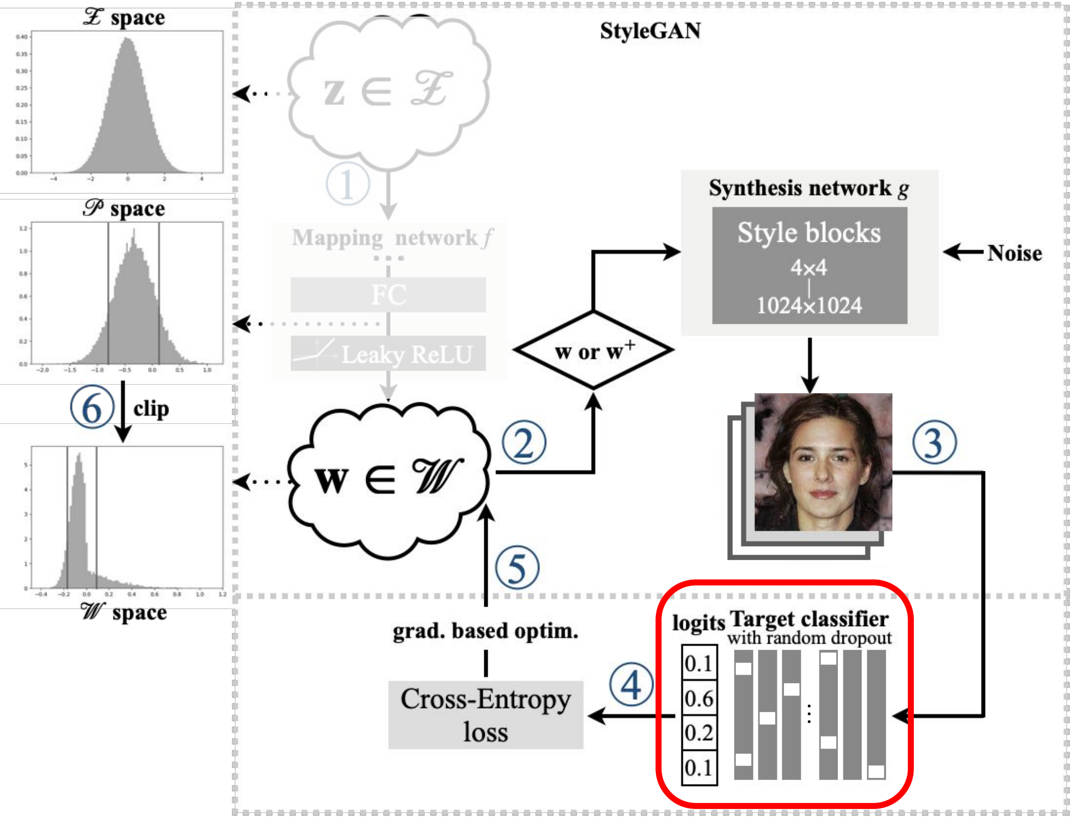


Target

Inversion

Solution: we randomly dropout neurons (set their activations to 0).

Design MIRROR (White-box) - Random Dropout

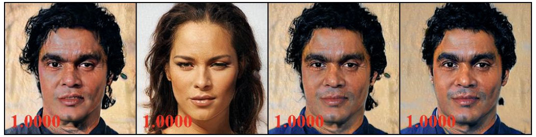


Target

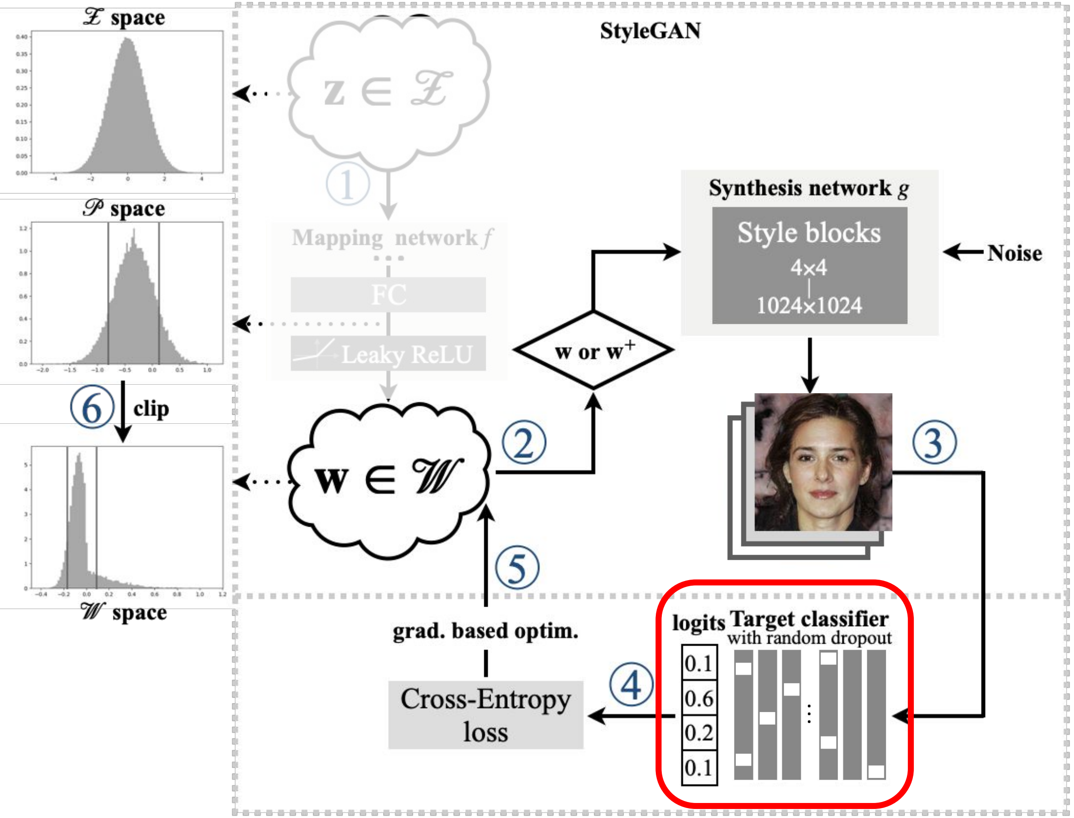
Inversion

Solution: we randomly dropout neurons (set their activations to 0).

Inversion



Design MIRROR (White-box) - Random Dropout

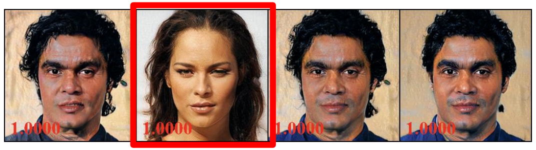


Target

Inversion

Solution: we randomly dropout neurons (set their activations to 0).

Inversion



Which one to return?
Highest confidence?

Design MIRROR (White-box) - Consistent Selection

Observation: wrong images label rankings are more diverse

Target

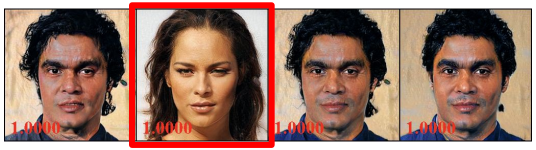
Inversion

Design MIRROR (White-box) - Consistent Selection

Observation: wrong images label rankings are more diverse
Strategy: select images with consistent label rankings

Target

Inversion



Top-5 labels	1	1	1	1
	2377	848	2377	2377
	17	1815	17	17
	2051	1806	2051	1570
	1570	853	1570	2241

Design MIRROR (White-box) - Consistent Selection

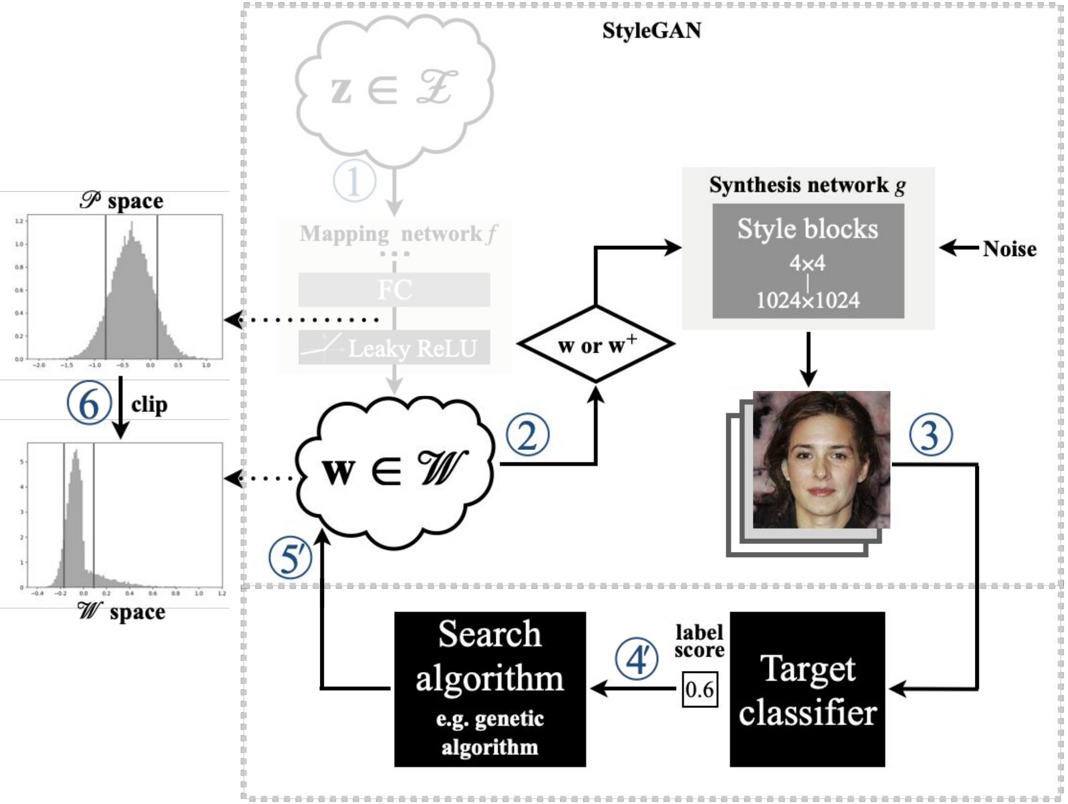
Observation: wrong images label rankings are more diverse
Strategy: select images with consistent label rankings

Target

Inversion

Top-5 labels				
	1	1	1	1
	2377	848	2377	2377
	17	1815	17	17
	2051	1806	2051	1570
	1570	853	1570	2241

Design MIRROR (Block-box)



Initialization:

Sample an initial z from Gaussian distribution

(Step 1) Generate the initial w by f(z)

Step 2:

w is fed to each style block

w is transformed into styles (means and stds)

Generate image g(w)

Step 3:

Feed g(w) to the subject model M

Step 4:

Compute the classification loss

Step 5:

Use the search algorithm to update w

Step 6:

Clip w in P space

Repeat Step 2-6

Evaluation - Datasets and Models

Dataset	VGGFace (2,622/2.6M)		VGGFace2 (9,131/3.3M)		CASIA (10,575/0.5M)	
Model	VGG16	VGG16BN	ResNet50	InceptionV1	InceptionV1	SphereFace
Accuracy	97.22%	96.29%	99.88%	99.65%	99.05%	99.22%
Input size	3x224x224	3x224x224	3x224x224	3x160x160	3x160x160	3x112x96

Non-overlapping Inversion: We only invert the labels which are not in the StyleGANs' training datasets.

Evaluation - Baselines and Metrics

Baselines in this slides: (please refer to our paper for more results)

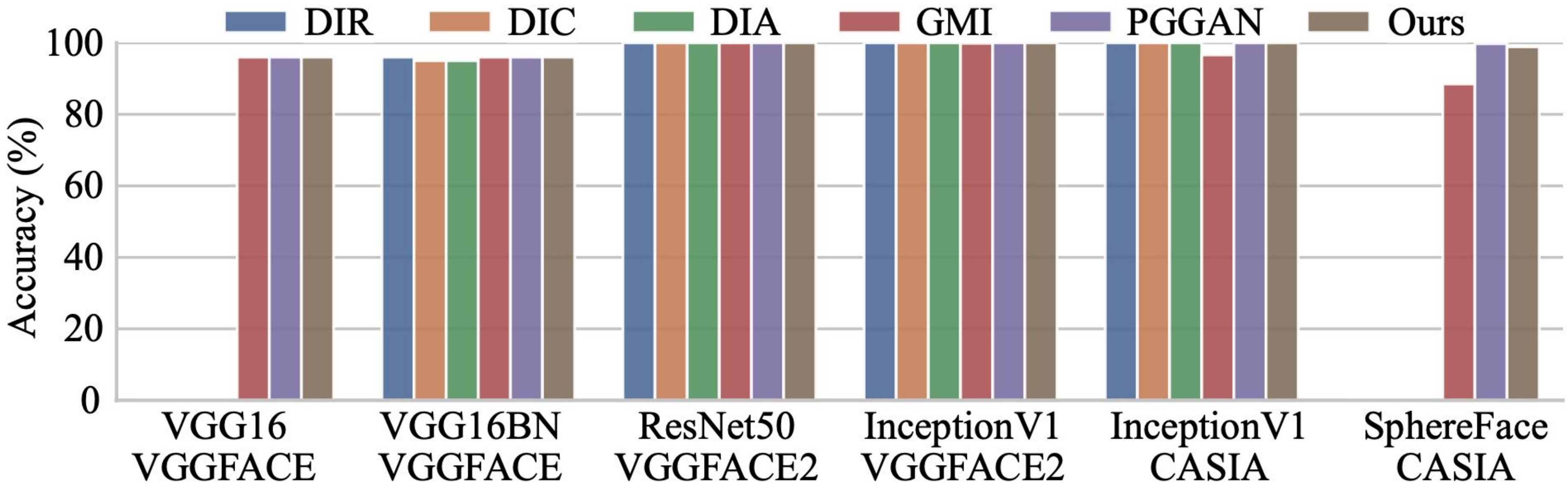
1. Existing AMI, GMI, DeepInversion.
   a. For AMI and GMI, use the same training dataset of the StyleGAN.
   b. For DeepInversion, we try different initializations.
      i. (DIR) Random noises
      ii. (DIA) Average faces
      iii. (DIC) Cartoon faces

2. Our proposed baselines: Use high-resolution PGGAN in GMI

Evaluation - White-box Inversion Qualitative Results

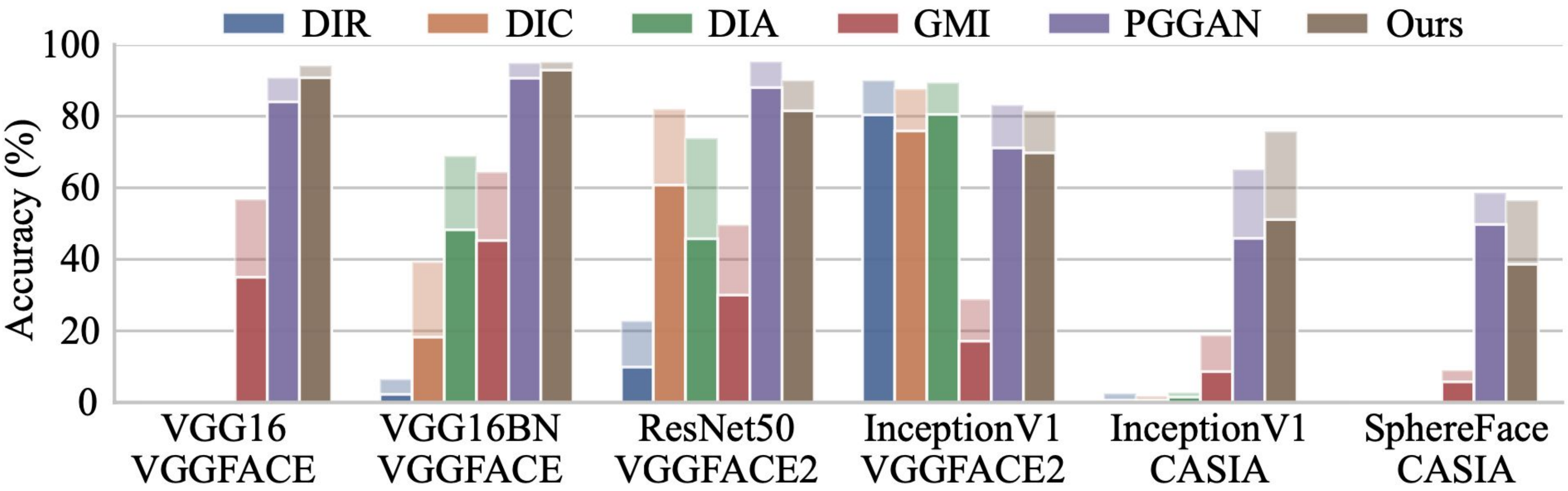
Evaluation - Whitebox Inversion Effectiveness

Can the subject model recognize the inverted images?



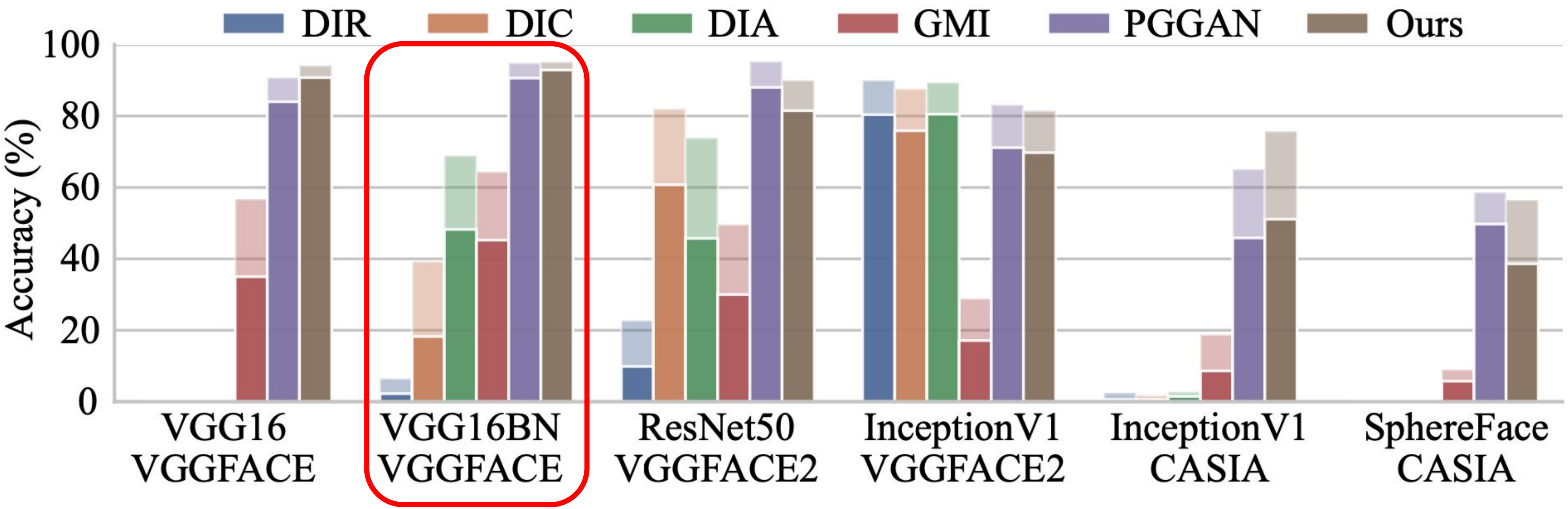
Evaluation - Whitebox Inversion Generalizability

Can different models trained on the same dataset recognize the inverted images?



Evaluation - Whitebox Inversion Generalizability

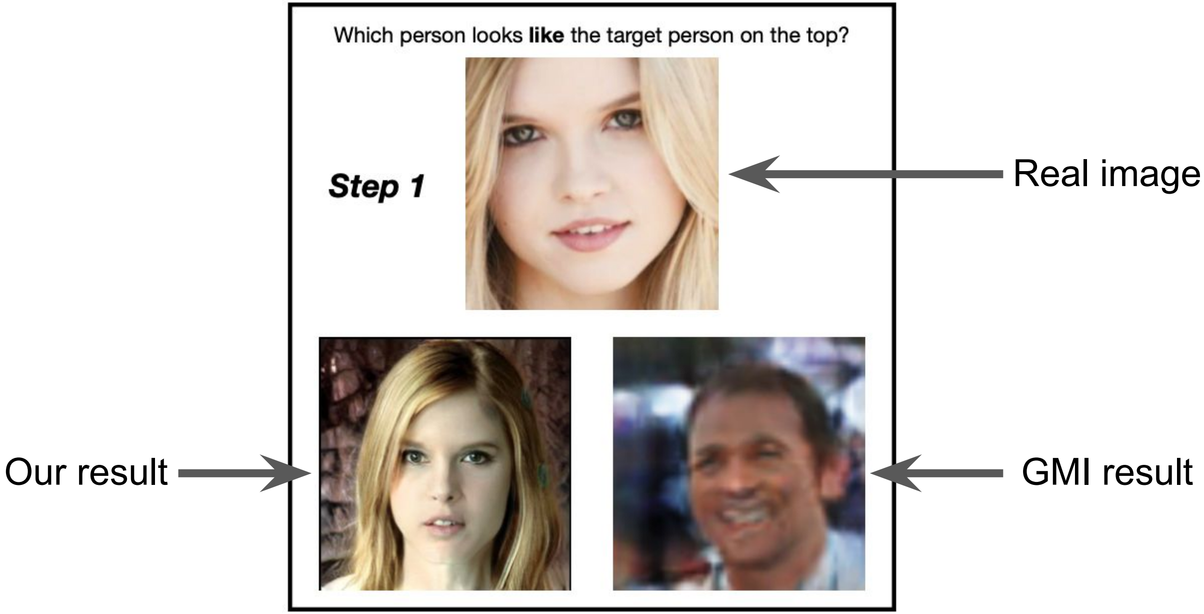
Can different models trained on the same dataset recognize the inverted images?



Invert VGG16BN
Evaluate on VGG16

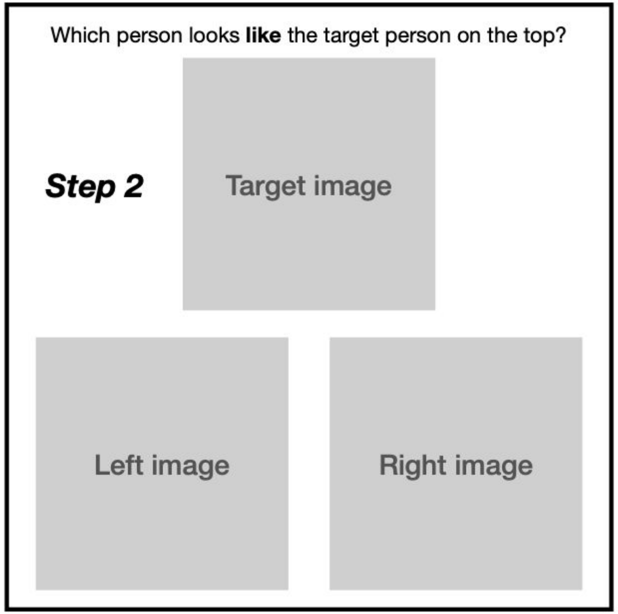
Evaluation - Whitebox Inversion Human Study (Relative)

Do humans think our method is better than baselines?



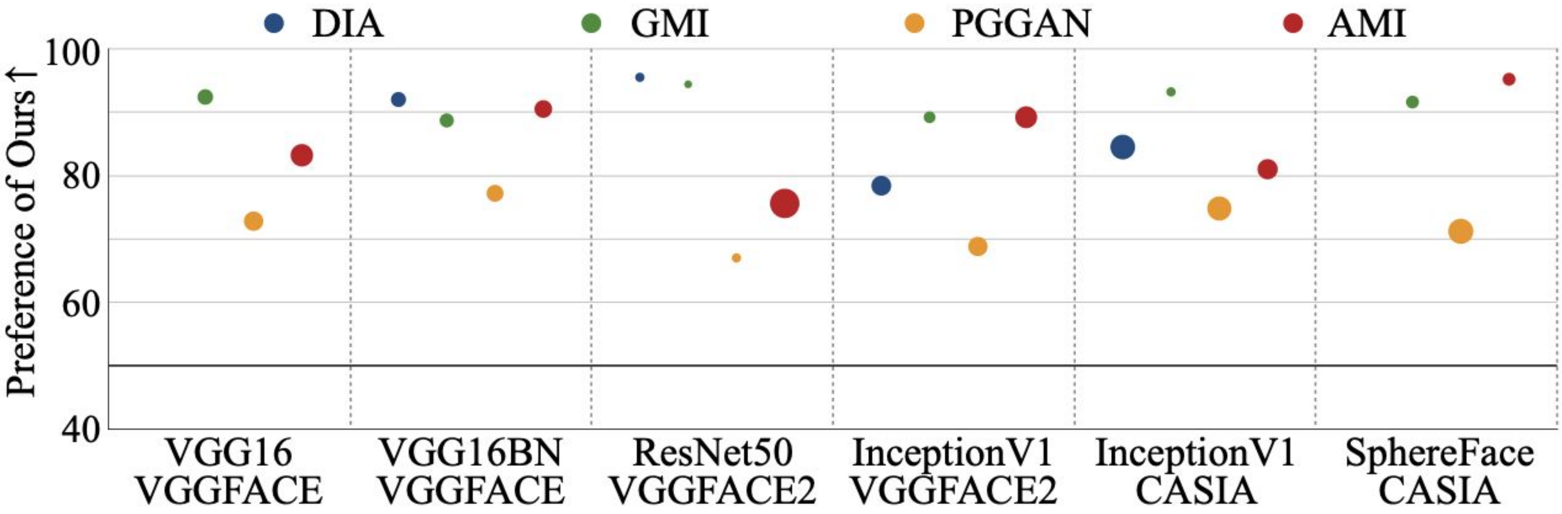
Evaluation - Whitebox Inversion Human Study (Relative)

Do humans think our method is better than baselines?



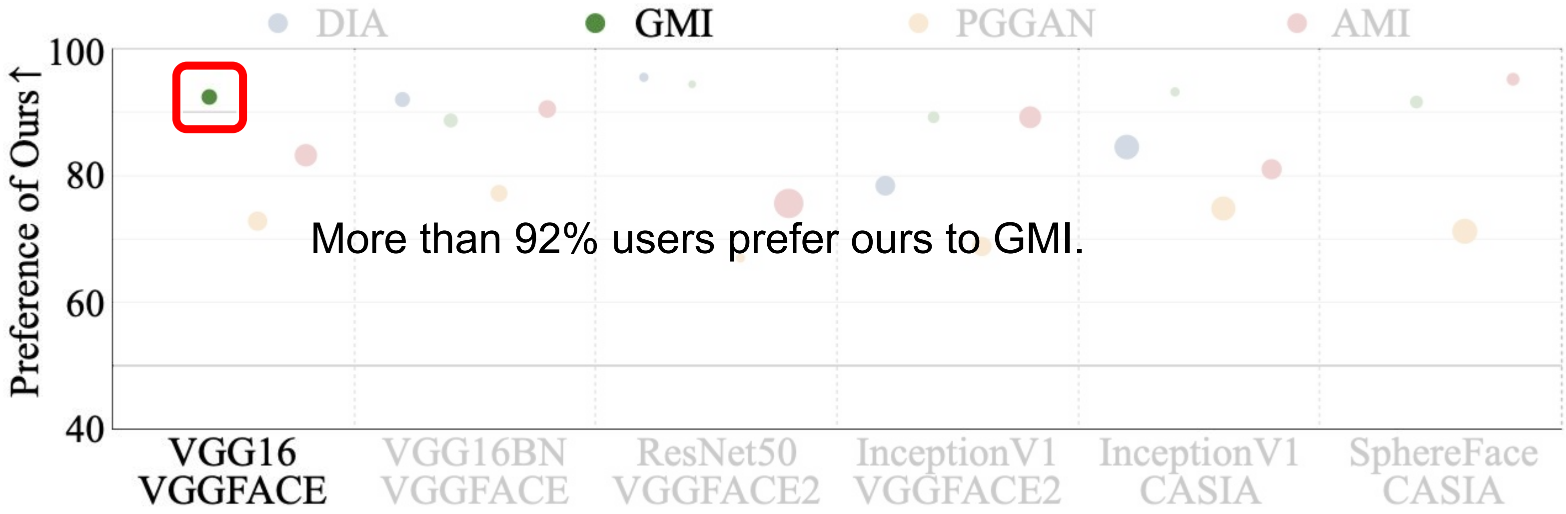
Evaluation - Whitebox Inversion Human Study (Relative)

Do humans think our method is better than baselines?



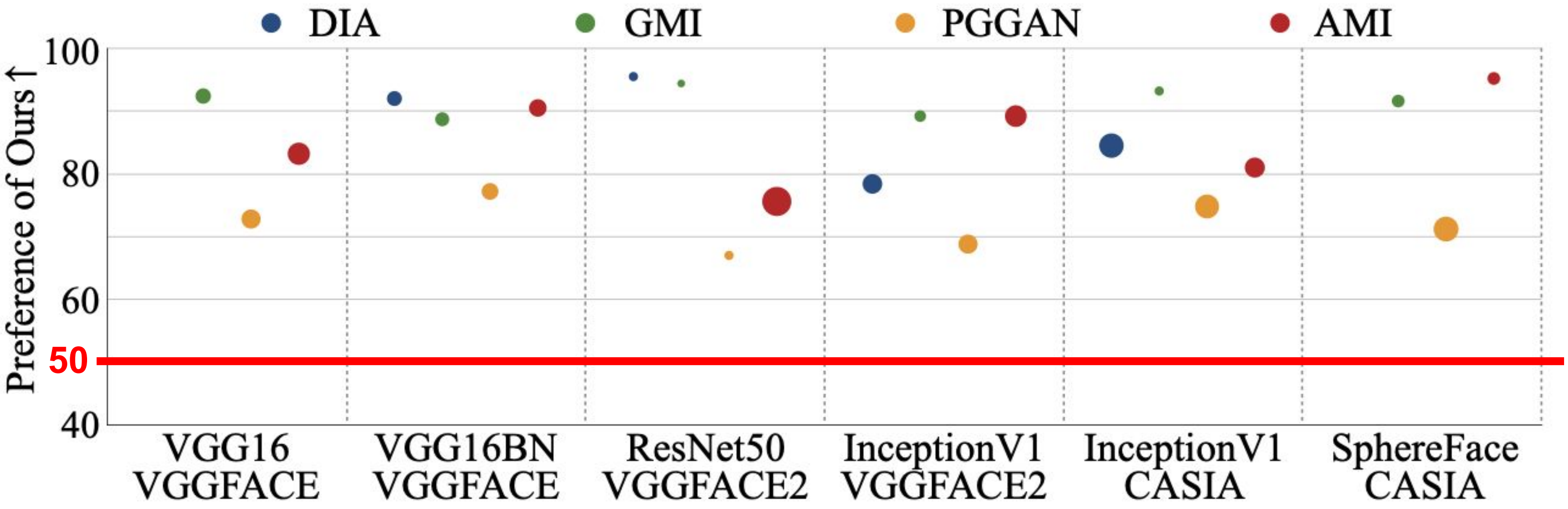
Evaluation - Whitebox Inversion Human Study (Relative)

Do humans think our method is better than baselines?



Evaluation - Whitebox Inversion Human Study (Relative)

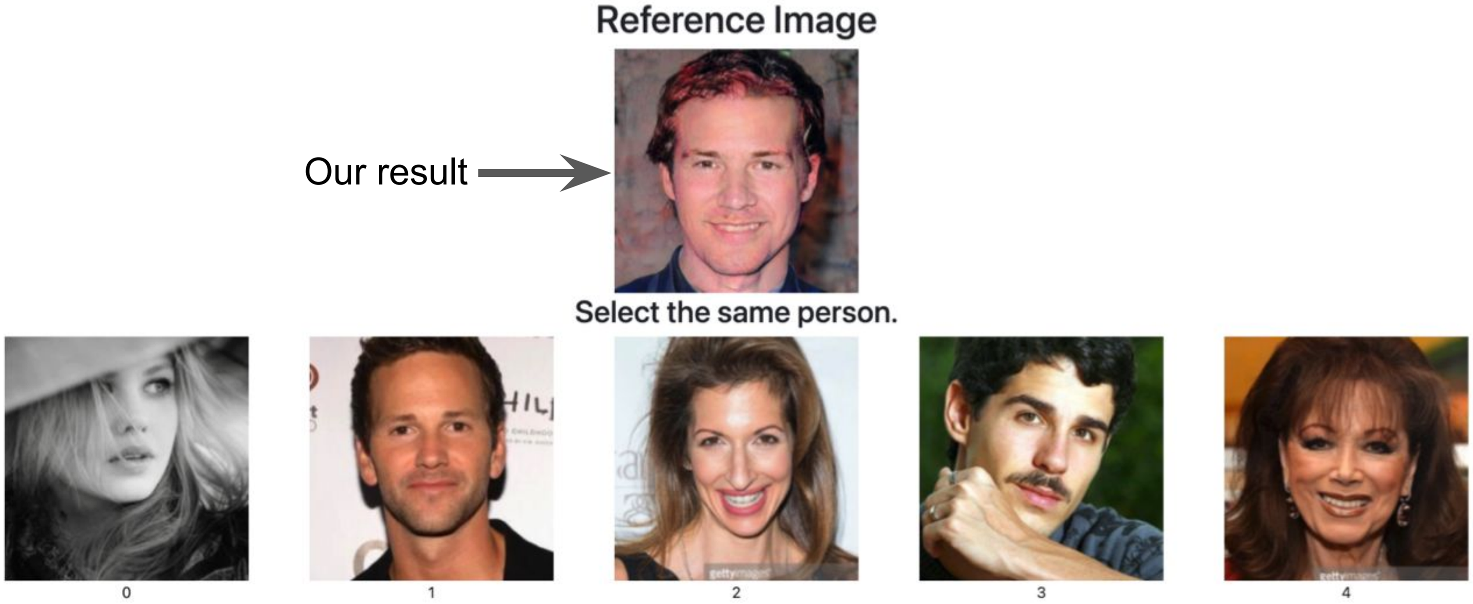
Do humans think our method is better than baselines?



Evaluation - Whitebox Inversion Human Study (Absolute)

Can they recognize the inverted person?

Our result

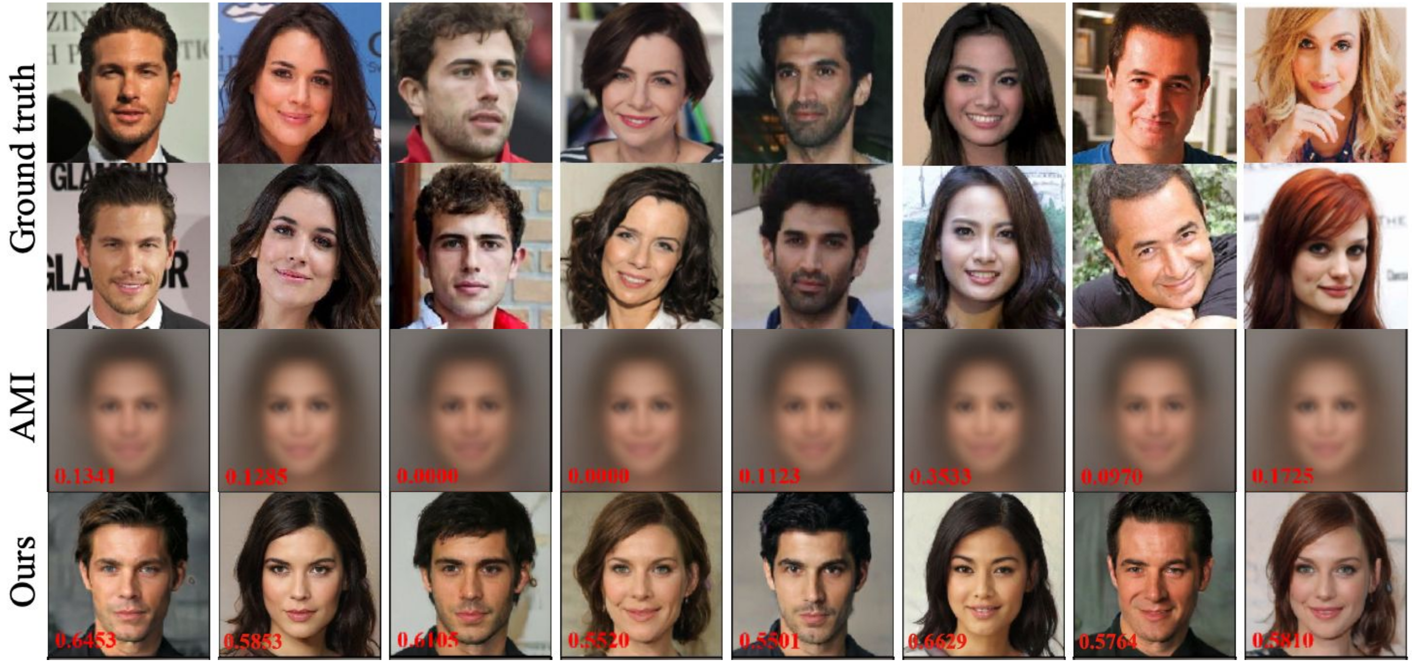


Average accuracy: 95.71%

Evaluation - Black-box Inversion Qualitative Results

Evaluation - Black-box Inversion on Commercial Services

Azure clarifai



Conclusion

Study challenges in the GAN-based model inversion

Propose StyleGAN-based model inversion in white-box and black-box settings

- Regularize W latent vectors in P space
- Use random dropout to mitigate feature overfitting
- Use consistent top-k labels to select the correct inversion

Images inverted by MIRROR have substantially better quality and fidelity compared to the existing methods

Open-sourced

Project page: https://model-inversion.github.io/mirror/

Code repo: https://github.com/njuaplusplus/mirror

Thank you!

Q & A