Neural Network Inversion in Adversarial Setting via Background Knowledge Alignment

Ziqi Yang
National University of Singapore
yangziqi@comp.nus.edu.sg

Jiyi Zhang
National University of Singapore
jzhang93@comp.nus.edu.sg

Ee-Chien Chang
National University of Singapore
changec@comp.nus.edu.sg

Zhenkai Liang
National University of Singapore
liangzk@comp.nus.edu.sg

- Linden, A.T., & Kindermann, J. (1989). ***Inversion of multilayer nets***. International 1989 Joint Conference on Neural Networks, 425-430 vol.2.
- Lee, S., & Kil, R.M. (1994). ***Inverse mapping of continuous functions using local and global information***. IEEE transactions on neural networks, 5 3, 409-23 .

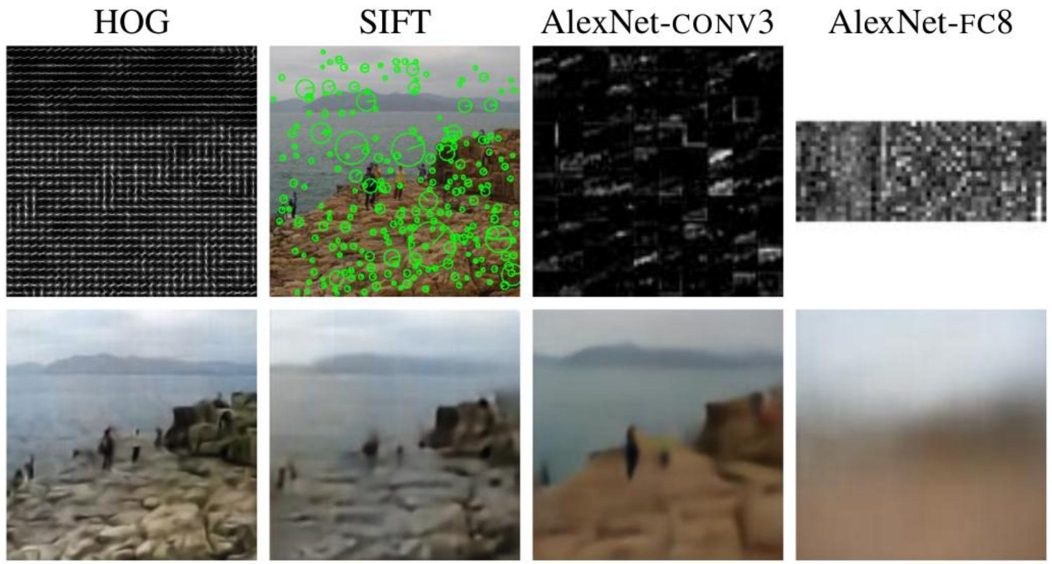


Figure 1: We train convolutional networks to reconstruct images from different feature representations. **Top row:** Input features. **Bottom row:** Reconstructed image. Re-

Figure 1: An image recovered using a new model inversion attack (left) and a training set image of the victim (right). The attacker is given only the person's name and access to a facial recognition system that returns a class confidence score.

Amazon Rekognition API

a cloud-based computer vision platform

Website: https://aws.amazon.com/rekognition/

Amazon Rekognition API

a real prediction sample

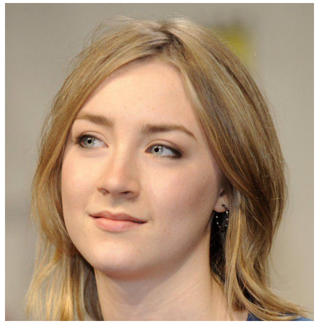

```
... ...
"Emotions": {
    "CONFUSED": 0.06156736373901367,
    "ANGRY": 0.5680691528320313,
    "CALM": 0.274930419921875,
    "SURPRISED": 0.01476531982421875,
    "DISGUSTED": 0.030669870376586913,
    "SAD": 0.044896211624145504,
    "HAPPY": 0.0051016128063201905
},
"Smile": 0.003313331604003933,
"MouthOpen": 0.0015682983398437322,
"Beard": 0.9883685684204102,
"Sunglasses": 0.00017322540283204457,
"EyesOpen": 0.9992143630981445,
... ...
```

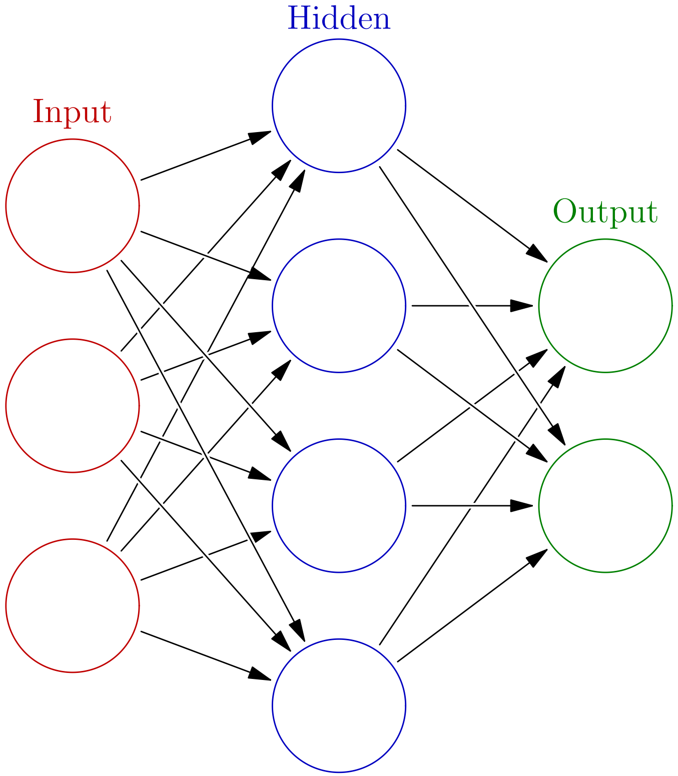


```
{
    "Emotions": {
        "CONFUSED": 0.06156736373901367,
        "ANGRY": 0.5680691528320313,
        "CALM": 0.274930419921875,
        "SURPRISED": 0.01476531982421875,
        "DISGUSTED": 0.030669870376586913,
        "SAD": 0.044896211624145504,
        "HAPPY": 0.0051016128063201905
    },
    "Smile": 0.003313331604003933,
    "MouthOpen": 0.0015682983398437322,
    "Beard": 0.9883685684204102,
    "Sunglasses": 0.00017322540283204457,
    "EyesOpen": 0.9992143630981445,
    "Mustache": 0.07934749603271485,
    "Eyeglasses": 0.0009058761596679732,
    "Gender": 0.998325424194336,
    "AgeRange": {
        "High": 0.52,
        "Low": 0.35
    },
    "Pose": {
        "Yaw": 0.398555908203125,
        "Pitch": 0.532116775512695,
        "Roll": 0.47806625366211
    },
    "Landmarks": {
        "eyeLeft": {"X": 0.2399402886140542, "Y": 0.3985823600850207},
        "eyeRight": {"X": 0.5075000426808342, "Y": 0.3512716902063248},
        "mouthLeft": {"X": 0.294372202920132, "Y": 0.7884027359333444},
        "mouthRight": {"X": 0.5111179957624341, "Y": 0.7514958062070481},
        "nose": {"X": 0.26335677944245883,"Y": 0.5740609671207184},
        "leftEyeBrowLeft": {"X": 0.16586835071688794, "Y": 0.33359158800003375},
        "leftEyeBrowRight": {"X": 0.2344663348354277, "Y": 0.27319636750728526},
        "leftEyeBrowUp": {"X": 0.1791416455487736, "Y": 0.27319679970436905},
        "rightEyeBrowLeft": {"X": 0.39377442930565504, "Y": 0.24260599816099127},
        "rightEyeBrowRight": {"X": 0.653192506461847, "Y": 0.24797691132159944},
        "rightEyeBrowUp": {"X": 0.4985808427216577, "Y": 0.21011433981834574},
        "leftEyeLeft": {"X": 0.2108403727656505, "Y": 0.40527320313960946},
        "leftEyeRight": {"X": 0.29524428727196866, "Y": 0.3945644398953052},
        "leftEyeUp": {"X": 0.2320460442636834, "Y": 0.38003991664724146},
        "leftEyeDown": {"X": 0.24090847324152462, "Y": 0.4139932115027245},
        "rightEyeLeft": {"X": 0.4582430085197824, "Y": 0.3677093338459096},
        "rightEyeRight": {"X": 0.5775697973907971, "Y": 0.34774452980528486},
        "rightEyeUp": {"X": 0.5040715541995939, "Y": 0.3371239347660795},
        "rightEyeDown": {"X": 0.5091470851272833, "Y": 0.37251352858036124},
        "noseLeft": {"X": 0.2878986010785963, "Y": 0.6362120963157492},
        "noseRight": {"X": 0.40161600660105223, "Y": 0.6085103161791537},
        "mouthUp": {"X": 0.34124040994487825, "Y": 0.705847150214175},
        "mouthDown": {"X": 0.3709446289500252, "Y": 0.8184411896036027},
        "leftPupil": {"X": 0.2399402886140542, "Y": 0.3985823600850207},
        "rightPupil": {"X": 0.5075000426808342, "Y": 0.3512716902063248},
        "upperJawlineLeft": {"X": 0.3066862049649973, "Y": 0.4463287926734762},
        "midJawlineLeft": {"X": 0.36578599351351376, "Y": 0.8324899719116535},
        "chinBottom": {"X": 0.45123760622055803, "Y": 1.0087064474187},
        "midJawlineRight": {"X": 0.8626791375582336, "Y": 0.7551260456125787},
        "upperJawlineRight": {"X": 0.9242277731660937,"Y": 0.348934908623391}
    }
}
```


the complete result of the left partial prediction

Generic Neural Network





0.76
0.01
0.03
0.04
0.01
0.01
0.08
0.02
0.03
0.01

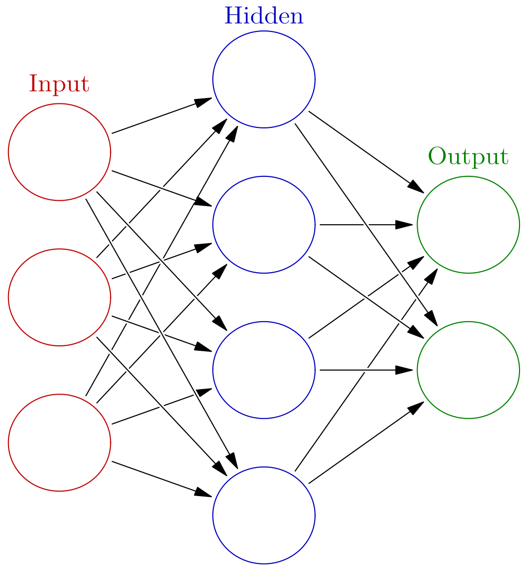
Input: $\mathbf{x}$

Classifier: F_w

Prediction: $F_w(\mathbf{x})$

Model Inversion Attack

Can we inverse the prediction process, inferring input $\mathbf{x}$ from prediction $F_w(\mathbf{x})$?

? ←  ←

0.76
0.01
0.03
0.04
0.01
0.01
0.08
0.02
0.03
0.01

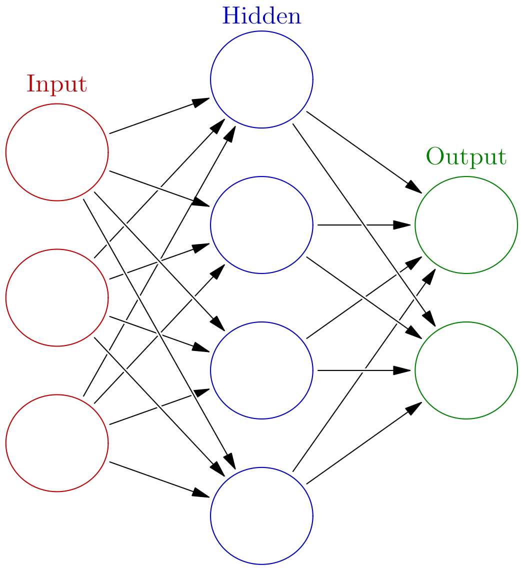
Input: $\mathbf{x}$

Classifier: F_w

Prediction: $F_w(\mathbf{x})$

Adversarial Settings

? ←



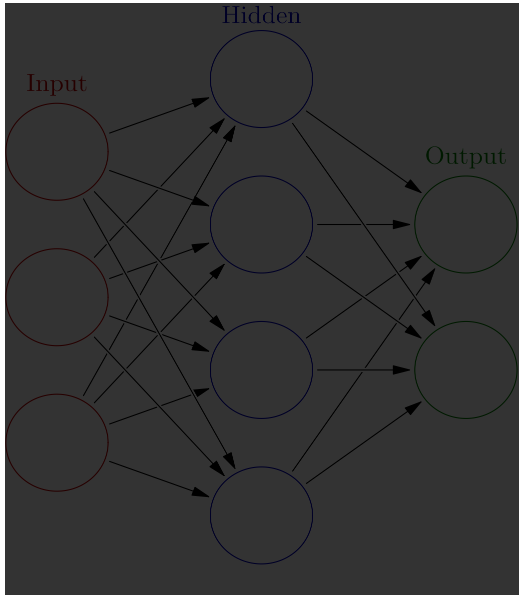
←

0.76
0.01
0.03
0.04
0.01
0.01
0.08
0.02
0.03
0.01

Input: $\mathbf{x}$ Classifier: F_w Prediction: $F_w(\mathbf{x})$

For a realistic adversary, access to many components should be restricted.

Adversarial Settings

? ← 

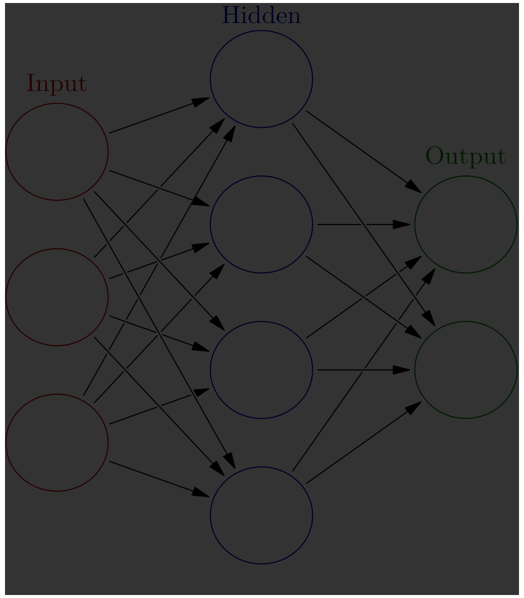
←

0.76
0.01
0.03
0.04
0.01
0.01
0.08
0.02
0.03
0.01

Input: $\mathbf{x}$ Classifier: F_w Prediction: $F_w(\mathbf{x})$

- Black-box classifier F_w

Adversarial Settings



←

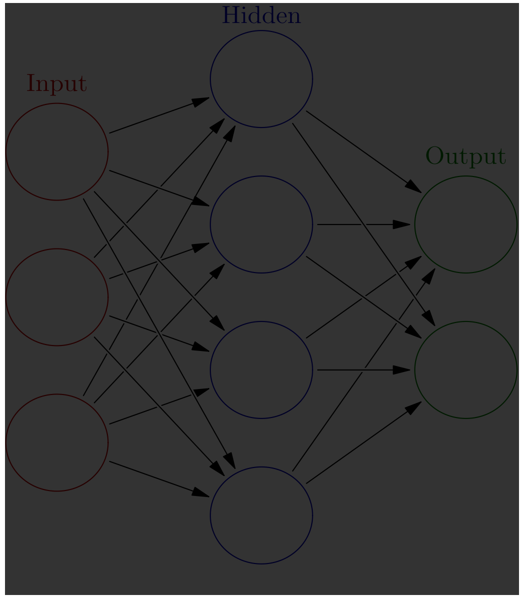
0.76
0.01
0.03
0.04
0.01
0.01
0.08
0.02
0.03
0.01

Classifier: F_w

Prediction: $F_w(\mathbf{x})$

- Black-box classifier F_w
- No access to training data

Adversarial Settings



0.76
0.00
0.00
0.04
0.00
0.00
0.08
0.00
0.00
0.00

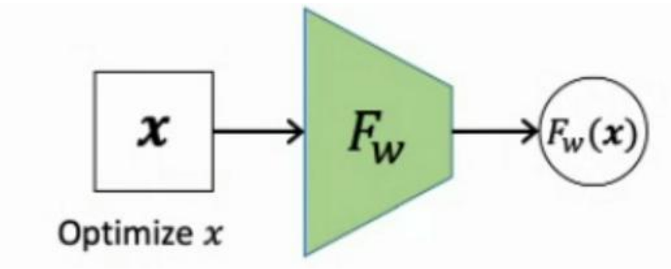
Classifier: F_w

Partial Prediction (top3 values): $F_w(\mathbf{x})'$

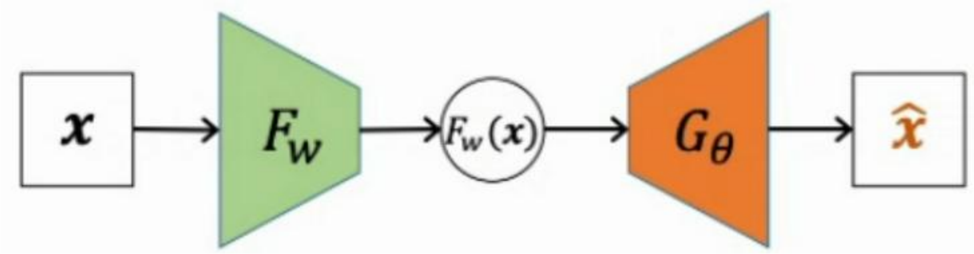
- Black-box classifier F_w
- No access to training data
- Partial prediction results $F_w(\mathbf{x})'$

Related Works

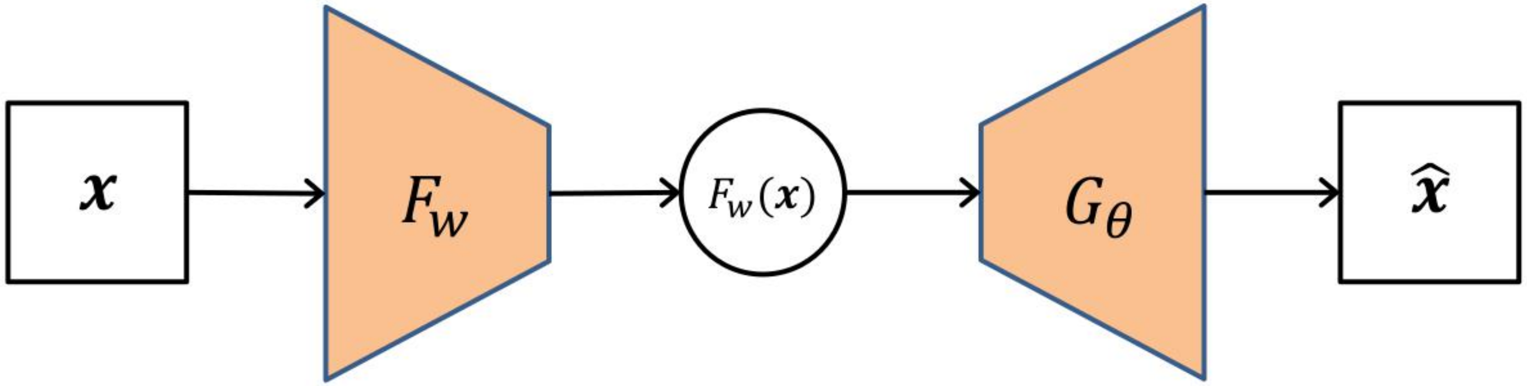
- Optimization-based inversion
  - White-box F_w
    - Cast it as an optimization problem of **x**
  - Unsatisfactory inversion quality
    - no notion of semantics in optimization
  - Simple F_w only
    - not for complex neural network (6s on GPU, while training-based 5ms)



- Training-based inversion (non-adversarial)
  - Learn a second model G_ϑ
    - act as the inverse of F_w
  - Train G_ϑ on F_w's training data
  - Full prediction results $F_w(\mathbf{x})$



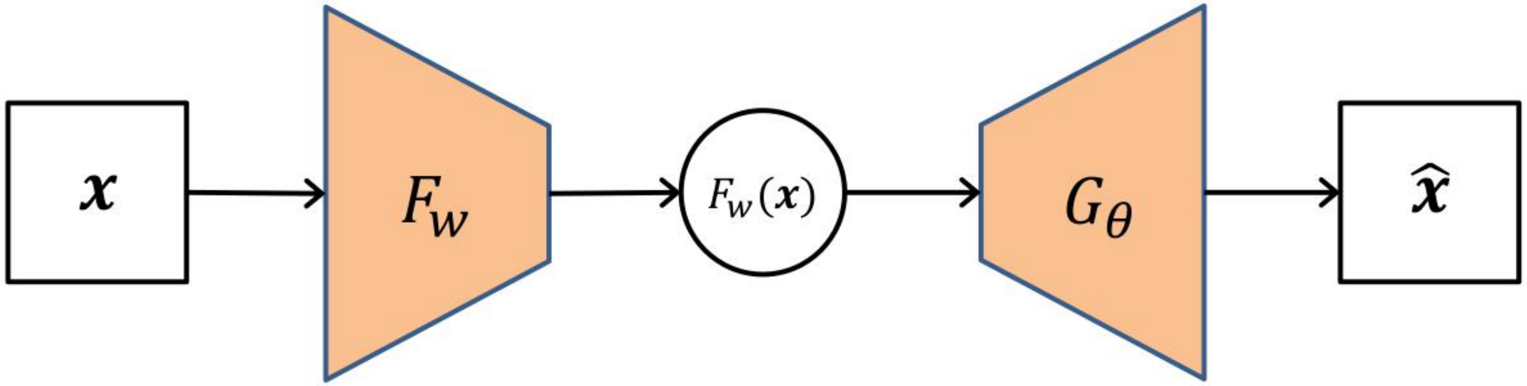
Training-based Inversion



Notations

- F_w: black-box classifier
- $F_w(\mathbf{x})$: prediction
- $\text{trunc}_m(F_w(\mathbf{x}))$: truncated (partial) prediction. m is the number of retained values after truncation, e.g., retaining top-3 values, m = 3
- G_ϑ: inversion model

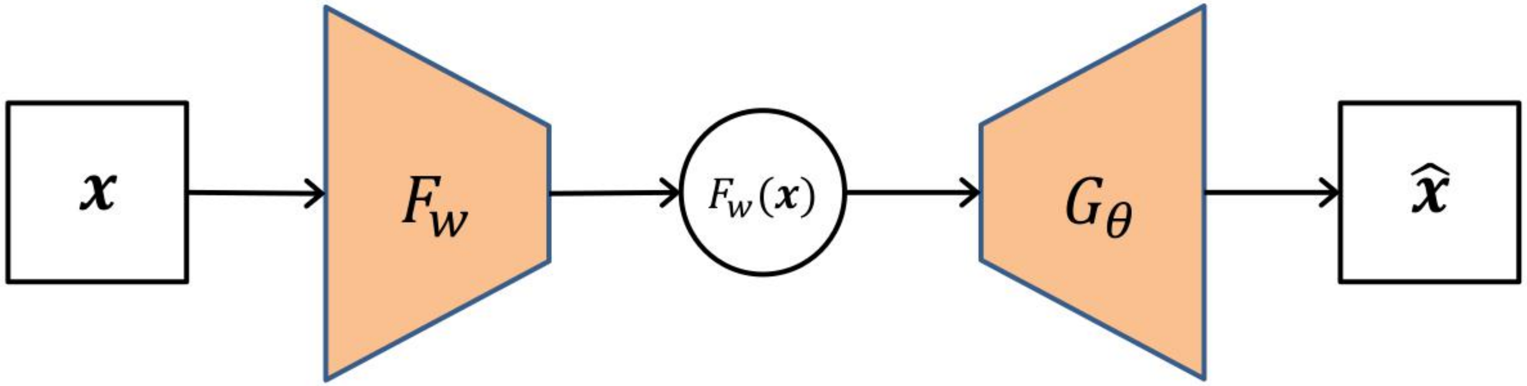
Training-based Inversion



So we have,

- $\hat{\mathbf{x}} = G_\theta(\,\mathrm{trunc}_m(\,F_w(\,\mathbf{x}\,)\,)\,)$

Training-based Inversion



Inversion model training objective: to minimize the reconstruction loss between **x** and $\hat{\mathbf{x}}$ (The author used **a** in the paper)

$$C(G_\theta) = \mathbb{E}_{\mathbf{a} \sim p_a}[\mathcal{R}(G_\theta(\mathrm{trunc}_m(F_w(\mathbf{a}))), \mathbf{a})]$$

R is the reconstruction loss, usually implemented as Mean Square Loss. And p_a is the training data distribution.

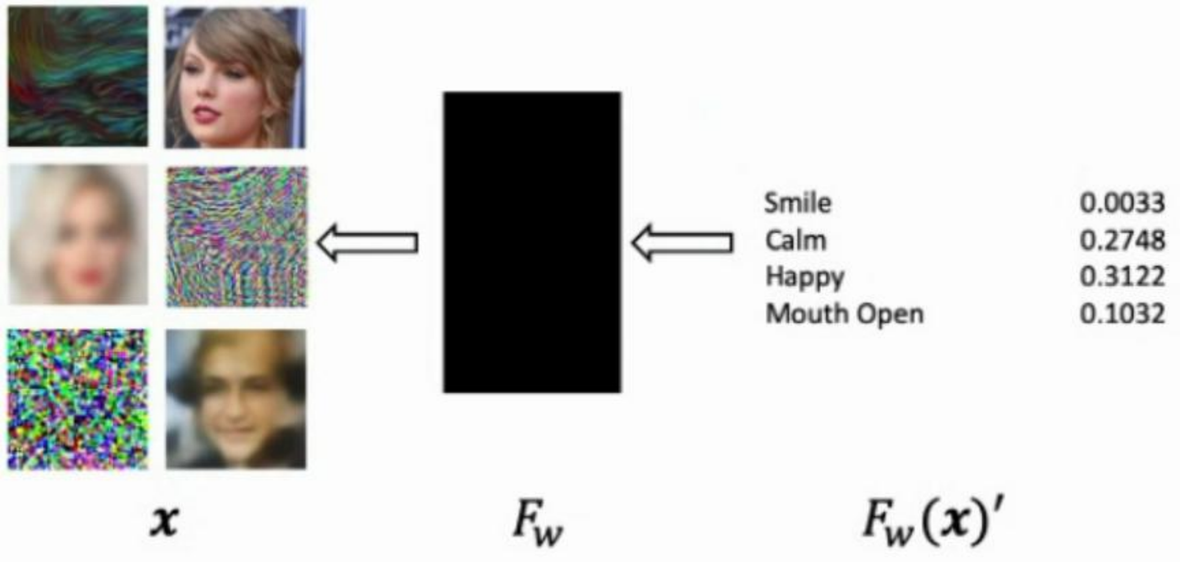
Training-based Inversion

$$C(G_\theta) = \mathbb{E}_{\mathbf{a}\sim p_a}[\mathcal{R}(G_\theta(\mathrm{trunc}_m(F_w(\mathbf{a}))), \mathbf{a})]$$

- Two practical problems
  - training data distribution p_a is intractable
    - use training dataset *D* to approximate p_a
  - adversary can't access training dataset *D*
    - use auxiliary dataset *D'*, which is sampled from a more generic distribution than p_a, e.g., crawl face images from the Internet, as auxiliary dataset for attacking Amazon Rekognition

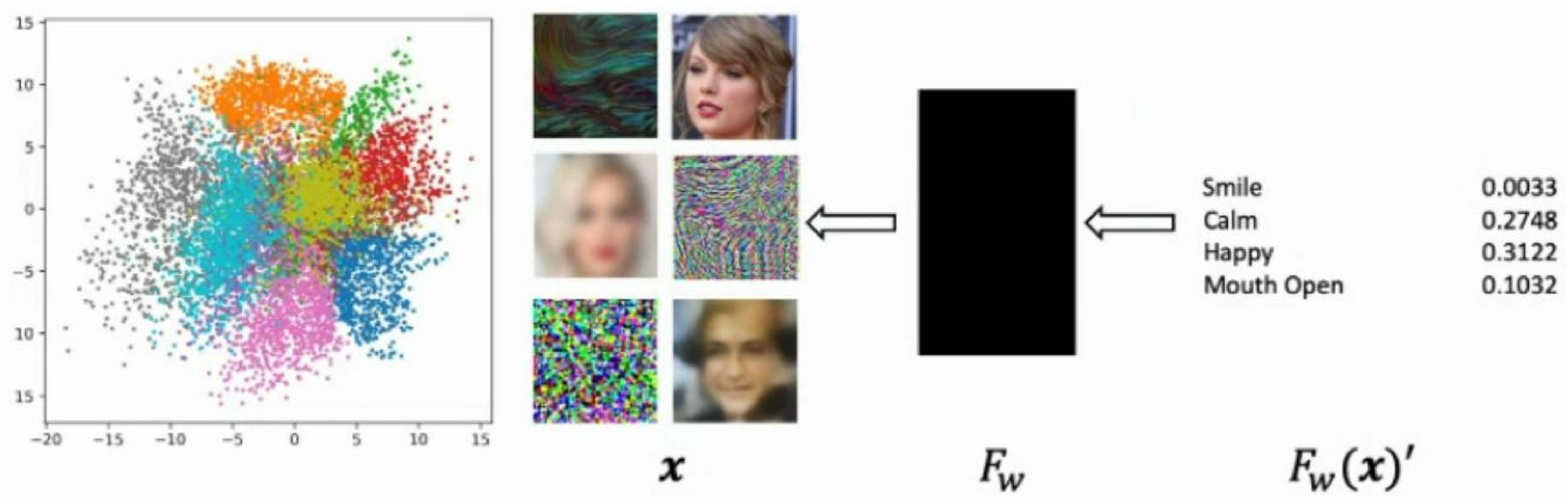
Background Knowledge Alignment

- Neural network inversion is an ill-posed problem
  - Many inputs can yield the same truncated prediction
  - Which **x** is the one we want?



Background Knowledge Alignment

- Neural network inversion is an ill-posed problem
  - Which **x** is the one we want?
  - Expected **x** should follow the underlying data distribution



Background Knowledge Alignment

- Neural network inversion is an ill-posed problem
  - Which **x** is the one we want?
  - Expected **x** should follow the underlying data distribution
  - Learn training data distribution from auxiliary dataset, which is sampled from a more generic distribution

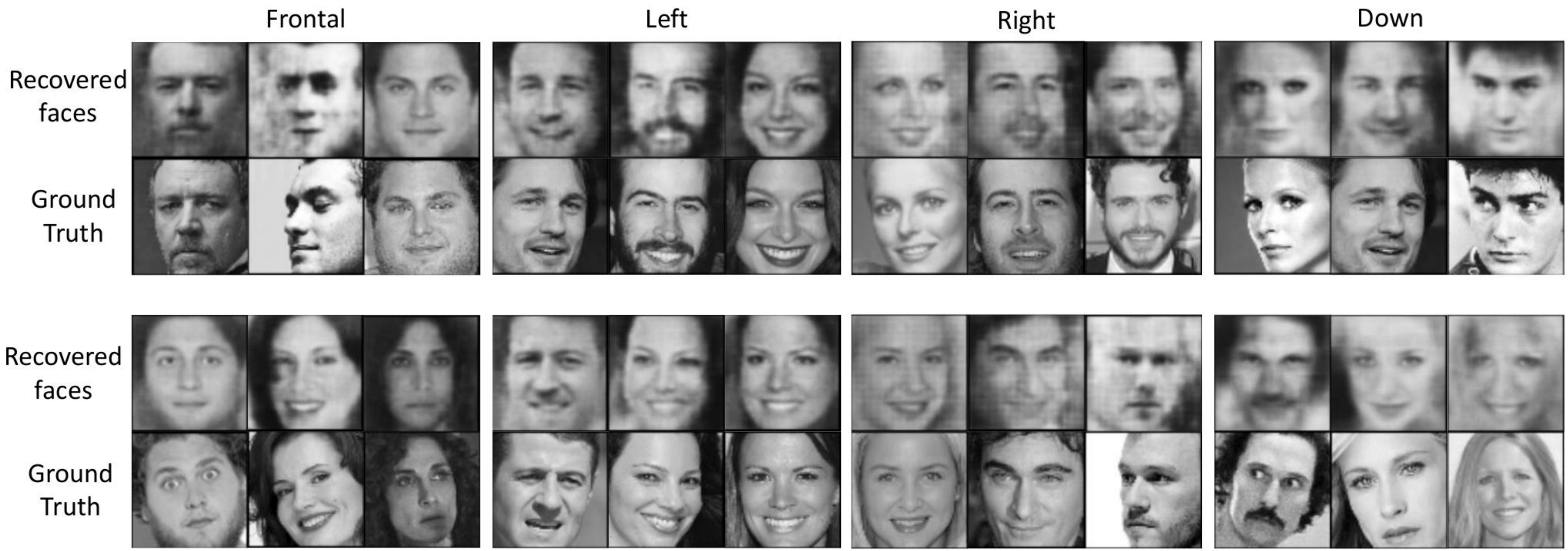
Background Knowledge Alignment

An example to show how the inversion model learns data distribution from the aligned auxiliary dataset.

- Sample images that look to different directions
- Align them to four different inversion model training set

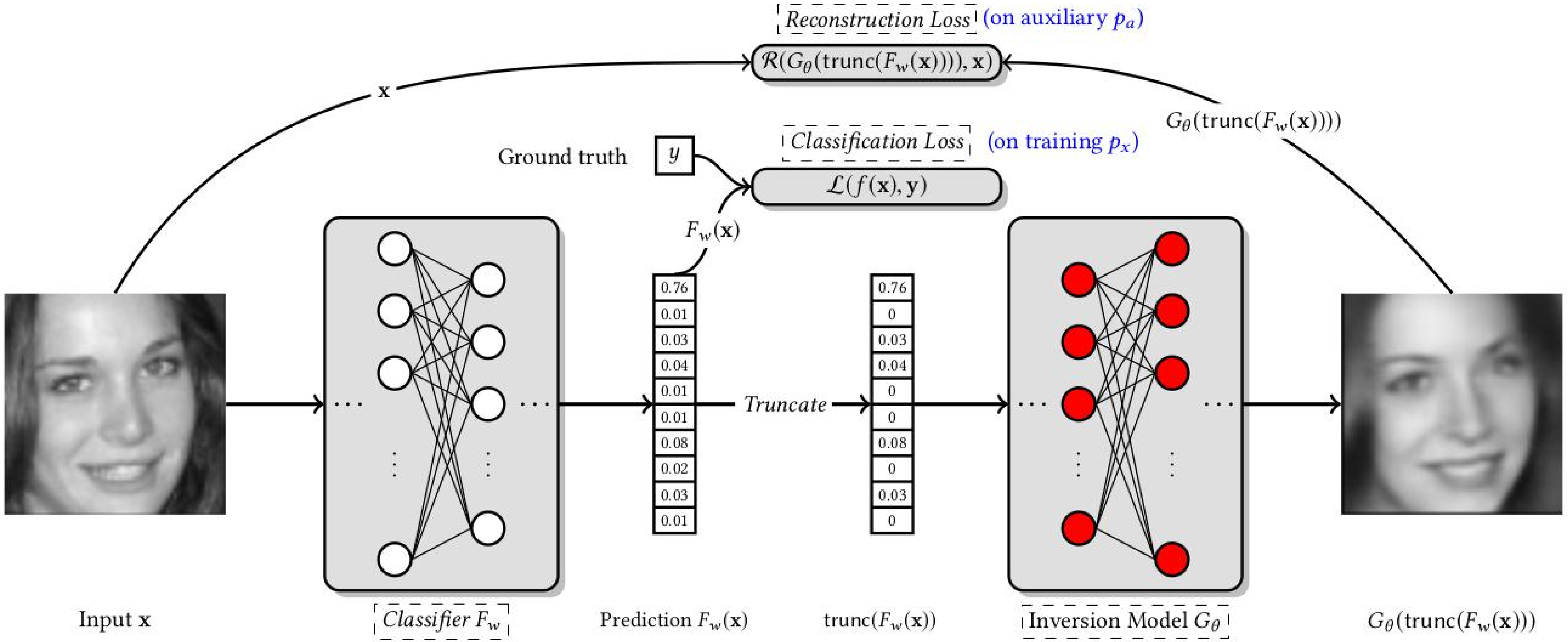
D_{left} D_{right} D_{down} D_{up}

Background Knowledge Alignment



Ground truth faces may look to different directions, but the recovered faces all look to the aligned direction.

Methodology



Evaluation

- Effect of auxiliary set
- Effect of truncation
- Attacking commercial prediction API

Datasets

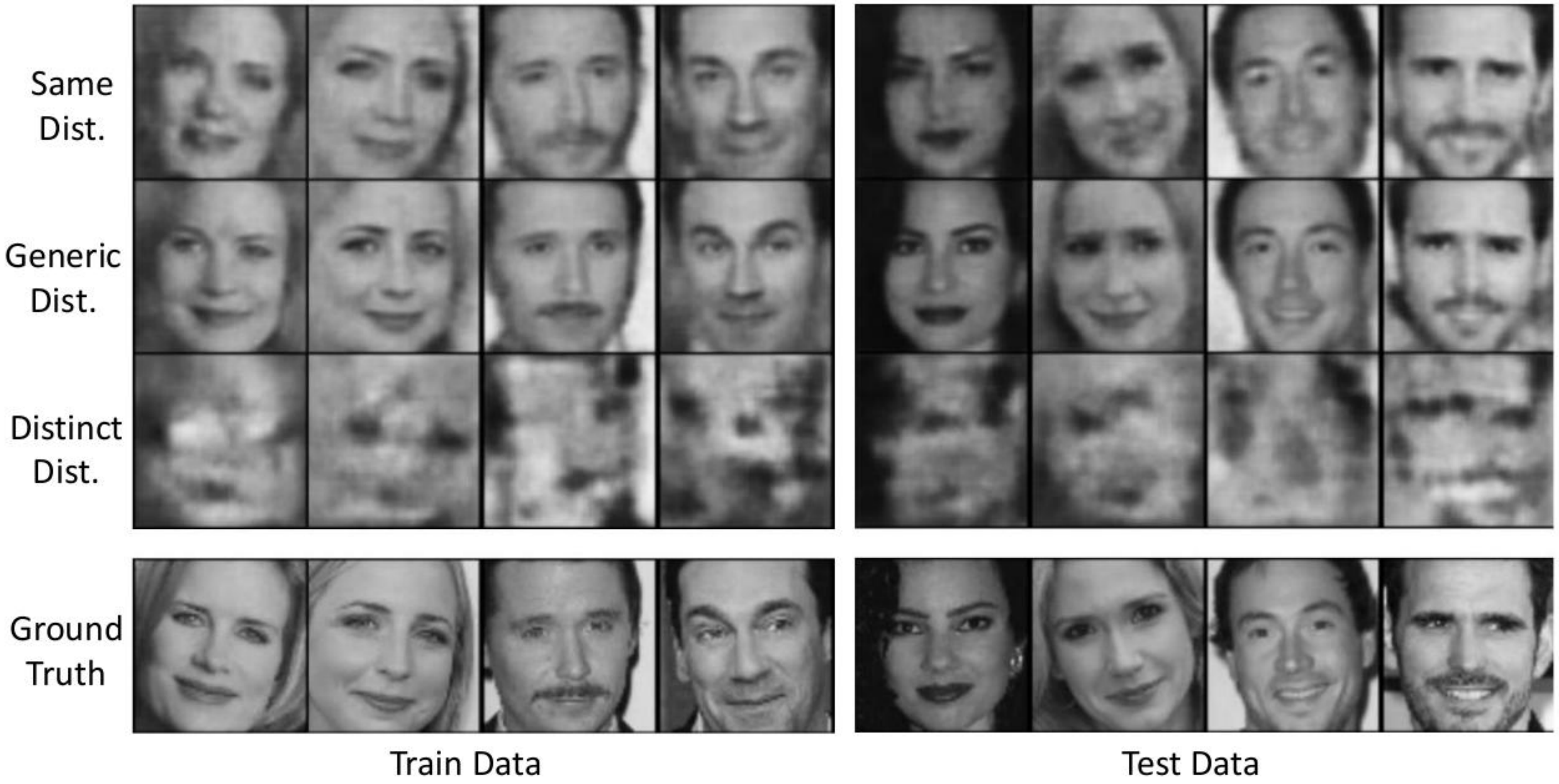
- FaceScrub: 100,000 images of 530 individuals
- CelebA: 202,599 images of 10,177 celebrities. Remark that the author removed 297 celebrities included in FaceScrub
- CIFAR10
- MNIST

Effect of Auxiliary Set

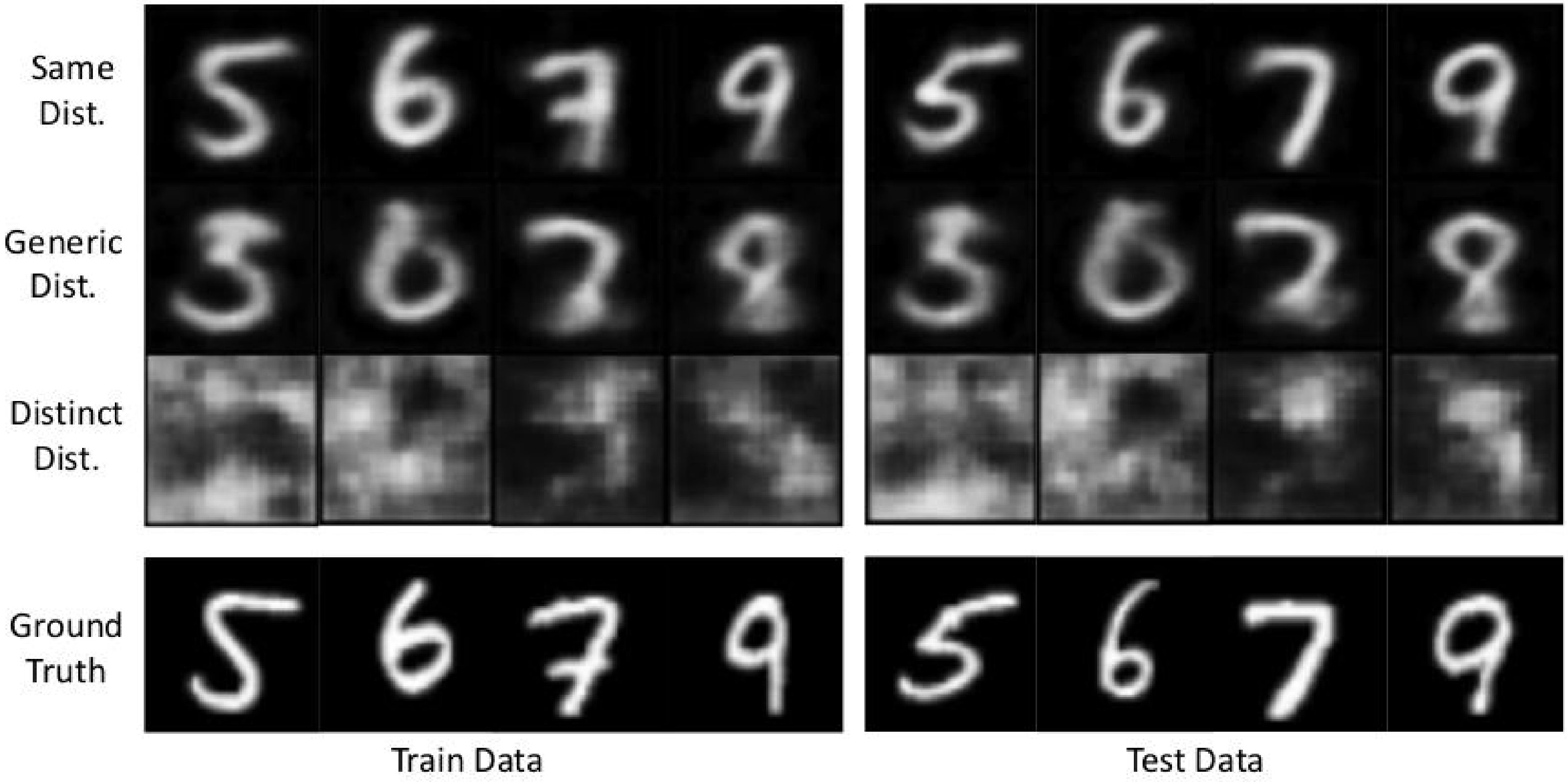
Three parts:

- train inversion model on classifier F_w's training dataset (Same distribution)
- a more generic dataset (Generic distribution), e.g. train classifier on FaceScrub, and train inversion model on CelebA
- a distinct dataset (Distinct distribution), e.g. train classifier on FaceScrub, and train inversion model on CIFAR10

Effect of Auxiliary Set



Effect of Auxiliary Set



Effect of Auxiliary Set

Summary I: Even with no full knowledge about the classifier F_w's training data, accurate inversion is still possible by training G_θ using auxiliary samples drawn from a more generic distribution derived from background knowledge.

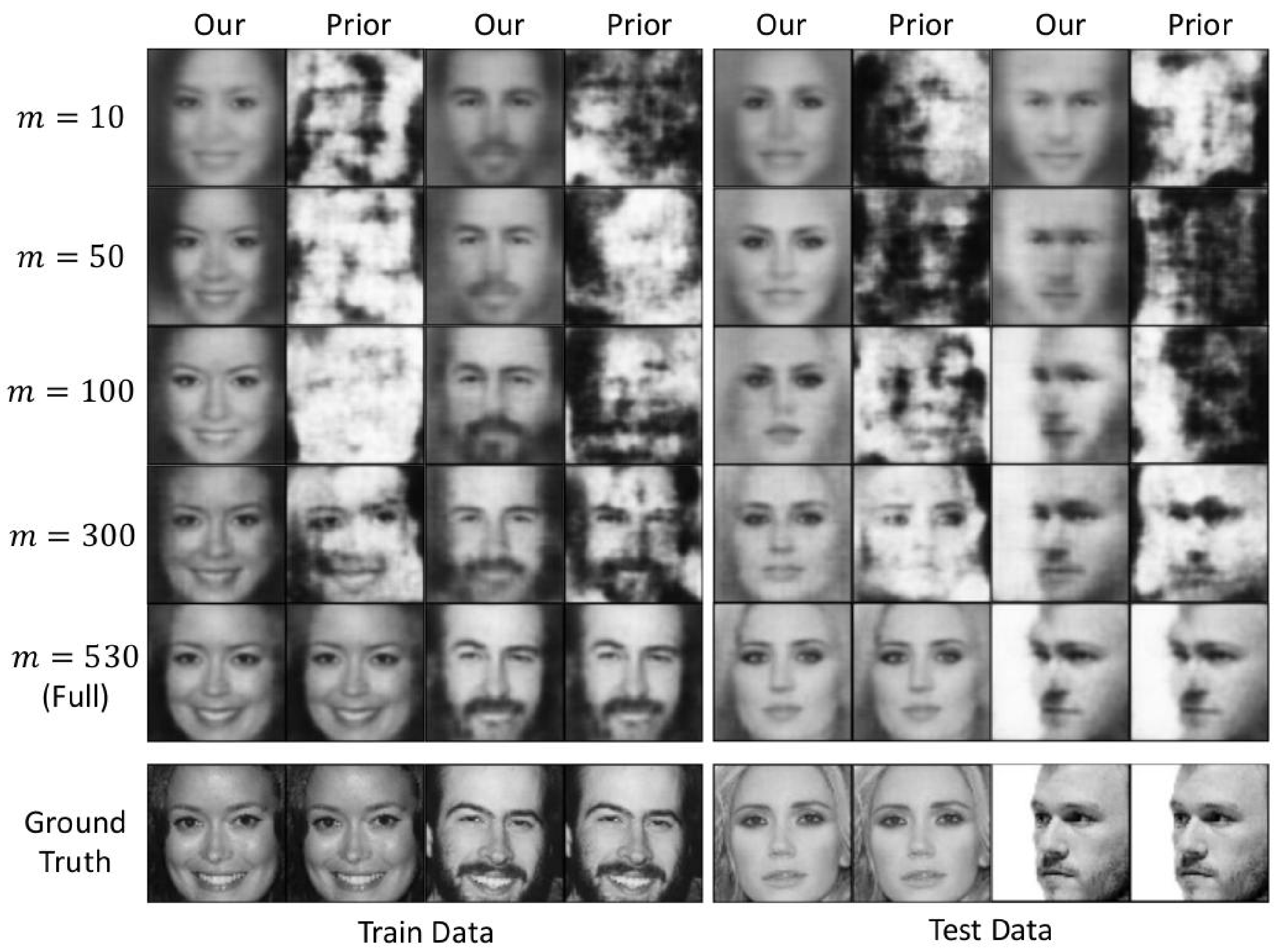
Effect of Truncation

$F_w(\mathbf{x})' = \text{trunc}_m(F_w(\mathbf{x}))$

Experiments: set m to different values

- 530 features in total, set m = 10, 50, 100, 300, 530

Effect of Truncation



Prior: prior works

Effect of Truncation

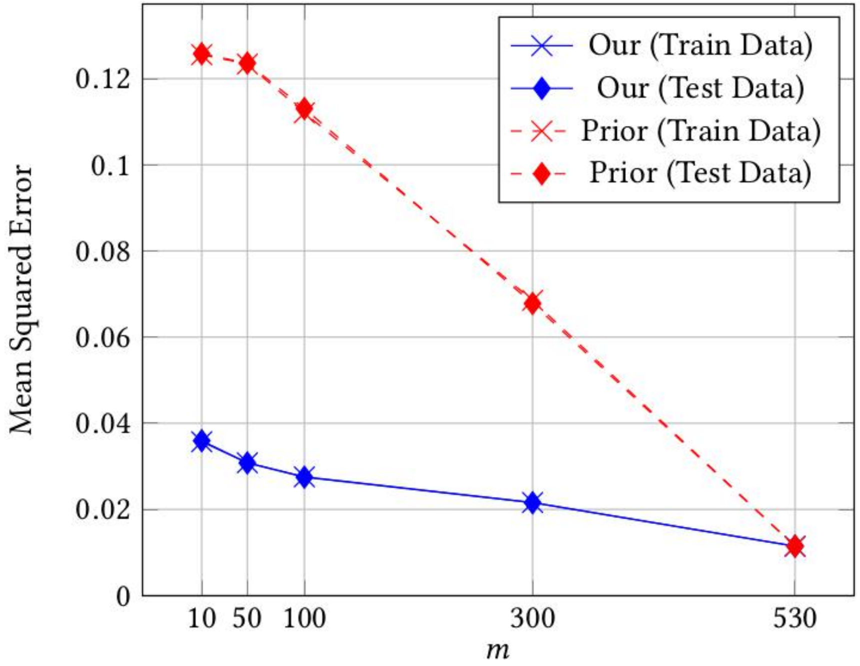


Figure 8: Quantitative measurement of the effect of truncation (m) for G_θ on the inversion quality on FaceScrub F_w. The x-axis is the m, and the y-axis is mean squared error.

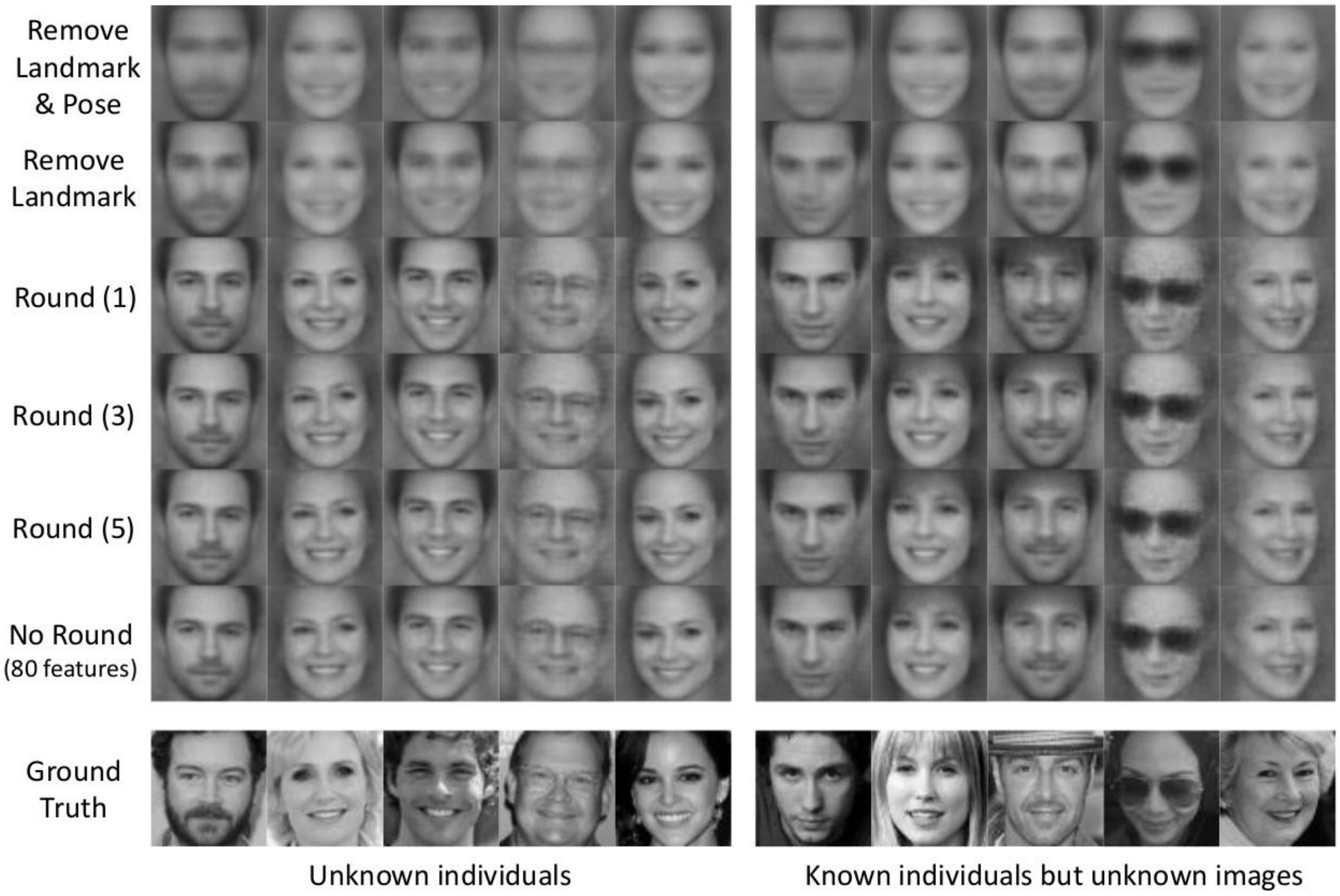
Summary II: Our truncation method of training the inversion model G_θ makes it still possible to perform accurate inversion when the adversary is given only partial prediction results.

Attacking commercial prediction API

Amazon Rekognition API

- no knowledge of backend model
- query API with auxiliary dataset to get training data for inversion model

Attacking commercial prediction API



Attacking commercial prediction API

Table 4: Quantitative measurement (mean squared error) of the inversion on Amazon Rekognition API.

Features	Unknown individuals	Known individuals but unknown images
Remove Landmark & Pose	0.0472	0.0469
Remove Landmark	0.0470	0.0462
Round(1)	0.0454	0.0443
Round(3)	0.0437	0.0438
Round(5)	0.0437	0.0438
No round (80 features)	0.0437	0.0438

Discussion

Contributions

- a successful training-based black-box model inversion attack
- extended experiments that provide insights into how inversion model learns data distribution from auxiliary dataset