Deep Generative models for Inverse Problems

Alex Dimakis

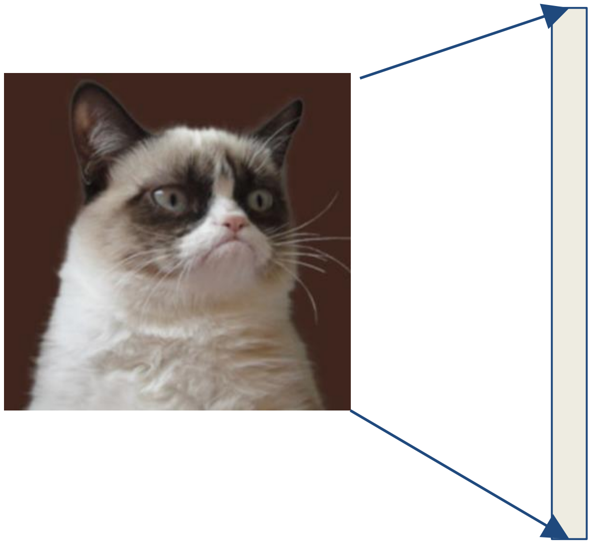
joint work with
Ashish Bora, Dave Van Veen and Ajil Jalal,
Sriram Vishwanath and Eric Price, UT Austin

Outline

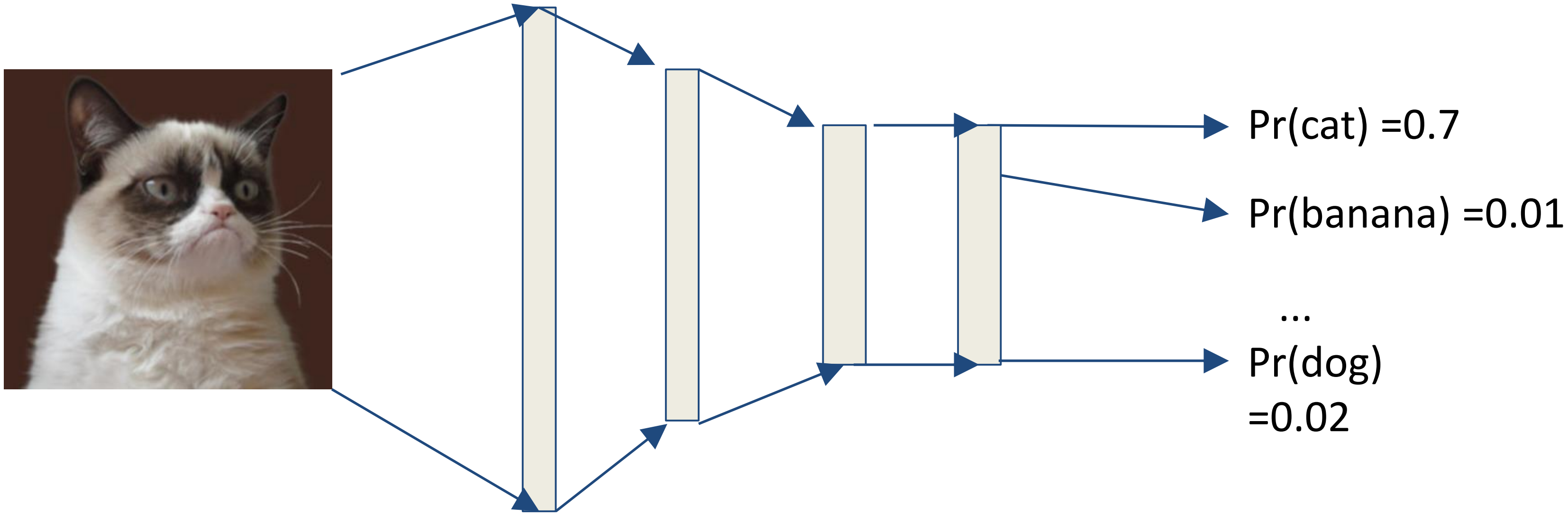
- Generative Models
- Using generative models for Inverse problems/compressed sensing
- Main theorem and proof technology
- Using an untrained GAN (Deep Image Prior)
- Conclusions

- Other extensions:
- Using non-linear measurements
- Using GANs to defend from Adversarial examples.
- AmbientGAN: Learning a distribution from noisy samples
- CausalGAN: Learning causal interventions.

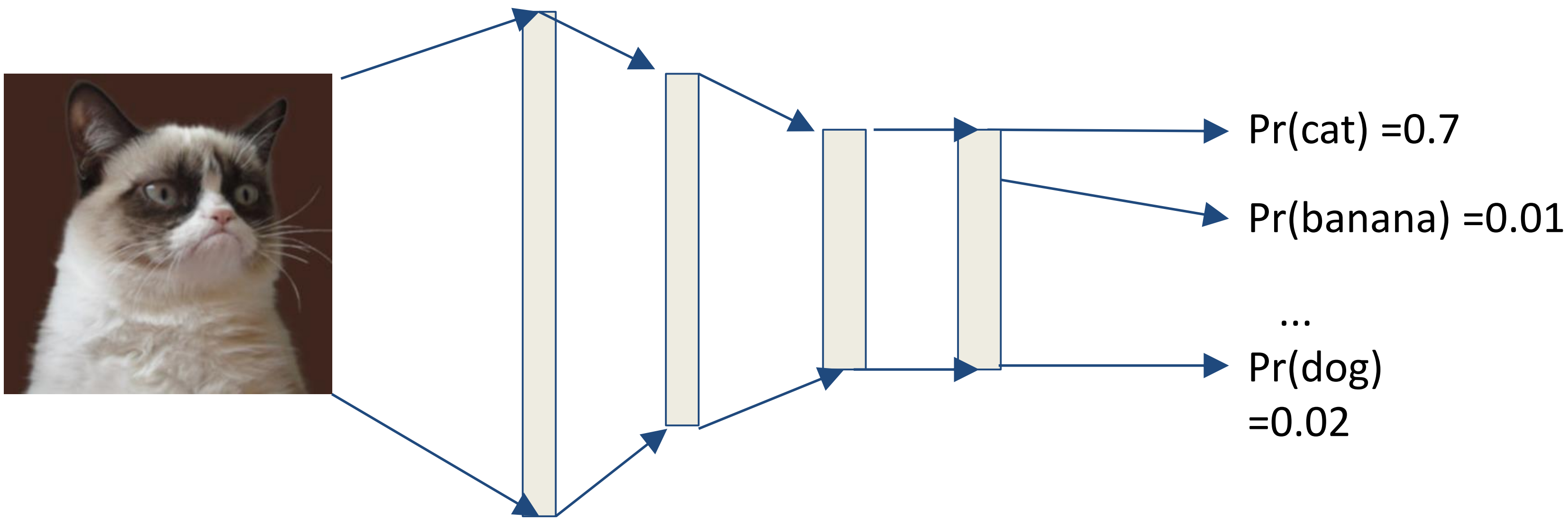
Types of Neural nets: Classifiers



Types of Neural nets: Classifiers

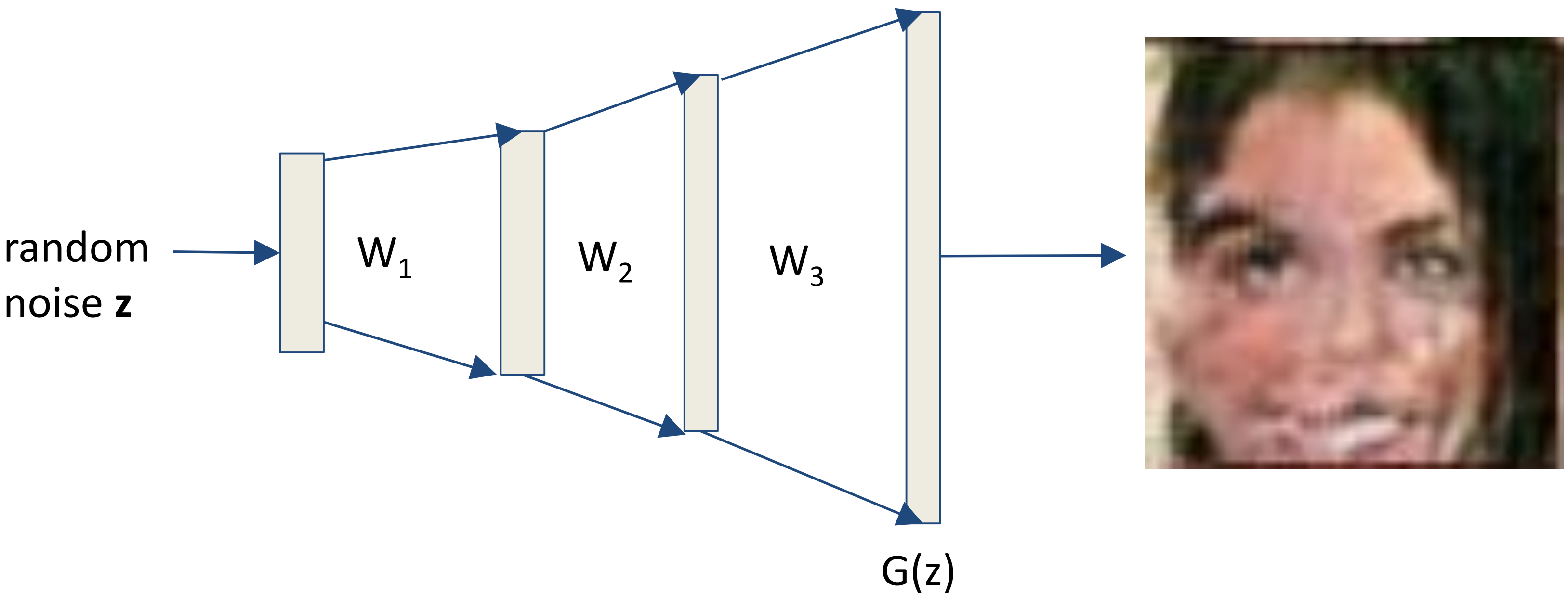


Types of Neural nets: Classifiers



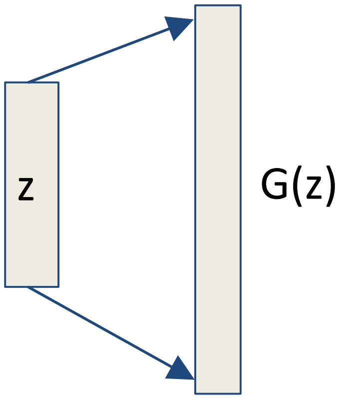
Supervised Learning= needs labeled data

Types of Neural nets: Generators



Unsupervised Learning= needs unlabeled data
Learns a high-dimensional distribution

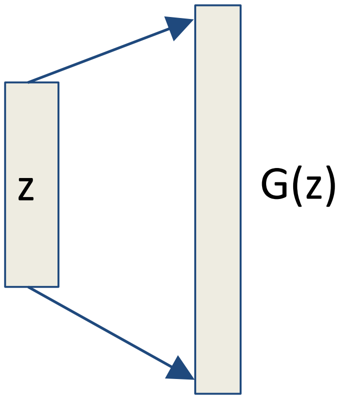
Generative models



- A generative model is a magical black box that takes a vector **z** in R^k and produces a vector **G(z)** in R^n

- A new way to parametrize high-dimensional distributions.
- (vs Graphical Models, HMMs etc)

Generative models

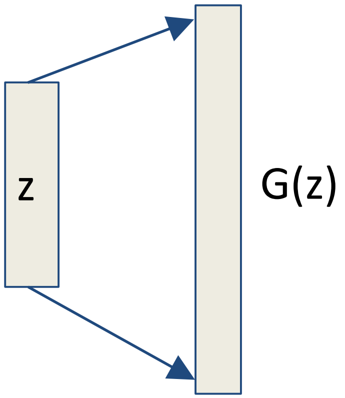


- A generative model is a magical black box that takes a vector **z** in R^k and produces a vector **G(z)** in R^n

- Differentiable Compression:
- k=100, n=64 × 64×3 ≈ 13000

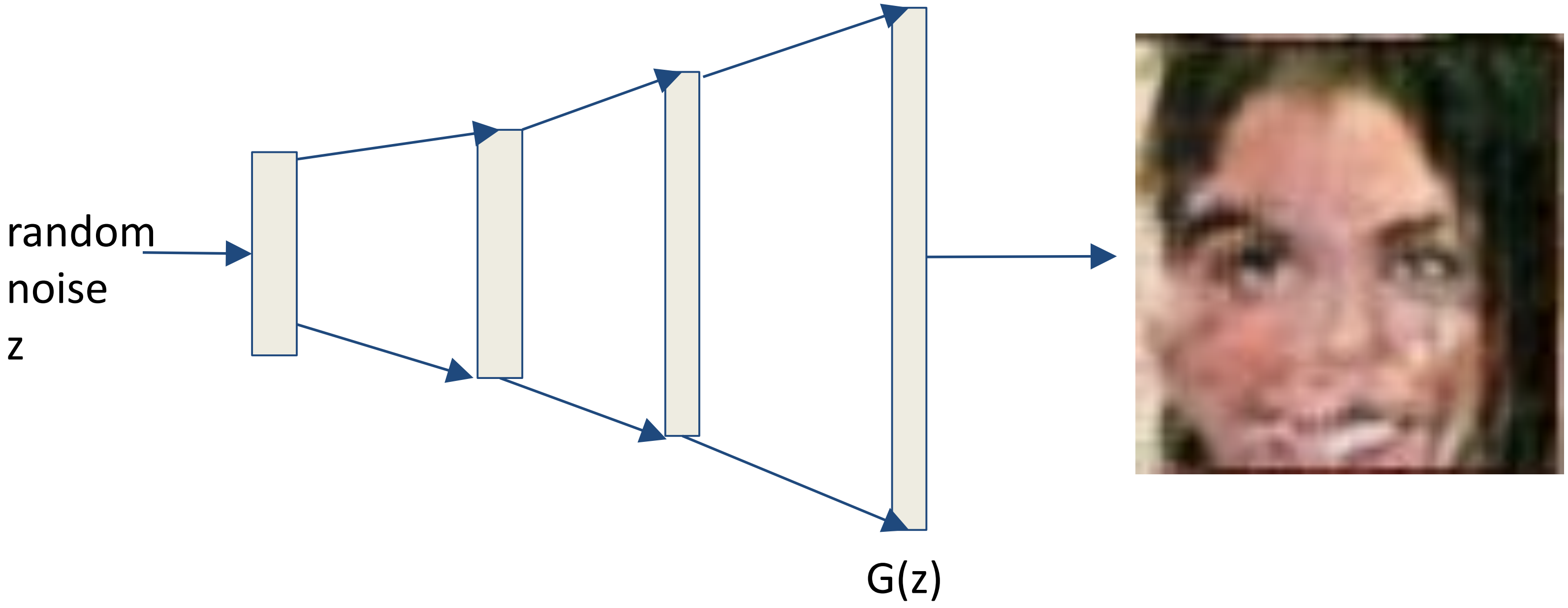
- It can be trained to take gaussian iid z and produce samples of complicated distributions, like human faces.

Generative models

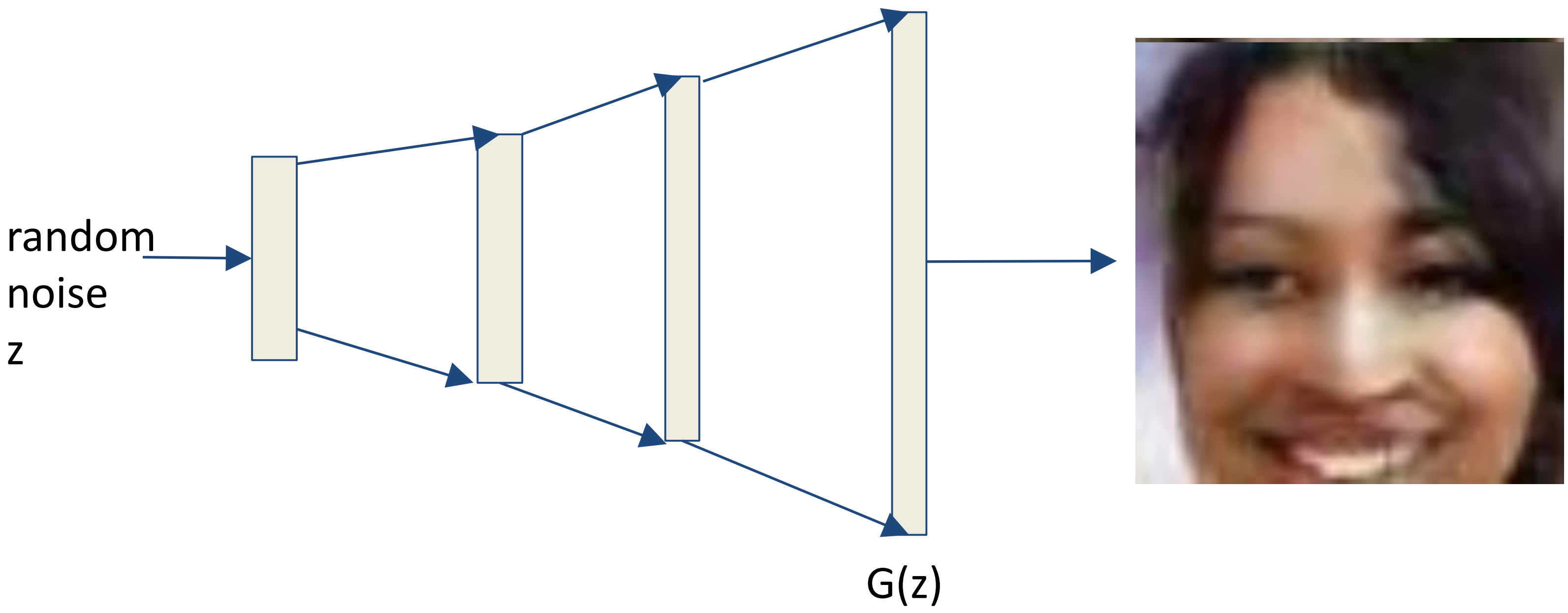


- A generative model is a magical black box that takes a vector **z** in R^k and produces a vector **G(z)** in R^n
- k=100, n=64 × 64×3 ≈ 13000
- It can be trained to take gaussian iid z and produce samples of complicated distributions, like human faces.
- Training can be done using standard ML (Autoencoders/VAE) or using adversarial training (GANs)
- It is a **differentiable** function

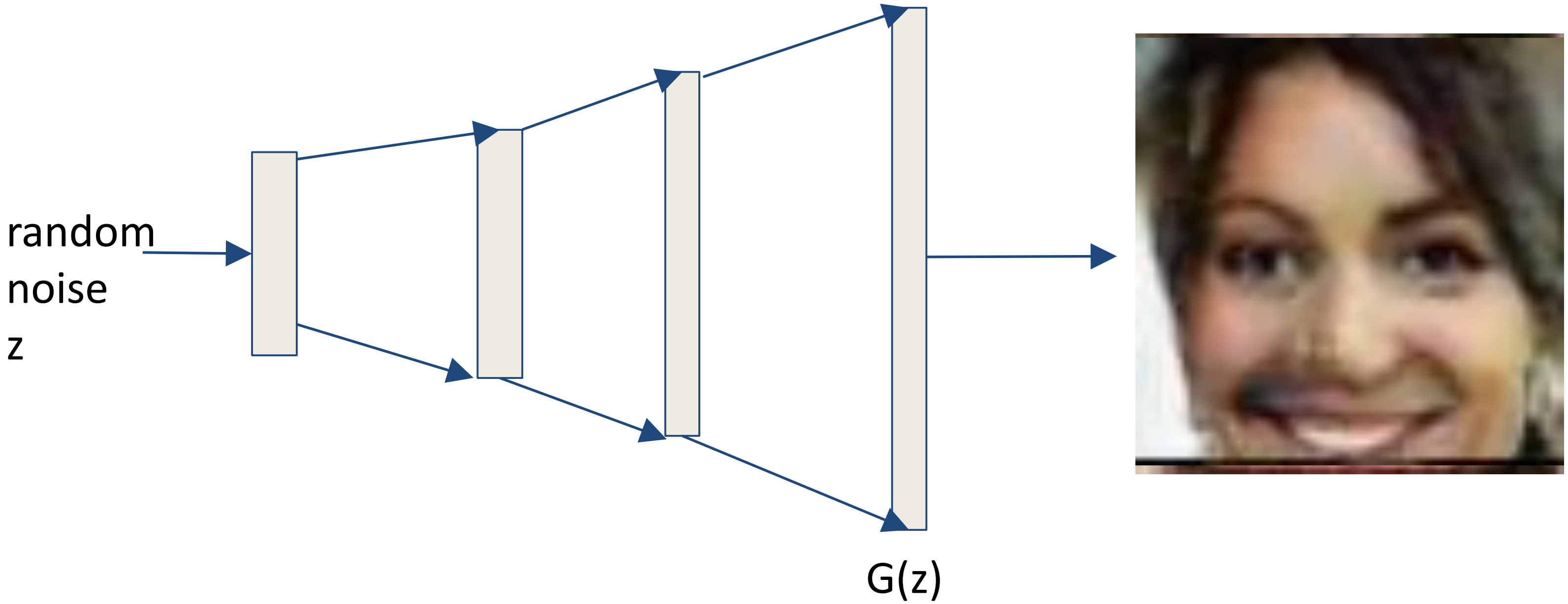
How training a GAN looks like



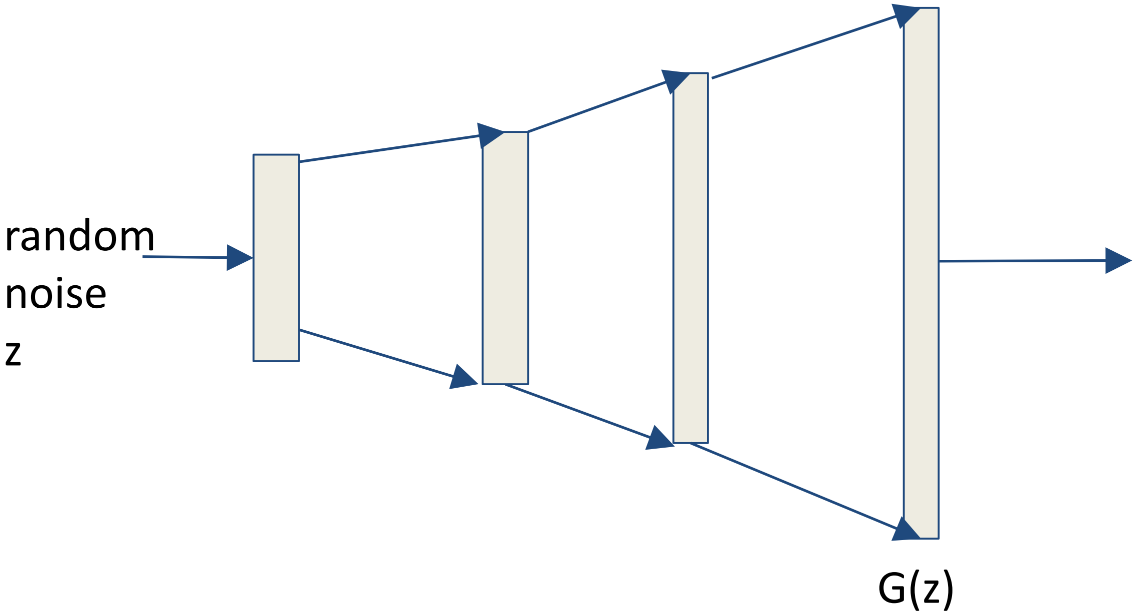
How training a GAN looks like



How training a GAN looks like

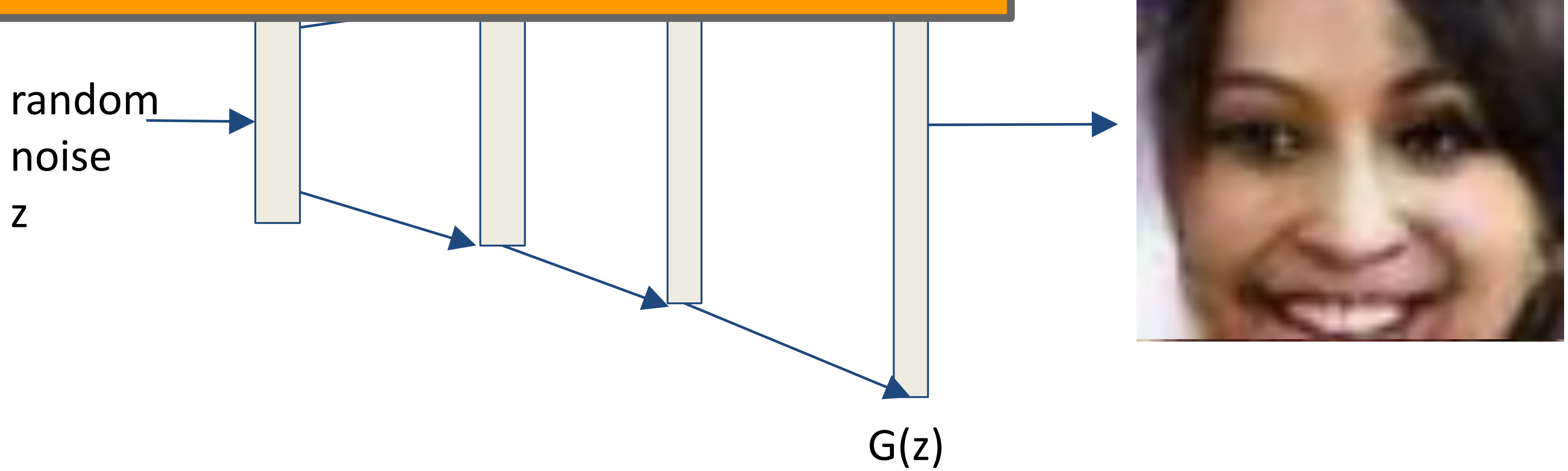


How training a GAN looks like

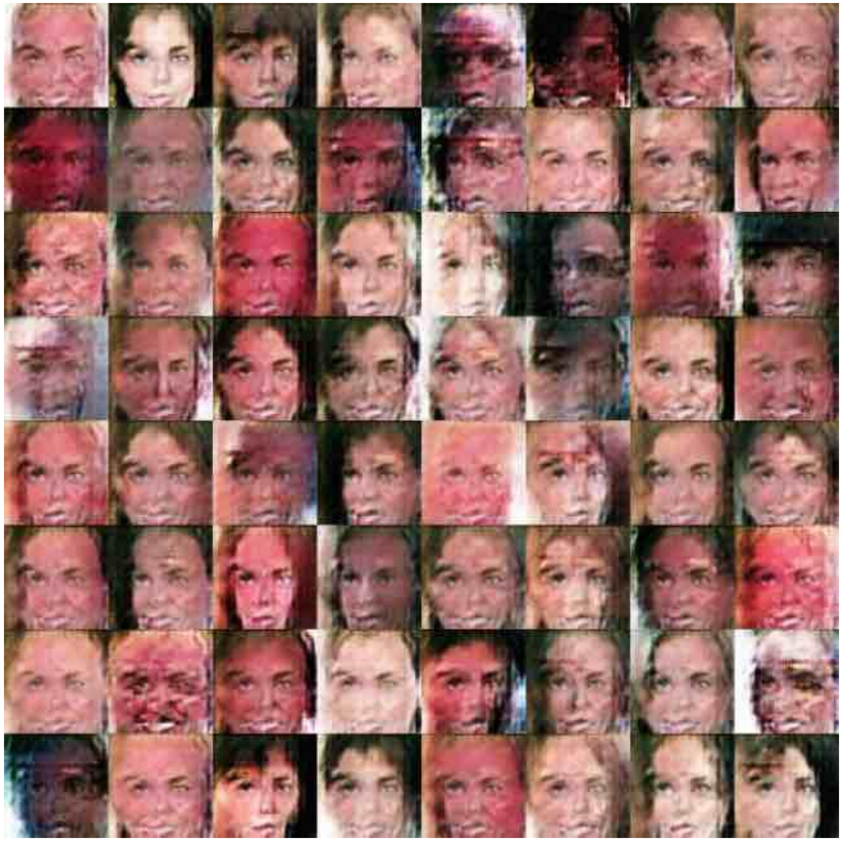


Any Resemblance
to Actual Persons,
Living or Dead,
is Purely Coincidental

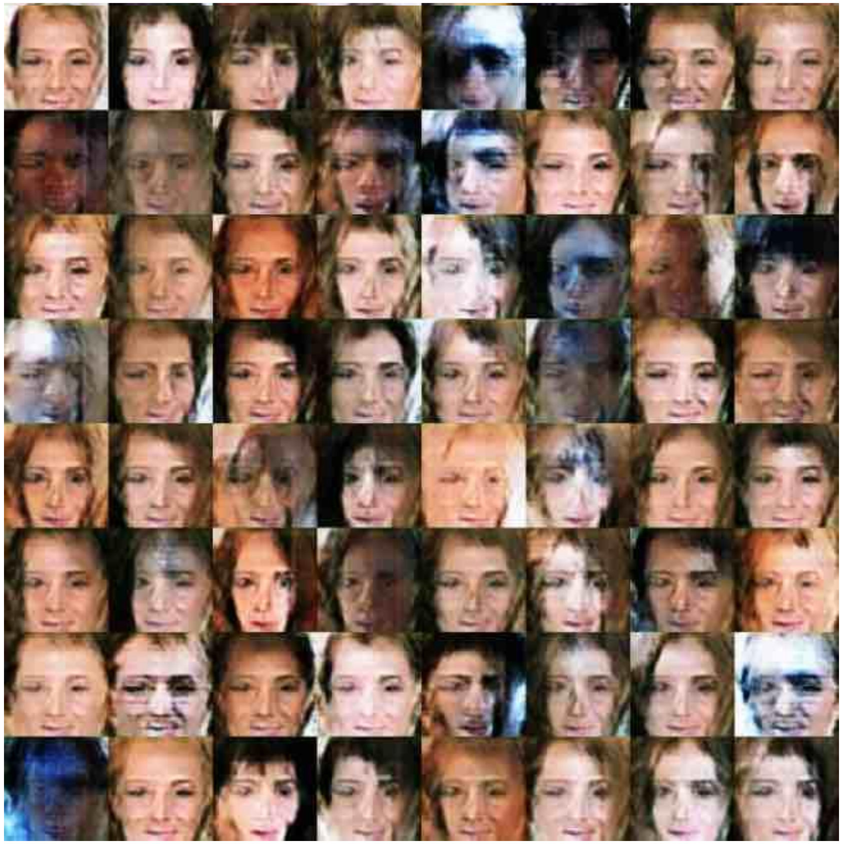
N looks like



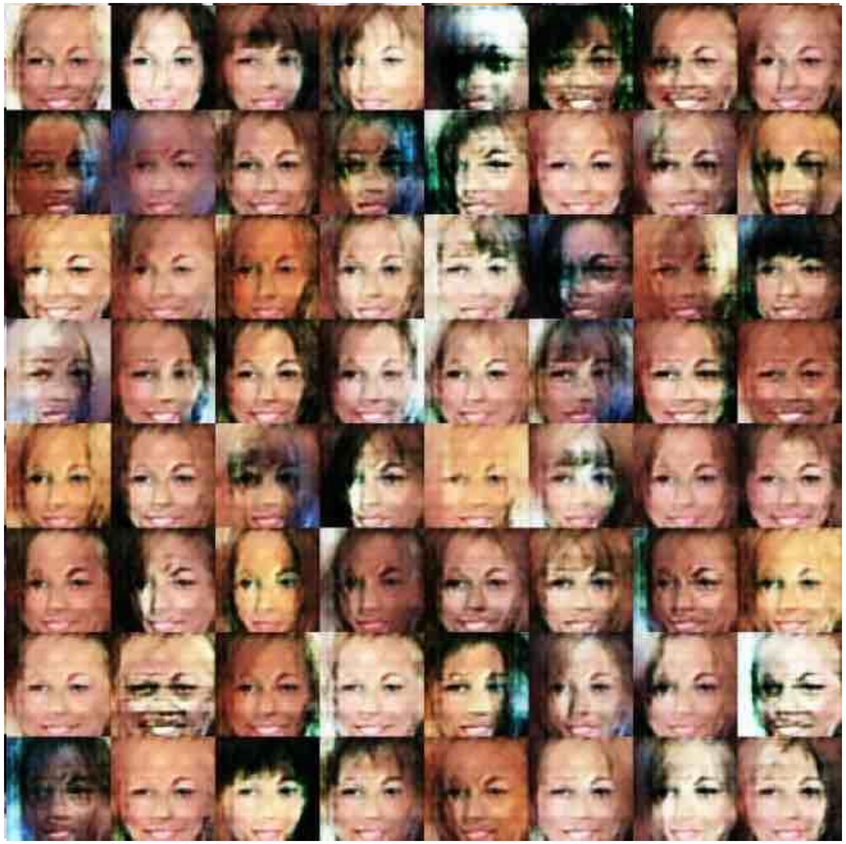
Adversarial Training



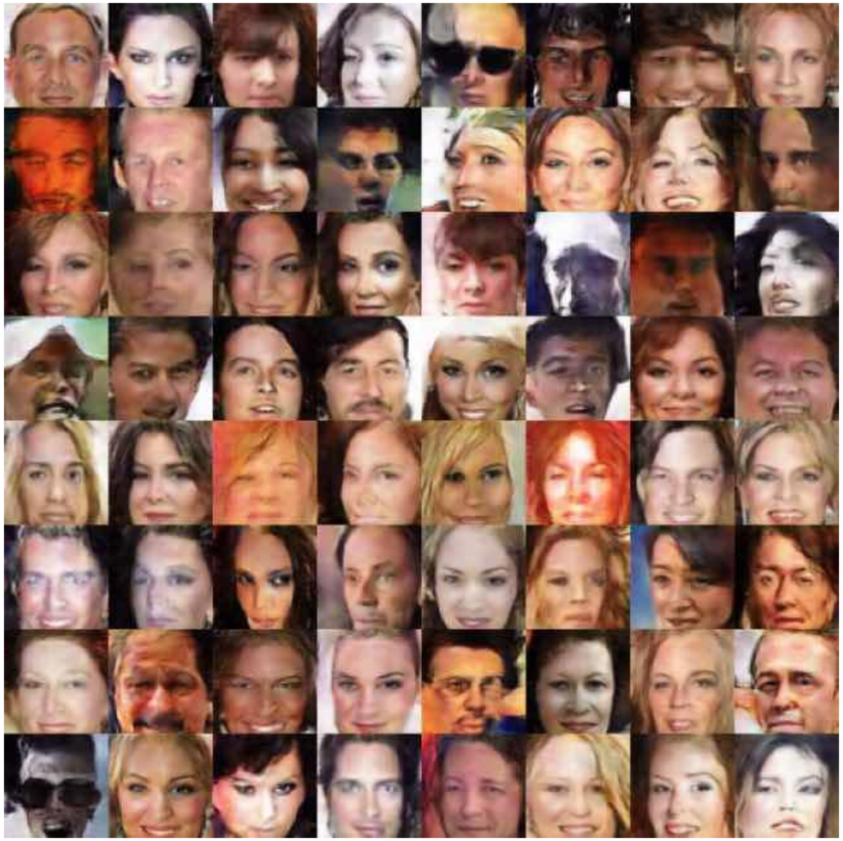
Adversarial Training



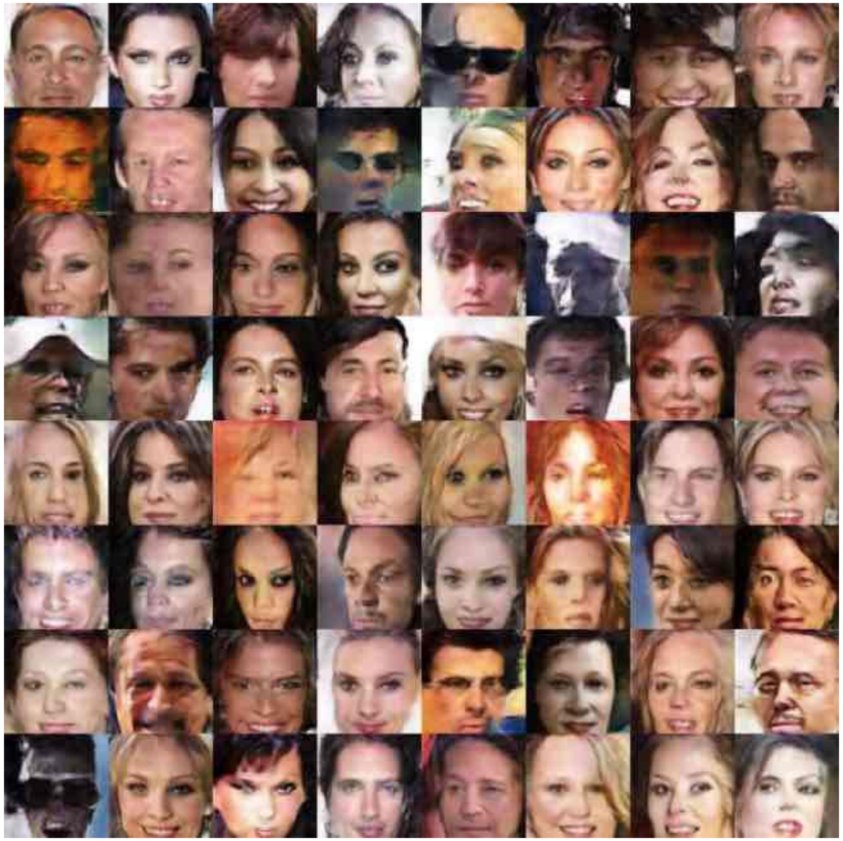
Adversarial Training



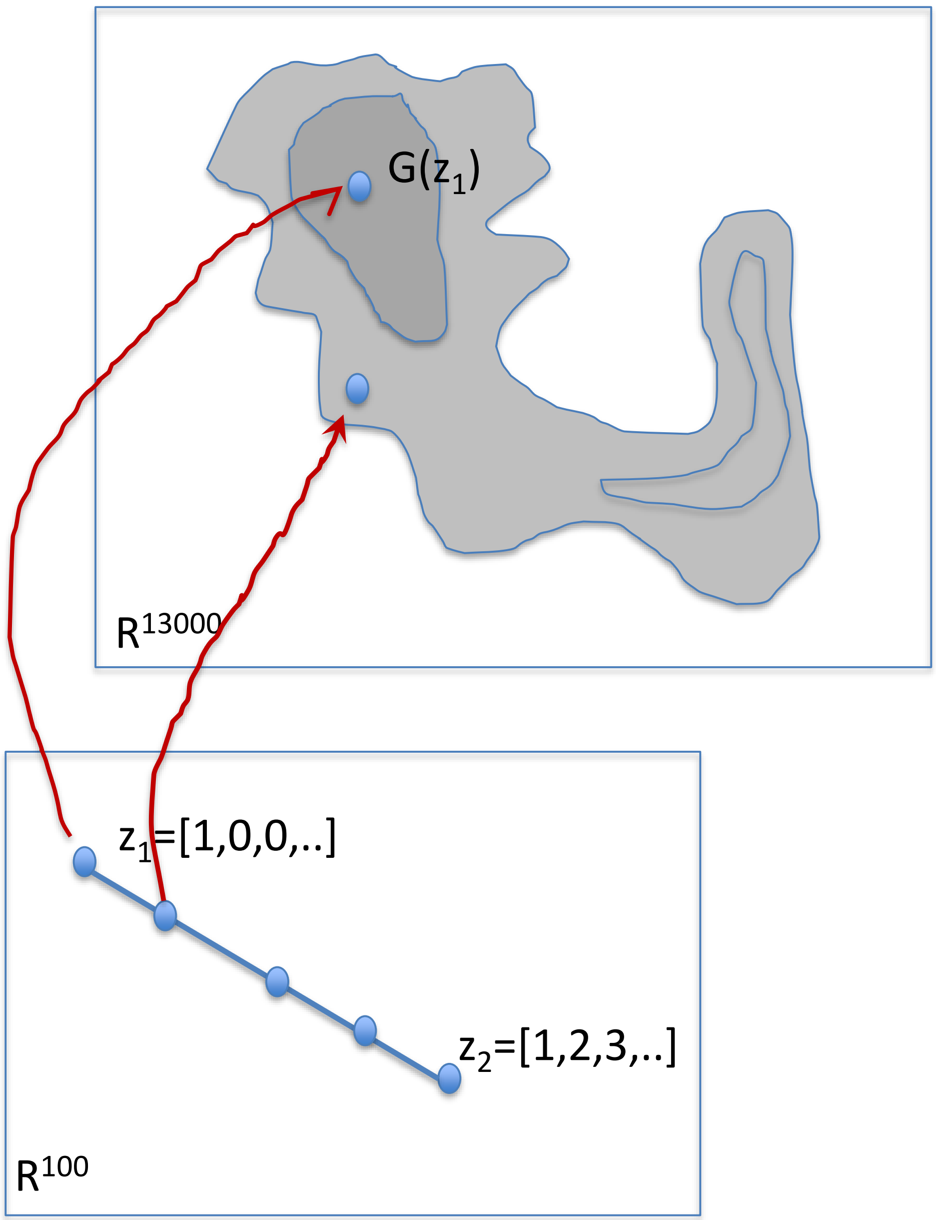
Adversarial Training



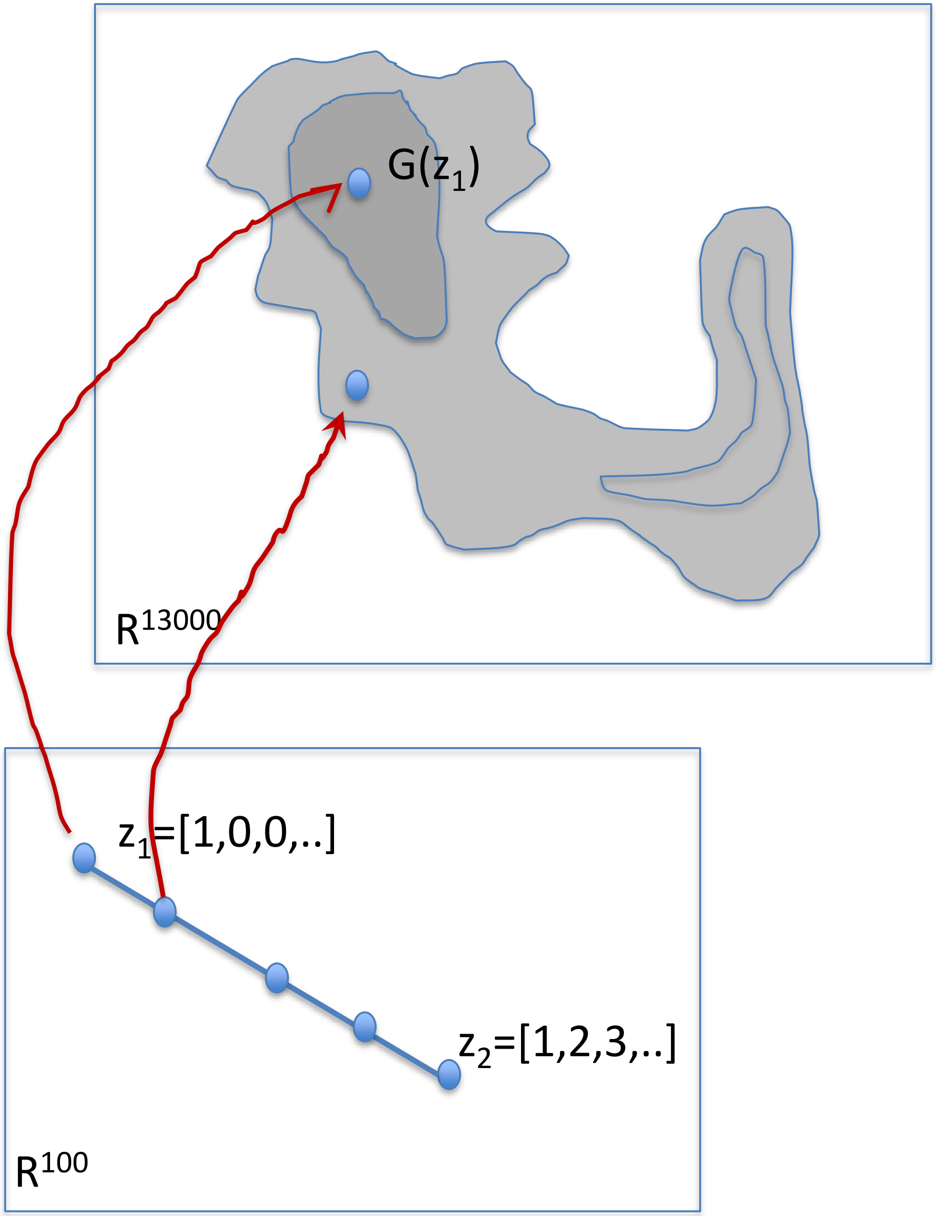
Adversarial Training



You can travel in z space too



You can travel in z space too

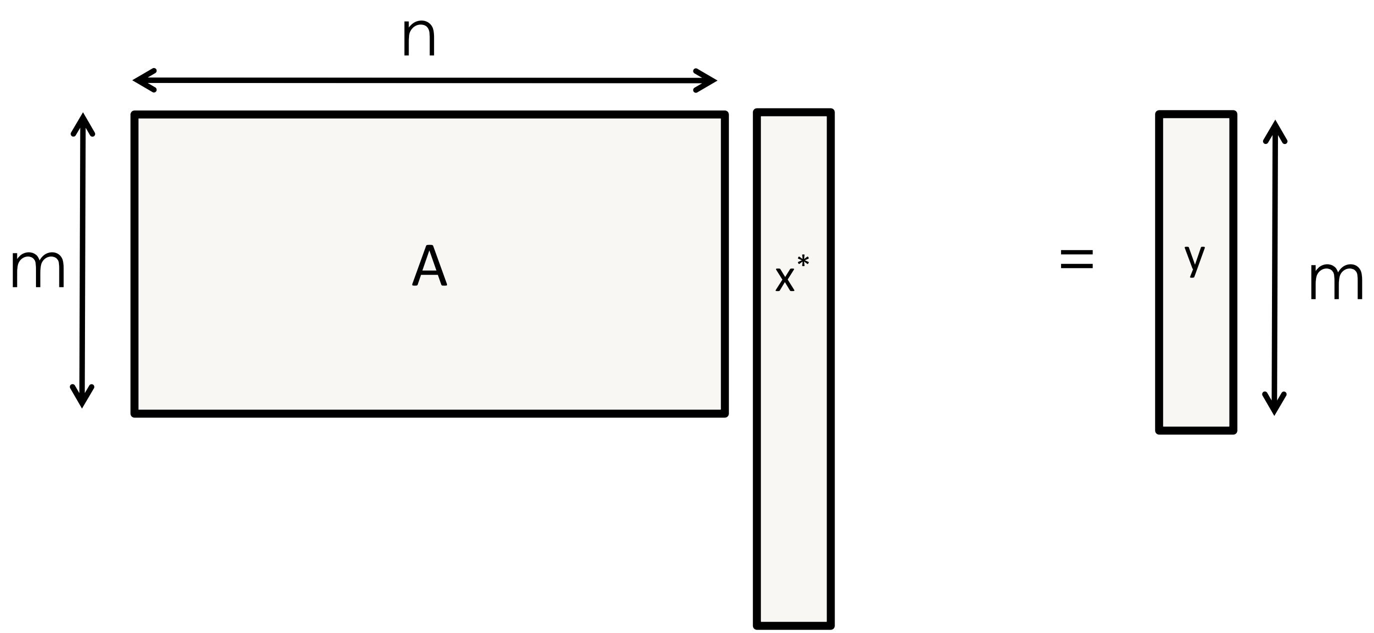


BEGANs produce amazing images

Ok, Modern deep generative models produce amazing pictures.

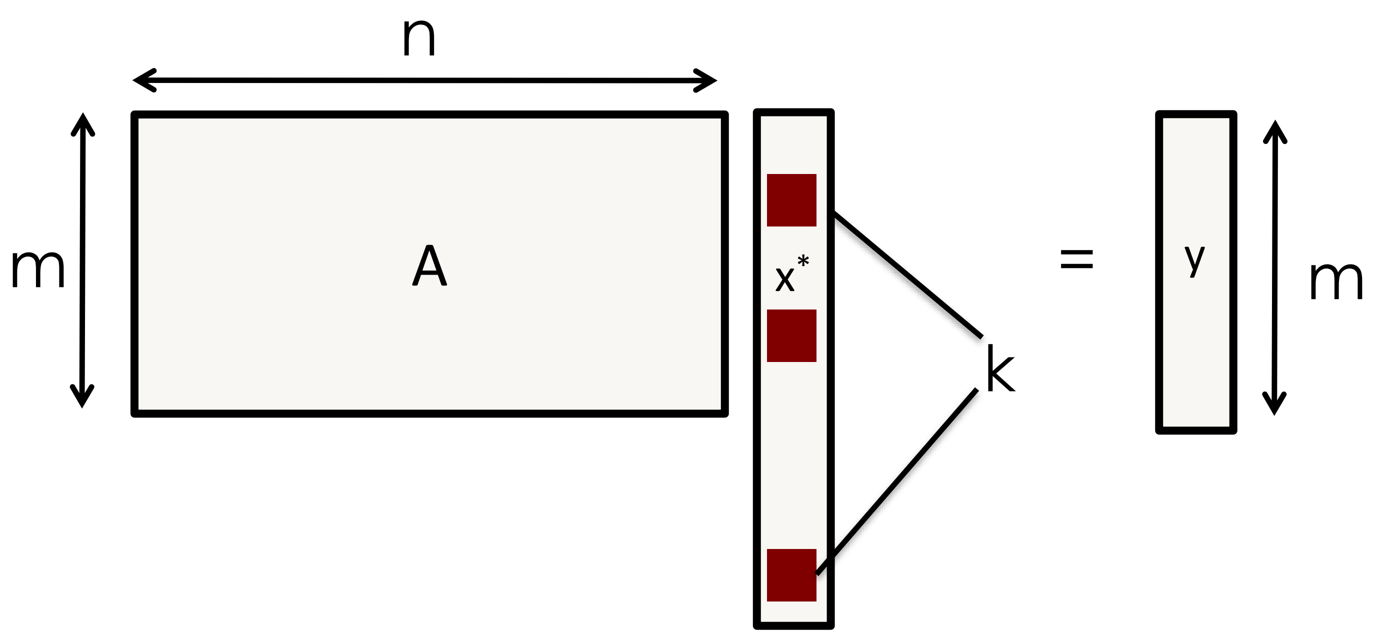
But what can we do with them ?

Compressed sensing



- You observe y = A x^* , *x in R^n , y in R^m, n>m*
- i.e. m (noisy) linear observations of an unknown vector y in R^n
- **Goal:** Recover x^* from y
- **ill-posed:** there are many possible x* that explain the measurements since we have m linear equations with n unknowns.
- **High-dimensional statistics**: Number of parameters n > number of samples m
- **Must make some assumption**: that x* is **natural** in some sense.

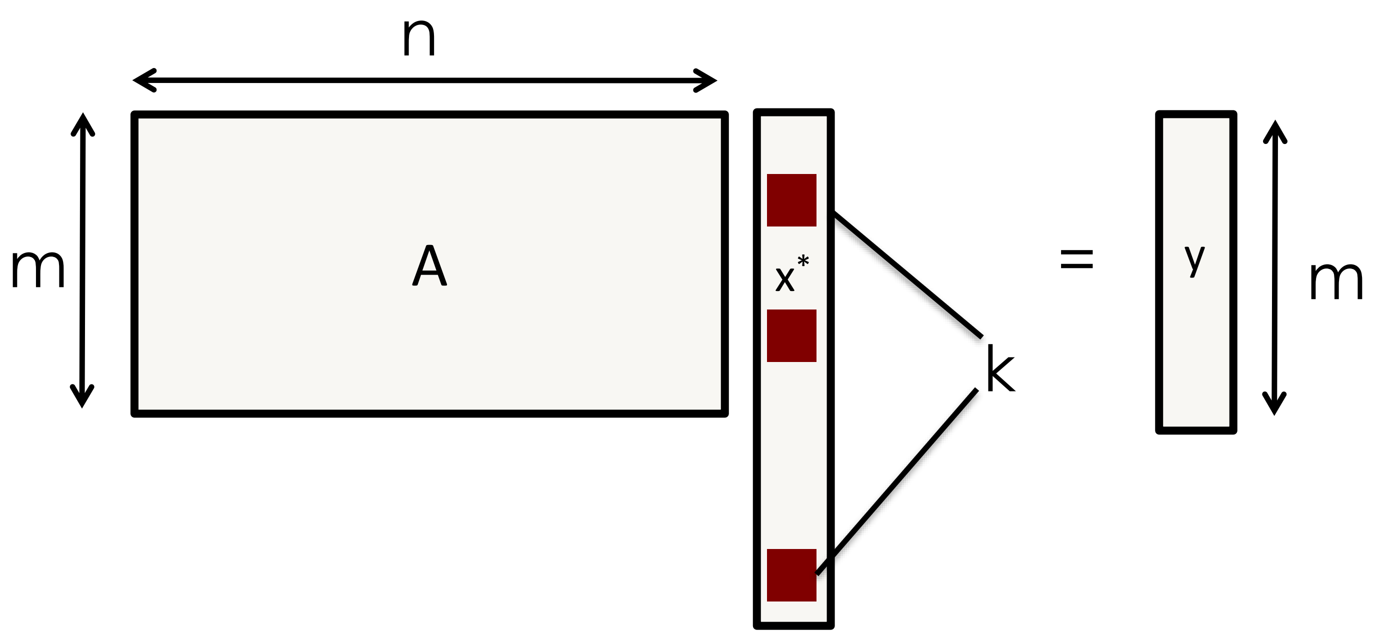
Compressed sensing



- Standard assumption: x is k-sparse. $|x|_0 = k$
- Noiseless CompSensing optimal recovery problem:

$$\min_{x:\, Ax=y} ||x||_0$$

Compressed sensing



- Standard assumption: x is k-sparse. $|x|_0 = k$
- Noiseless CompSensing optimal recovery problem:

$$\min_{x:\, Ax=y} ||x||_0 \longrightarrow \min_{x:\, Ax=y} ||x||_1$$

- NP-hard
- Relax to solving Basis pursuit
- Under what conditions is the relaxation tight?

Compressed sensing

$$\min_{x:\, Ax=y} ||x||_0 \longrightarrow \min_{x:\, Ax=y} ||x||_1$$

$$\downarrow \qquad\qquad\qquad\qquad \downarrow$$

$$x^* \qquad\qquad\qquad\qquad x^1$$

- **Question:** for which measurement matrices A, is $x^* = x^1$?

- [Donoho,Candes and Tao, RombergCandesTao]
- If A satisfies (RIP/REC/NSP) condition then $x^* = x^1$

- Also: If A is created random iid N(0, 1/m) with
- m = k log n/k then whp it will satisfy the RIP/REC condition.

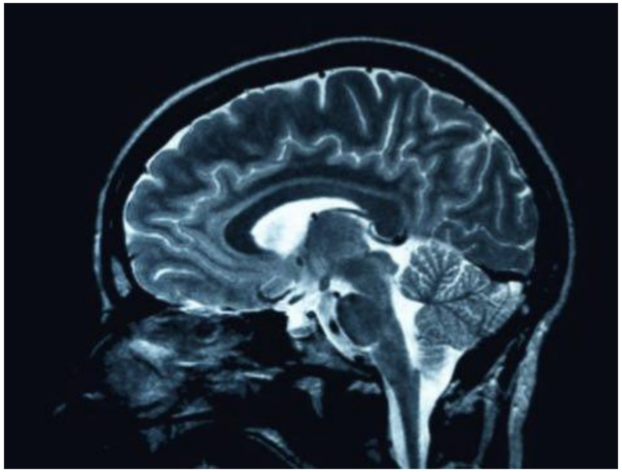
- So: A random measurement matrix A with enough measurements **suffices for the LP relaxation to produce the exact unknown sparse vector x^***

Sparsity in compressed sensing

- Q1: When do you want to recover some unknown vector by observing linear measurements on its entries?

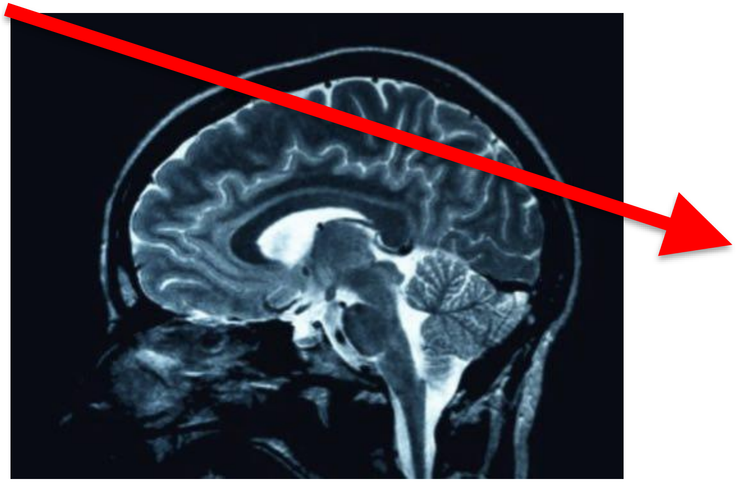
Sparsity in compressed sensing

- Q1: When do you want to recover some unknown vector by observing linear measurements on its entries?



Sparsity in compressed sensing

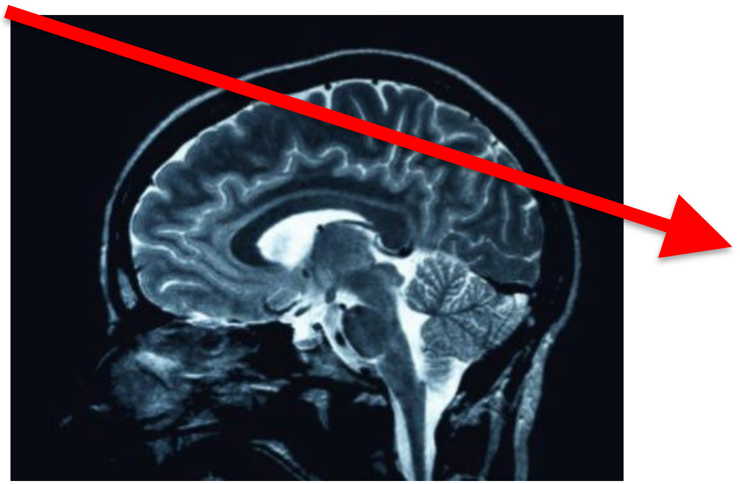
- Q1: When do you want to recover some unknown vector by observing linear measurements on its entries?



sum over values of pixels

Sparsity in compressed sensing

- Q1: When do you want to recover some unknown vector by observing linear measurements on its entries?



- Real images are not sparse (except night-time sky).
- But they can be sparse in a known basis , i.e. $x'' = D x^*$
- D can be DCT or Wavelet basis.

~~Sparsity~~ in compressed sensing

- Q1: Wh
  obser

of

- Re
- But
- D car

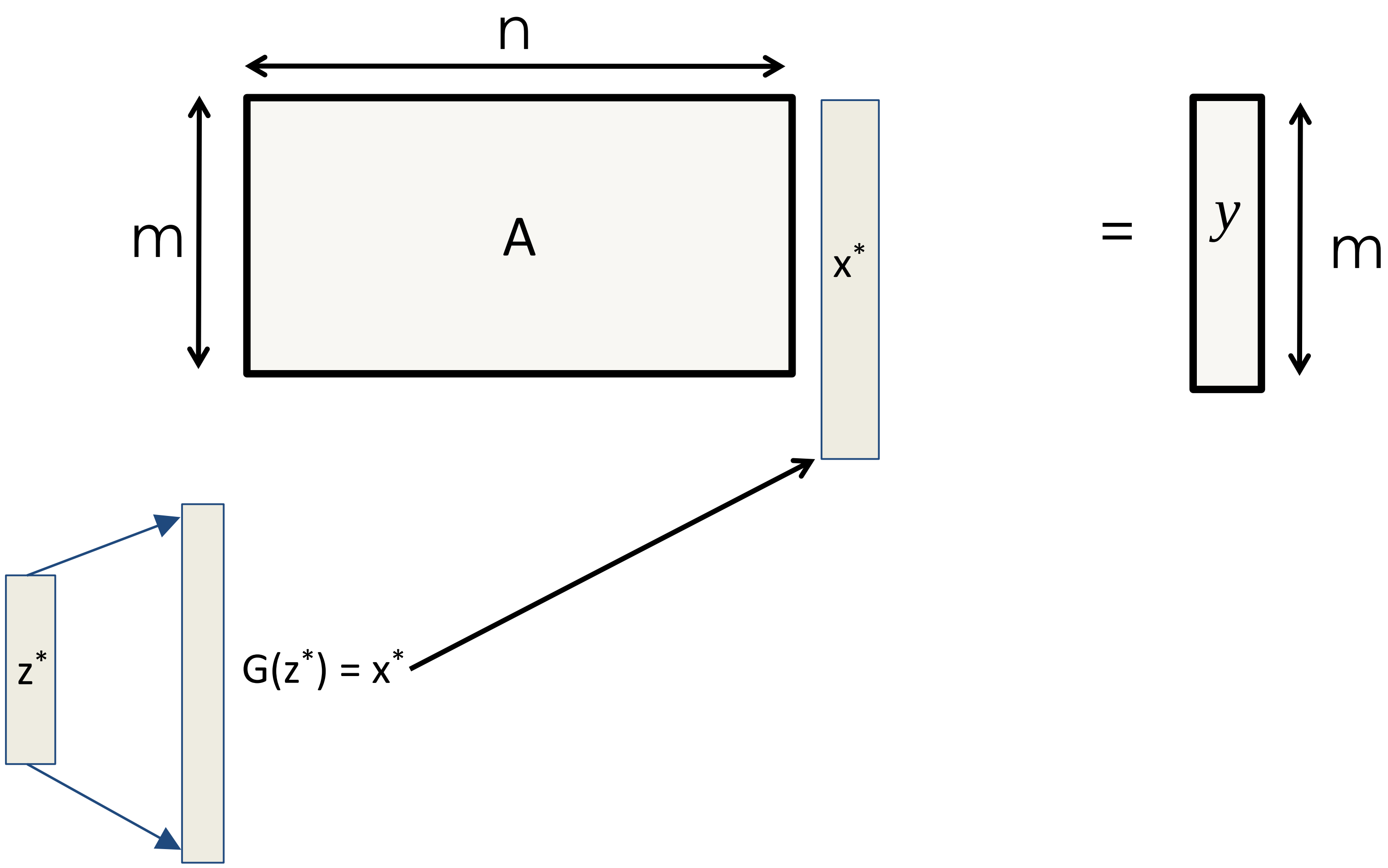
1. Sparsity in a basis is a decent model for natural images

2. But now **we have much better data driven models for natural images**: VAEs and GANs

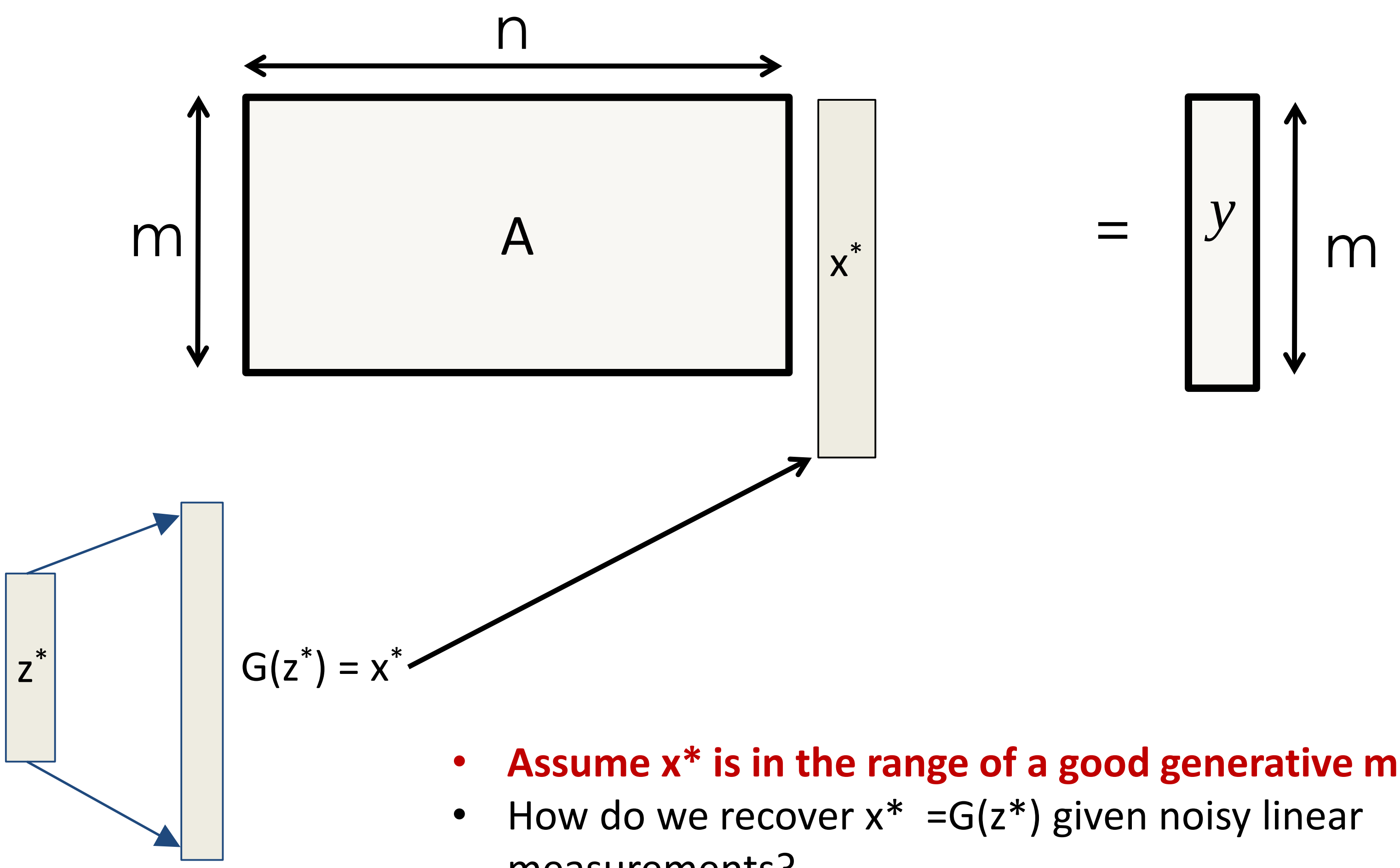
3. **Idea: Take sparsity out of compressed sensing**. Replace with GAN

4. Ok. But how to do that?

Generative model

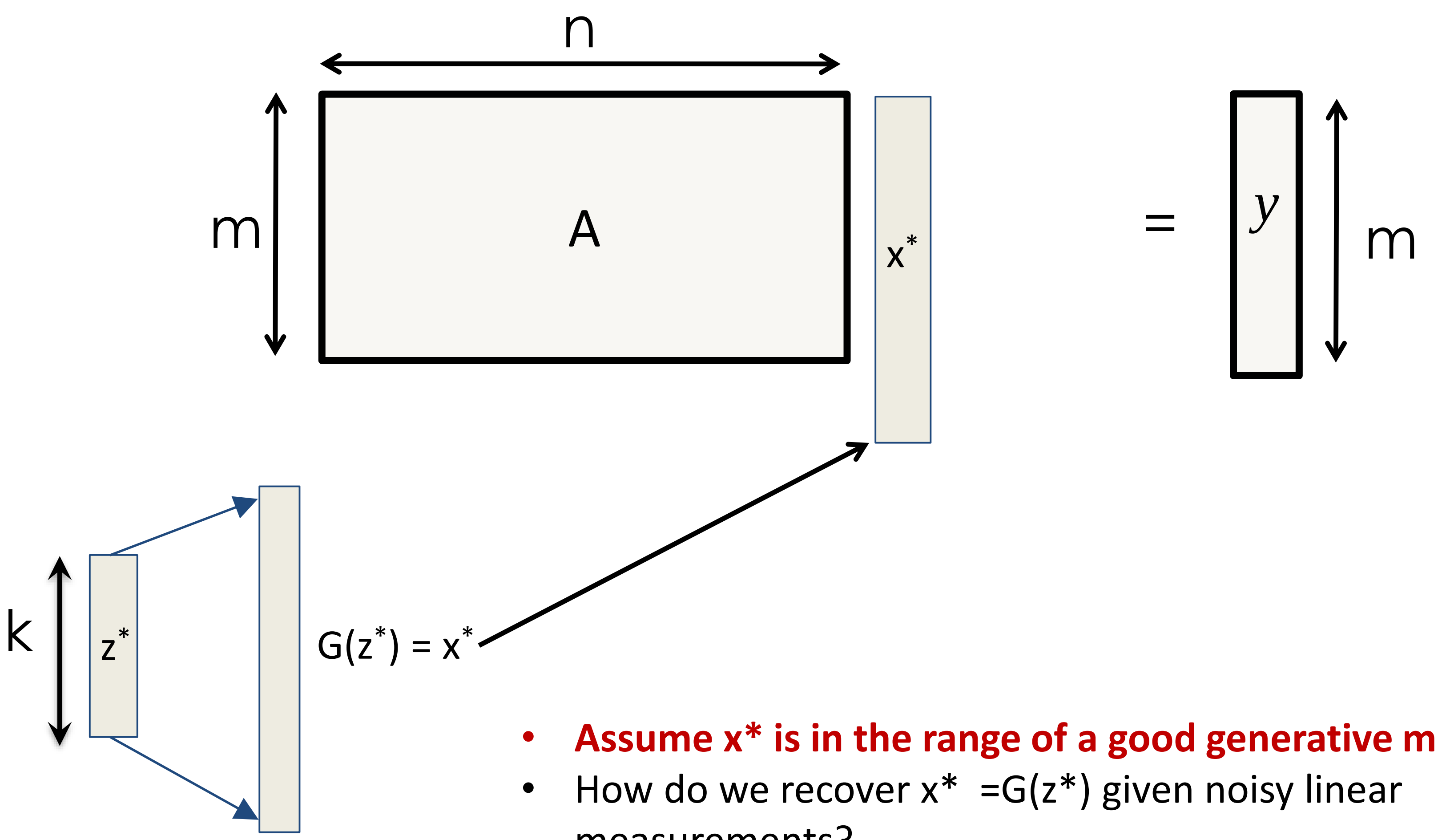


Generative model



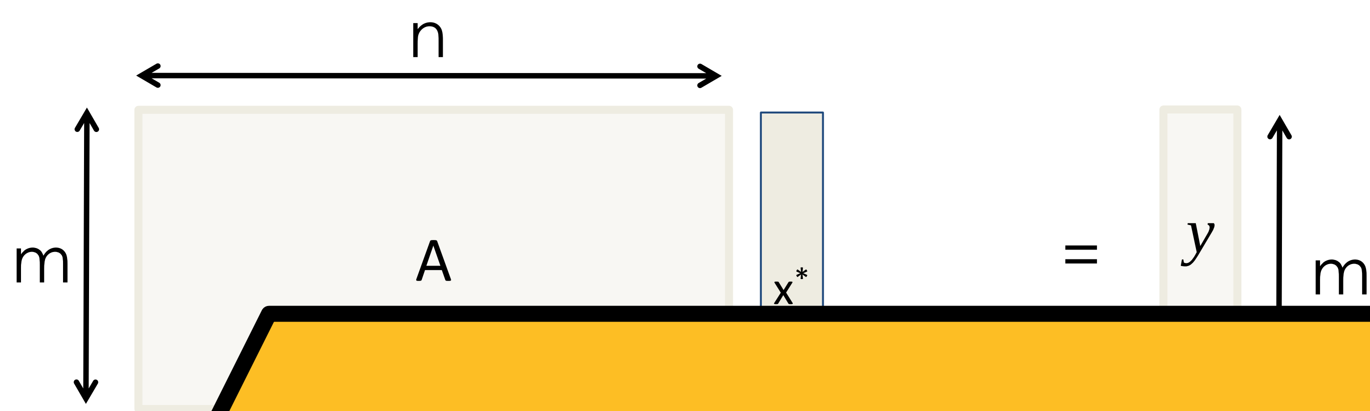
- **Assume x* is in the range of a good generative model G(z).**
- How do we recover x* =G(z*) given noisy linear measurements?
- $y = A x^* + \eta$
- ***What happened to sparsity k ?***

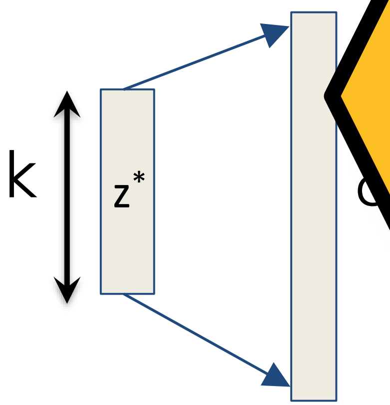
Generative model



- **Assume x* is in the range of a good generative model G(z).**
- How do we recover x* =G(z*) given noisy linear measurements?
- y = A x* + η

Generative model



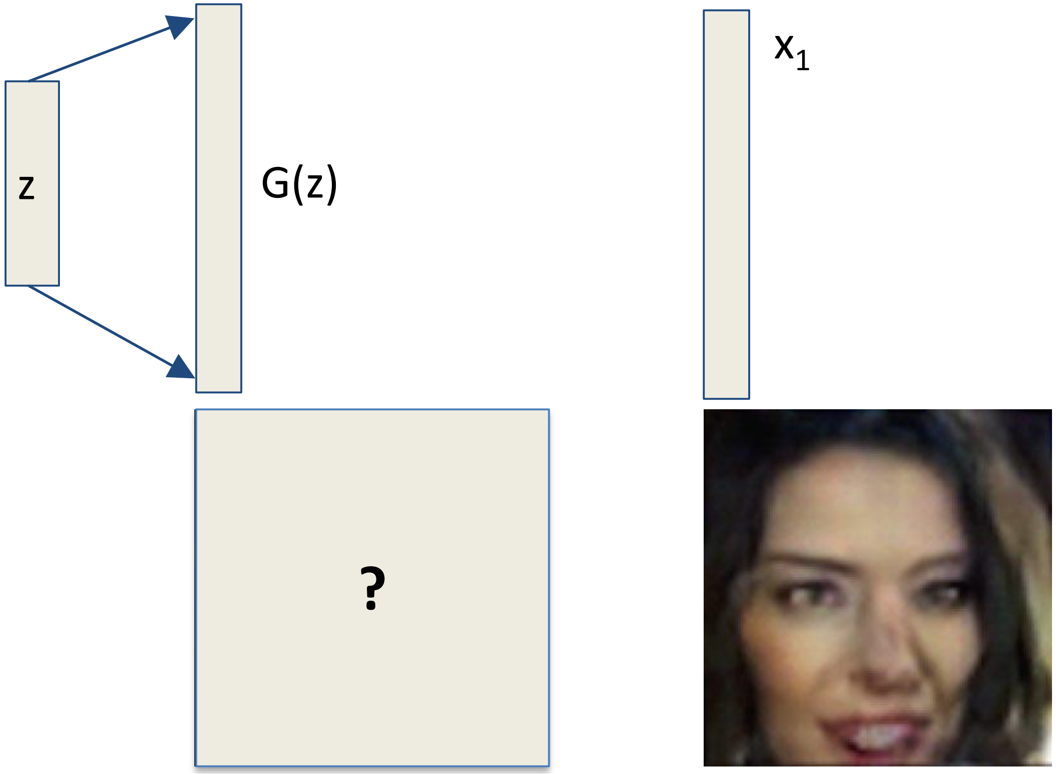


Ok, you are replacing sparsity with a neural network.
To recover before, we were using Lasso.
What is the **recovery algorithm** now?

G(z).

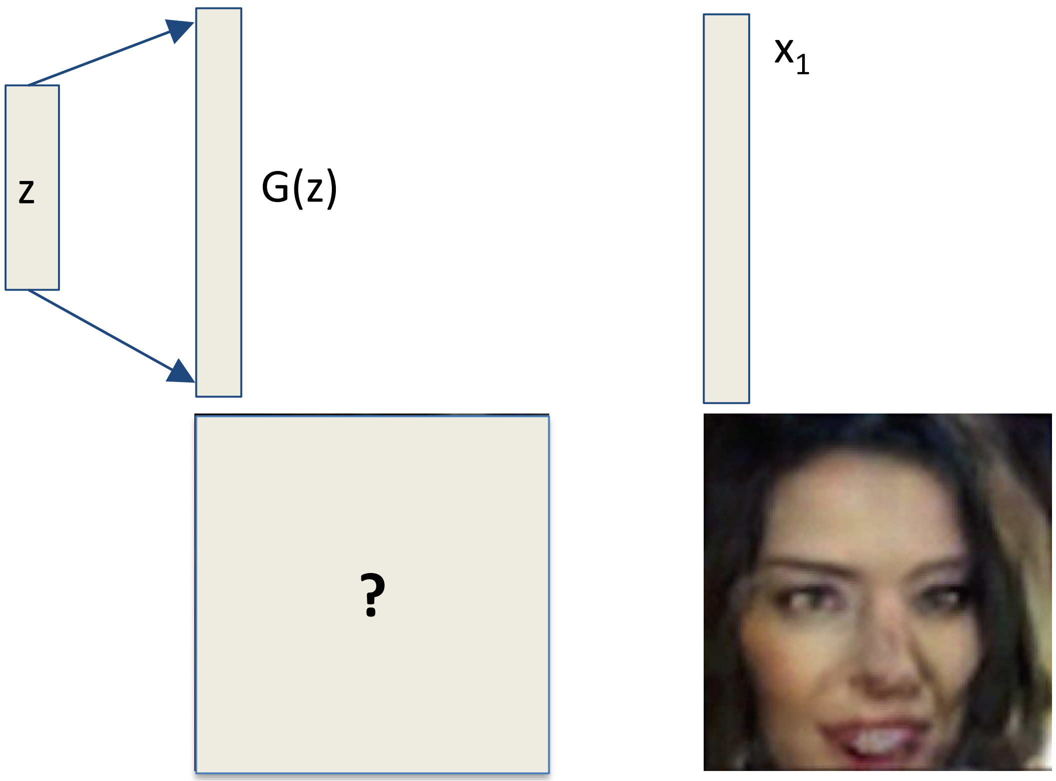
- $y = A x^* + \eta$

Recovery algorithm: Step 1: Inverting a GAN



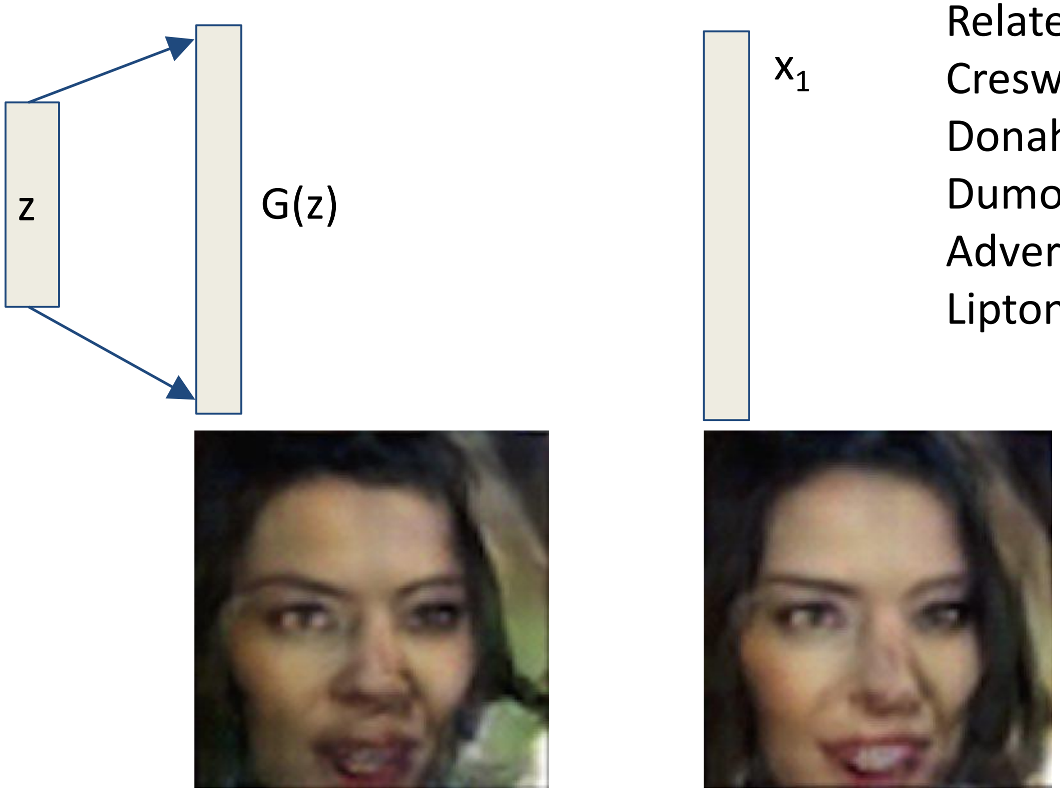
- Given a target image x_1 how do we invert the GAN, i.e. find a z_1 such that $G(z_1)$ is very close to x_1 ?

Recovery algorithm: Step 1: Inverting a GAN



- Given a target image x_1 how do we invert the GAN, i.e. find a z_1 such that $G(z_1)$ is very close to x_1 ?
- Just define a loss $J(z) = || G(z) – x_1||$
- Do gradient descent on z (network weights fixed).

Recovery algorithm: Step 1: Inverting a GAN



Related work :

Creswell and Bharath (2016)

Donahue, Krahenbuhl,Trevor 2016

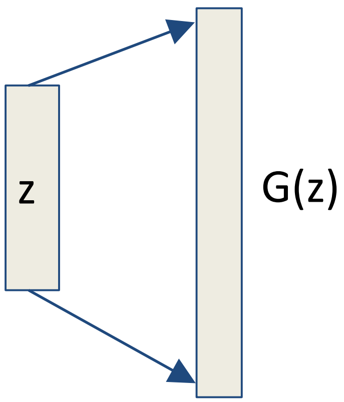
Dumoulin et al.

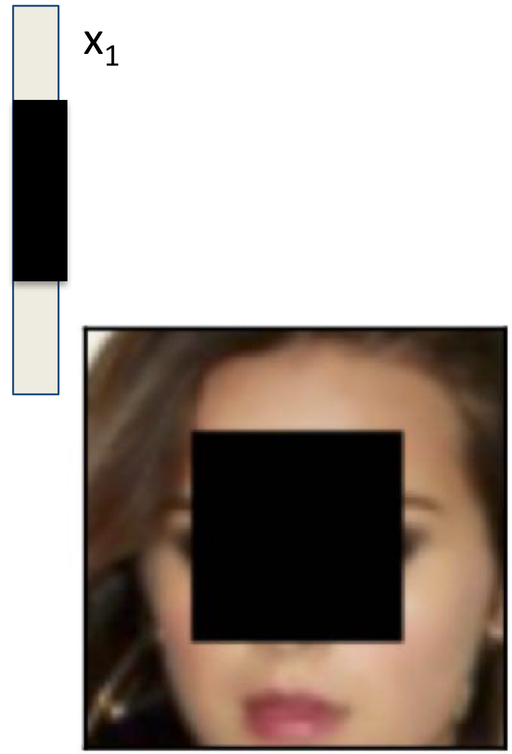
Adversarially learned Inference

Lipton and Tripathi 2017

- Given a target image x_1 how do we invert the GAN, i.e. find a z_1 such that $G(z_1)$ is very close to x_1 ?
- Just define a loss $J(z) = || G(z) – x_1||$
- Do gradient descent on z (network weights fixed).

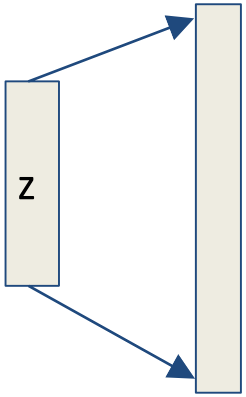
Recovery algorithm: Step 2: Inpainting

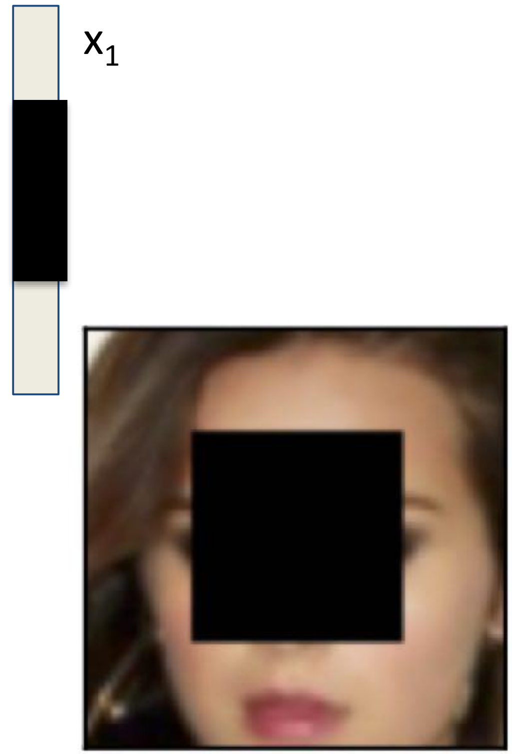




- Given a target image x_1 observe only some pixels.
- How do we invert the GAN now?

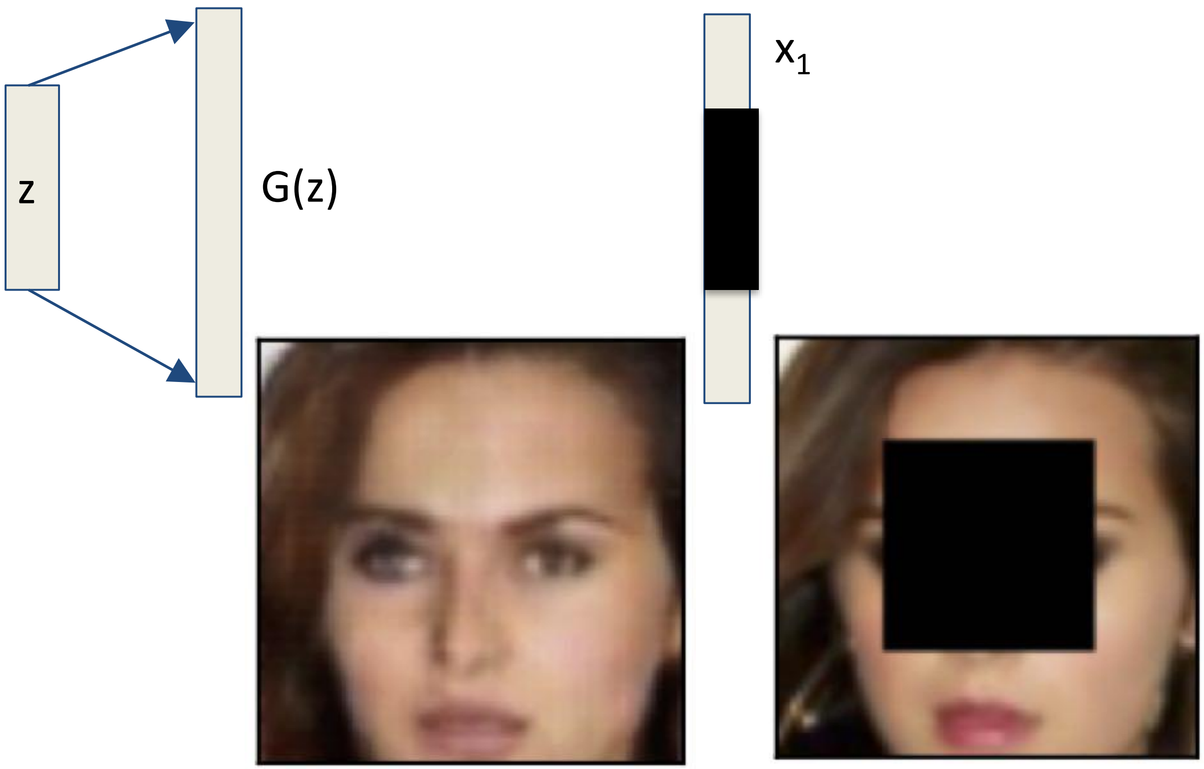
Recovery algorithm: Step 2: Inpainting





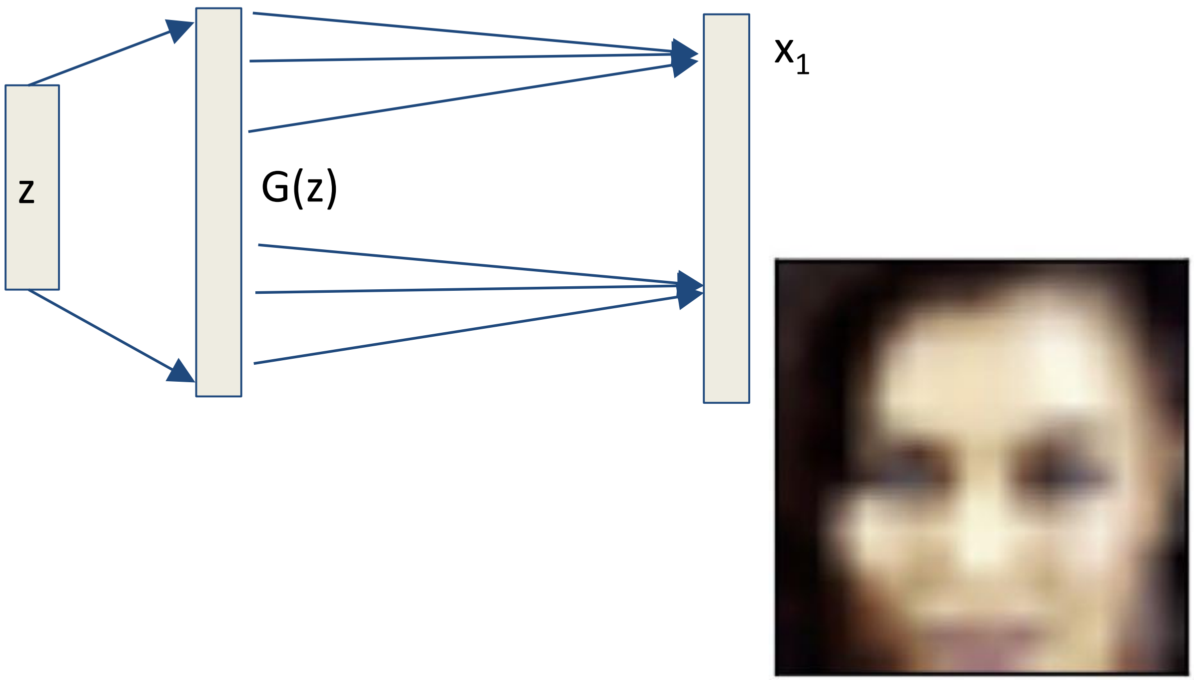
- Given a target image x_1 observe only some pixels.
- How do we invert the GAN, i.e. find a z_1 such that $G(z_1)$ is very close to x_1 **on the observed pixels**?
- Just define a loss $J(z) = || A\, G(z) - A\, x_1 ||$
- Do gradient descent on z (network weights fixed).

Recovery algorithm: Step 2: Inpainting



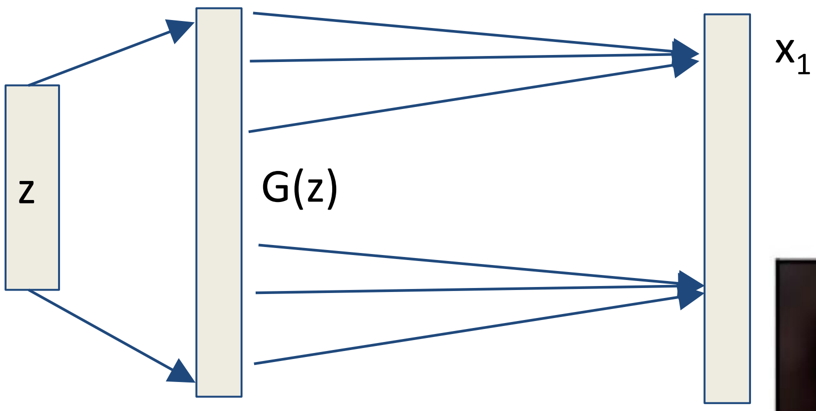
- Given a target image x_1 observe only some pixels.
- How do we invert the GAN, i.e. find a z_1 such that $G(z_1)$ is very close to x_1 **on the observed pixels**?
- Just define a loss $J(z) = || A\,G(z) - A\, x_1||$
- Do gradient descent on z (network weights fixed).

Recovery algorithm: Step 3: Super-resolution



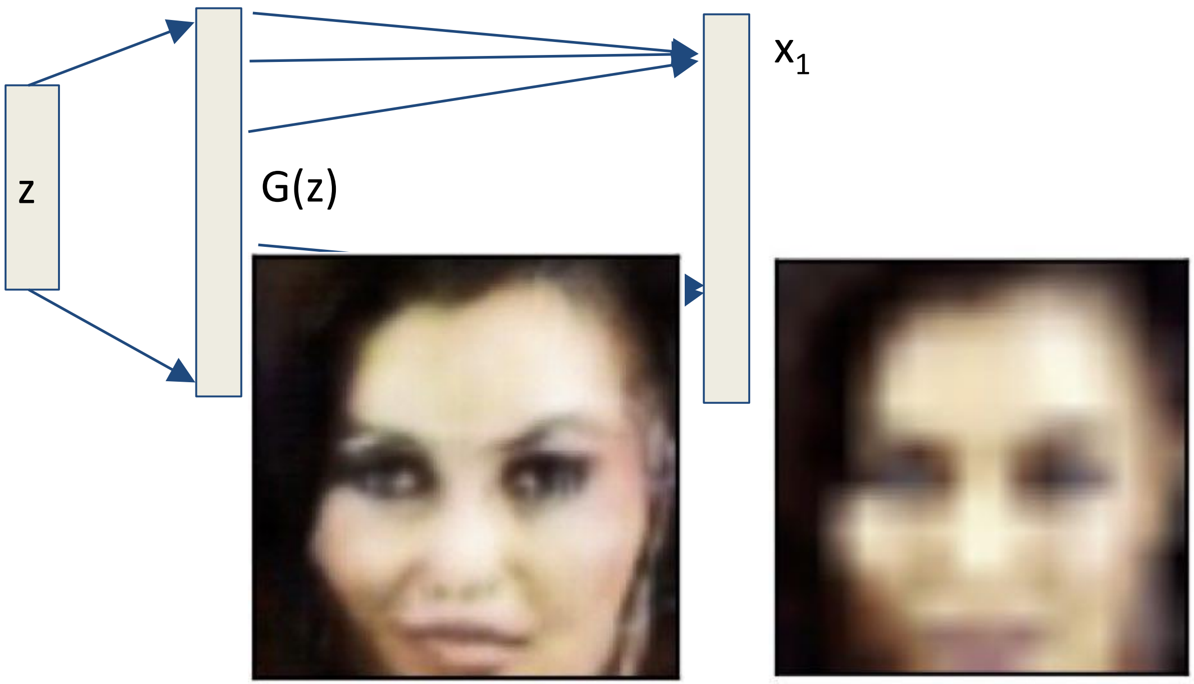
- Given a target image x_1 observe blurred pixels.
- How do we invert the GAN?

Recovery algorithm: Step 3: Super-resolution



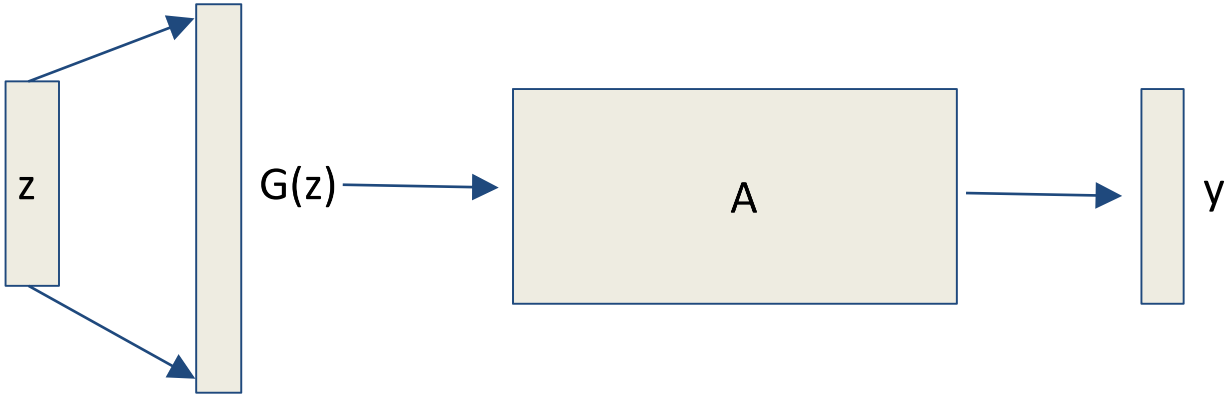
- Given a target image x_1 observe blurred pixels.
- How do we invert the GAN, i.e. find a z_1 such that $G(z_1)$ is very close to x_1 **After it has been blurred**?
- Just define a loss $J(z) = || A\, G(z) - A\, x_1 ||$
- Do gradient descent on z (network weights fixed).

Recovery algorithm: Step 3: Super-resolution



- Given a target image x_1 observe blurred pixels.
- How do we invert the GAN, i.e. find a z_1 such that $G(z_1)$ is very close to x_1 **After it has been blurred**?
- Just define a loss $J(z) = || A\,G(z) - A\ x_1||$
- Do gradient descent on z (network weights fixed).

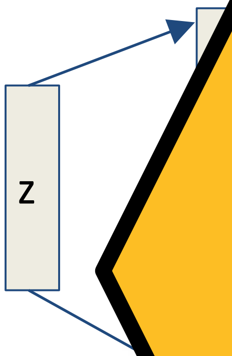
Recovery from linear measurements



$$\min_{z \in \mathbb{R}^k} ||y - AG(z)||_2$$

No Nebulous agenda

Recovery from linear measurements

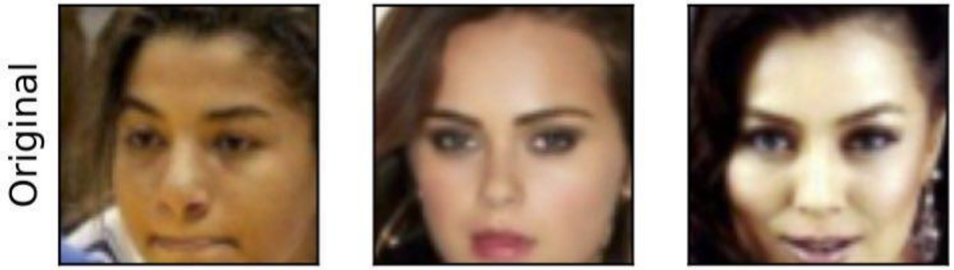


Our algorithm is:
Do gradient descent in z space to satisfy measurements.

Obtain useful gradients **through the measurements using backprop.**

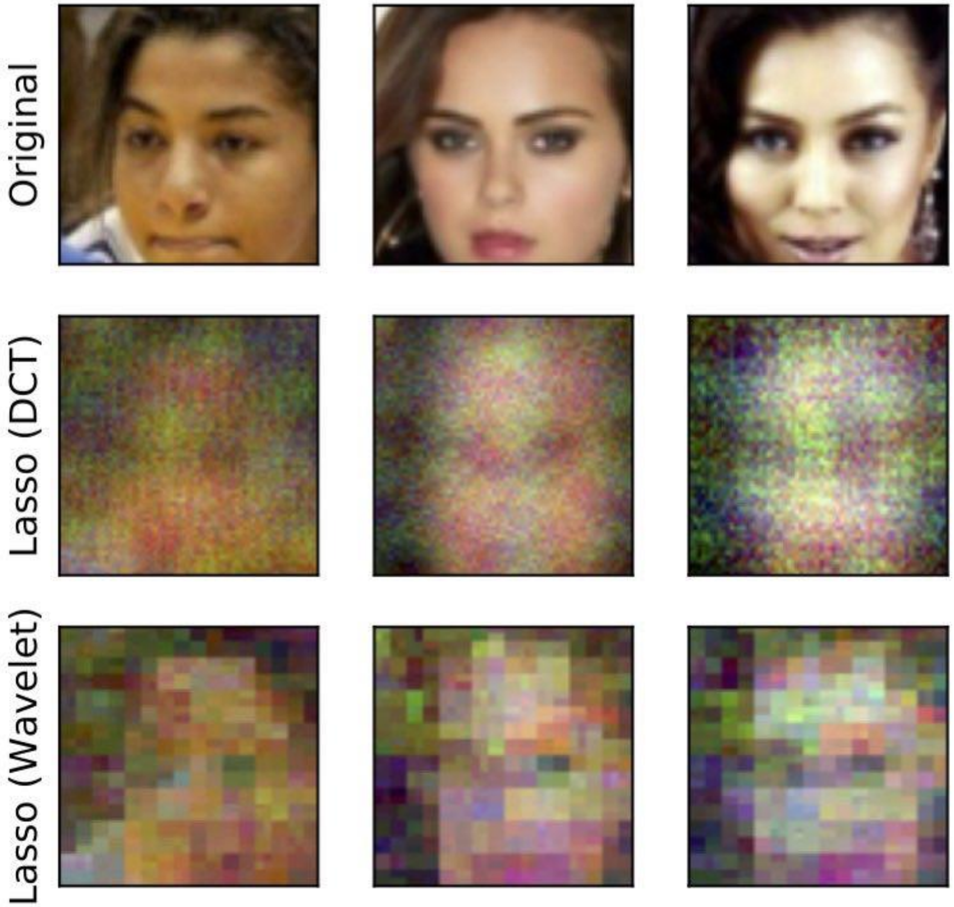
$$\min_{z \in \mathbb{R}^k} ||y - AG(z)||_2$$

Comparison to Lasso



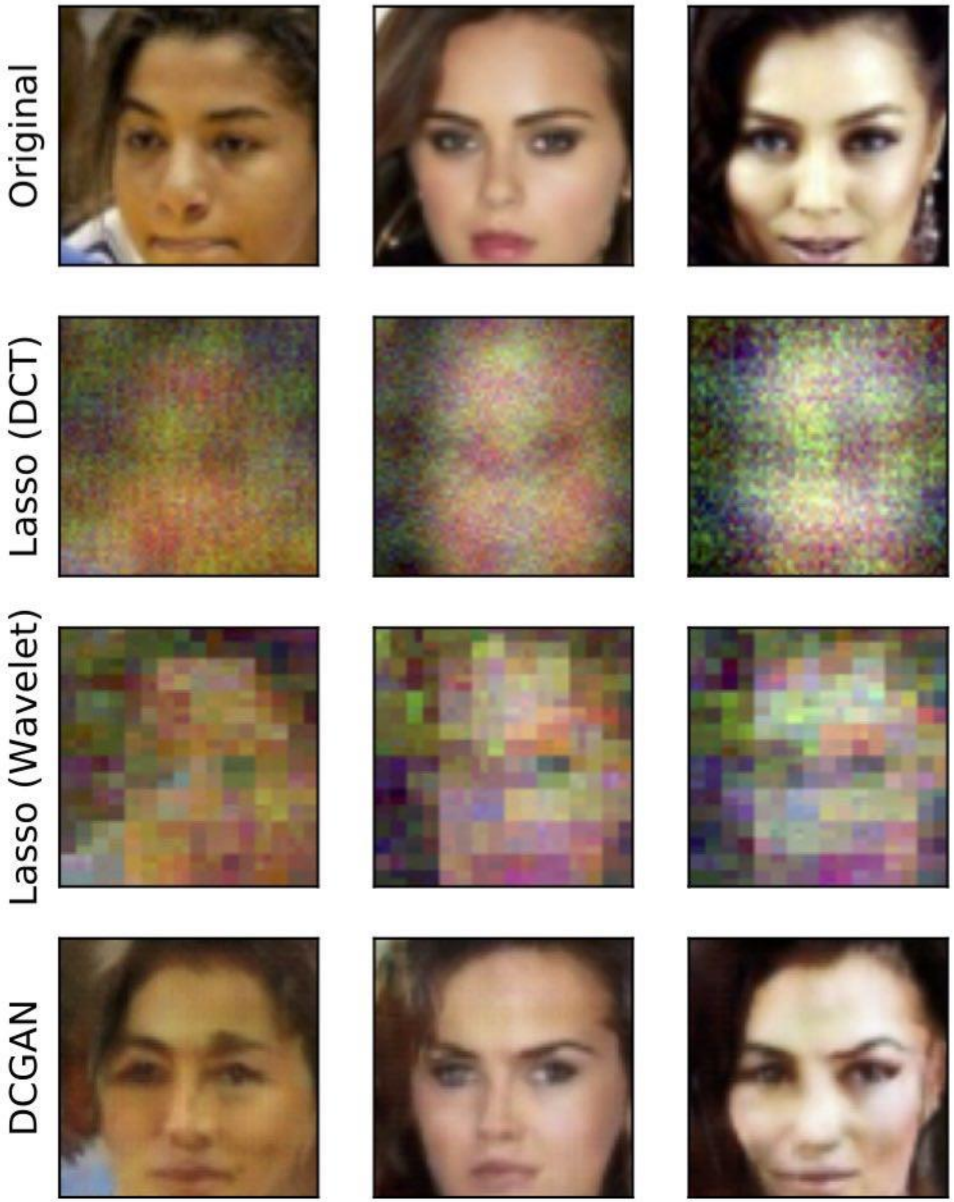
- m=500 random Gaussian measurements.
- n= 13k dimensional vectors.

Comparison to Lasso



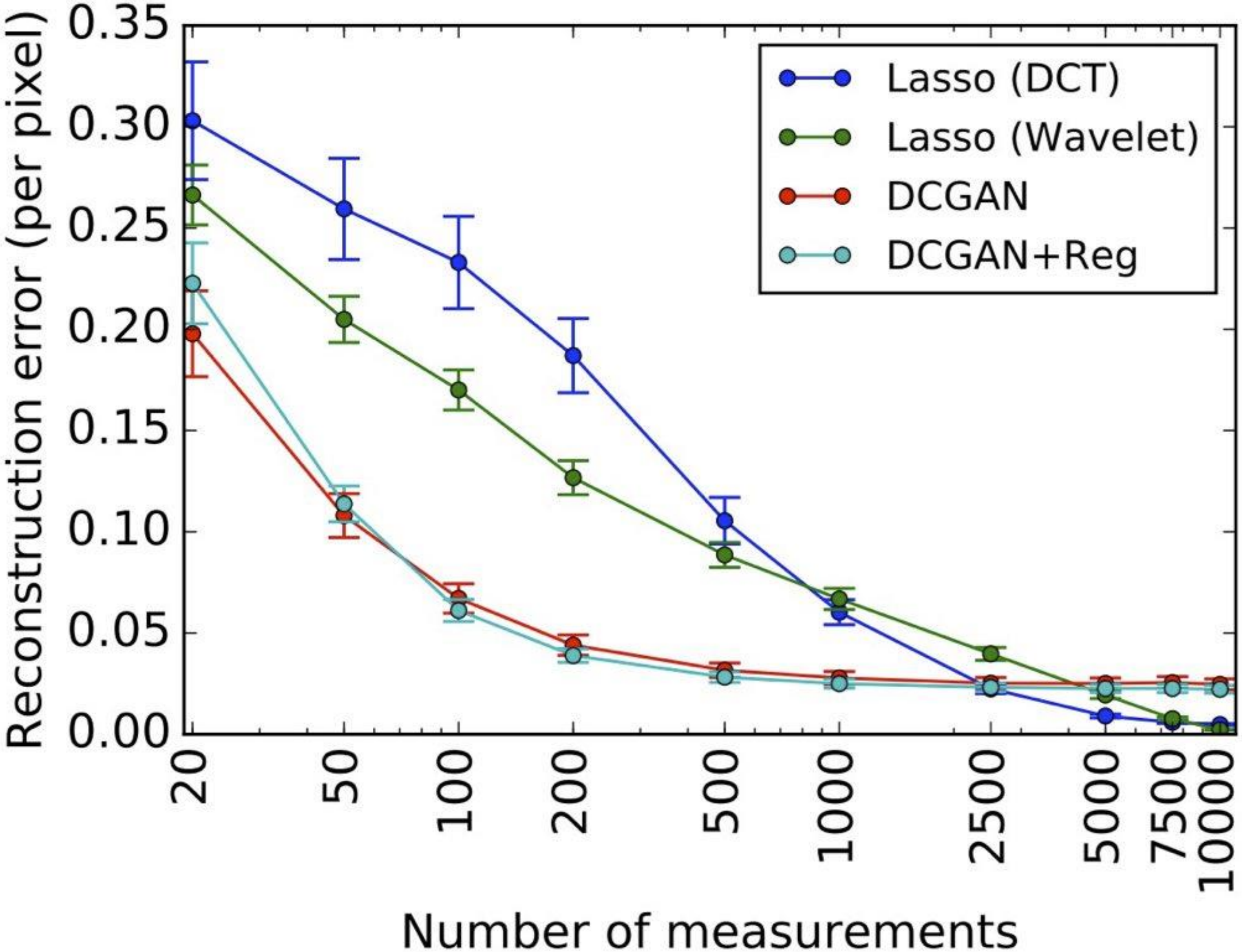
- m=500 random Gaussian measurements.
- n= 13k dimensional vectors.

Comparison to Lasso



- m=500 random Gaussian measurements.
- n= 13k dimensional vectors.

Comparison to Lasso



(b) Results on celebA

Related work

- Significant prior work on structure beyond sparsity
- **Model-based CS** (Baraniuk et al., Cevher et al., Hegde et al., Gilbert et al. , Duarte & Eldar)
- **Projections on Manifolds:**
- Baraniuk & Wakin (2009) Random projections of smooth manifolds. Eftekhari & Wakin (2015)
- **Deep network models:**
- Mousavi, Dasarathy, Baraniuk (here),
- Chang, J., Li, C., Poczos, B., Kumar, B., and Sankaranarayanan, ICCV 2017

Main results

- Let $y = Ax^* + \eta$
- Solve $\hat{z} = \min_z ||y - AG(z)||$

Main results

- Let $y = Ax^* + \eta$
- Solve $\hat{z} = \min_z ||y - AG(z)||$
- **Theorem 1:** If A is iid N(0, 1/m) with $m = O(kd \log n)$
- Then the reconstruction is close to optimal:

$$||G(\hat{z}) - x^*||_2 \leq C \min_z ||G(z) - x^*||$$

Main results

- Let $y = Ax^* + \eta$
- Solve $\hat{z} = \min_z ||y - AG(z)||$
- **Theorem 1:** If A is iid N(0, 1/m) with $m = O(kd \log n)$
- Then the reconstruction is close to optimal:

$$||G(\hat{z}) - x^*||_2 \leq C \min_z ||G(z) - x^*||$$

- (Reconstruction accuracy proportional to model accuracy)
- **Thm2:** More general result: m = O(k log L) measurements for any L-Lipschitz function G(z)

Main results

Theorem 1.1. Let $G : \mathbb{R}^k \to \mathbb{R}^n$ be a generative model from a d-layer neural network using ReLU activations. Let $A \in \mathbb{R}^{m\times n}$ be a random Gaussian matrix for $m = O(kd \log n)$, scaled so $A_{i,j} \sim N(0, 1/m)$. For any $x^* \in \mathbb{R}^n$ and any observation $y = Ax^* + \eta$, let $\widehat{z}$ minimize $\|y - AG(z)\|_2$ to within additive ϵ of the optimum. Then with $1 - e^{-\Omega(m)}$ probability,

$$\|G(\widehat{z}) - x^*\|_2 \leq 6 \min_{z^* \in \mathbb{R}^k} \|G(z^*) - x^*\|_2 + 3\|\eta\|_2 + 2\epsilon.$$

Representation error — noise — optimization error

- The first and second term are essentially necessary.
- The third term is the extra penalty ε for gradient descent sub-optimality.

Part 3

Proof ideas

Proof technology

Usual architecture of compressed sensing proofs for Lasso:

Lemma 1: A random Gaussian measurement matrix has **RIP/REC** whp for $m = k \log(n/k)$ measurements.

Lemma 2: Lasso works for matrices that have **RIP/REC**.

Lasso recovers a x_{hat} close to x^*

Proof technology

For a generative model defining a subset of images S:

Lemma 1: A random Gaussian measurement matrix has **S-REC** whp for sufficient measurements.

Lemma 2: The optimum of the squared loss minimization recovers a z_{hat} close to z^* **if A has S-REC.**

Proof technology

Why is the Restricted Eigenvalue Condition (REC) needed?

Lasso solves:

$$\min_{s.t.: ||Ax-y||_2 < \epsilon} ||x||_1$$

If there is a sparse vector x in the nullspace of A then this fails.

Proof technology

Why is the Restricted Eigenvalue Condition (REC) needed?

Lasso solves:

$$\min_{s.t.:||Ax-y||_2<\epsilon} ||x||_1$$

If there is a sparse vector x in the nullspace of A then this fails.

REC: All approximately k-sparse vectors x are far from the nullspace:

$$\gamma||x||_2 \leq ||Ax|||_2$$

A vector x is approximately k-sparse if there exists a set of k coordinates S such that

$$||x_S||_1 \geq ||x_{S^c}||_1$$

Proof technology

Unfortunate coincidence: The difference of two k-sparse vectors is 2k sparse.

But the difference of two natural images is not natural.

The correct way to state REC (That generalizes to our S-REC) is

For **any two k-sparse** vectors x1,x2 , their difference is far from the nullspace:

$$\gamma ||x_1 - x_2||_2 \leq ||A(x_1 - x_2)||_2$$

Proof technology

Our Set-Restricted Eigenvalue Condition (**S-REC**). For any set

$S \subset \mathbb{R}^n$

A matrix A satisfies **S-REC** if for all x_1, x_2 in S

For **any two natural images,** their difference is far from the nullspace of A:

$$\gamma||x_1 - x_2||_2 \leq ||A(x_1 - x_2)||_2$$

Proof technology

Our Set-Restricted Eigenvalue Condition (**S-REC**). For any set

$S \subset \mathbb{R}^n$

A matrix A satisfies S-REC if for all x_1, x_2 in S

The difference of two natural images is far from the nullspace of A:

$$\gamma||x_1 - x_2||_2 \leq ||A(x_1 - x_2)||_2$$

- **Lemma 1**: If the set S is the range of a generative model of d-relu layers then
- m= O (k d logn) measurements suffice to make a Gaussian iid matrix S-REC whp.

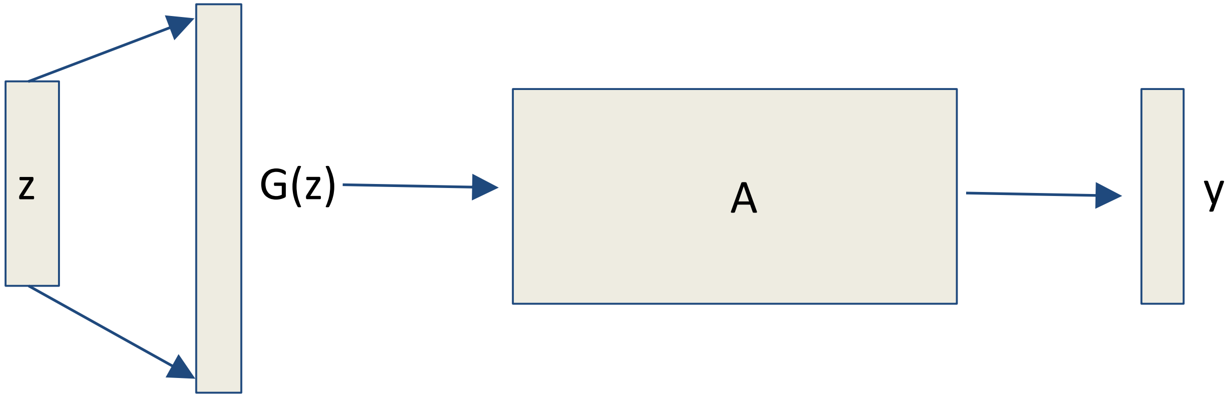
- **Lemma 2:** If the matrix has S-REC then squared loss optimizer z_{hat} must be close to z^*

Outline

- Generative Models
- Using generative models for compressed sensing
- Main theorem and proof technology
- Using an untrained GAN (Deep Image Prior)
- Conclusions

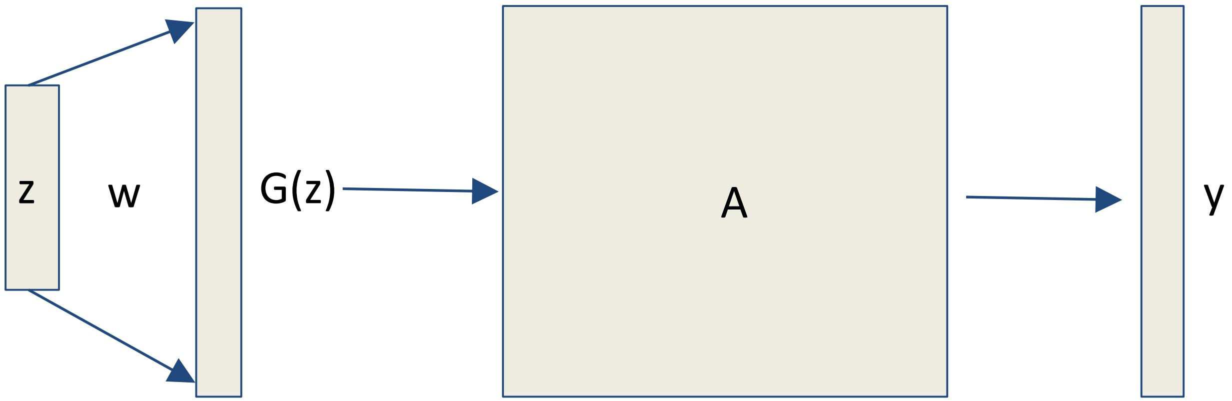
- Other extensions:
- Using non-linear measurements
- Using GANs to defend from Adversarial examples.
- AmbientGAN
- CausalGAN

Recovery from linear measurements



$$\min_{z \in \mathbb{R}^k} ||y - AG(z)||_2$$

Lets focus on A =I (Denoising)

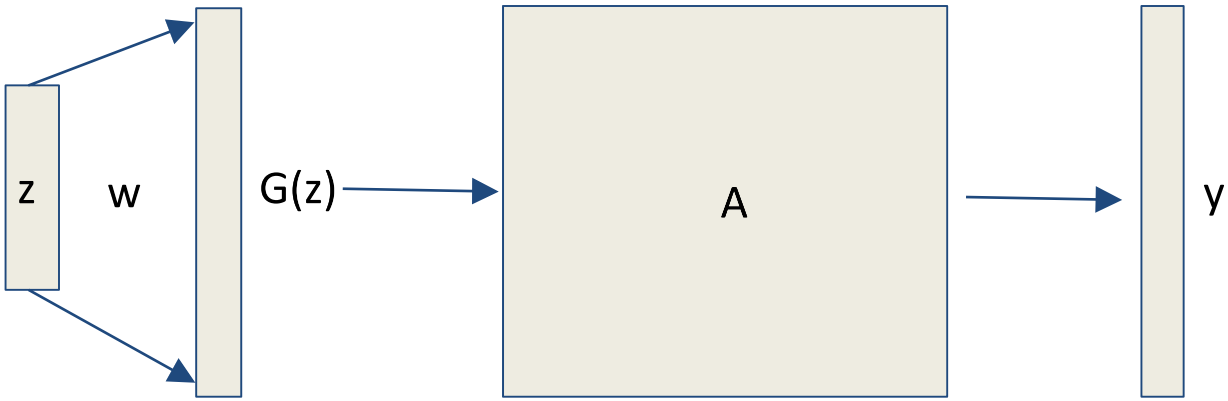


$$y = x + z$$

$$\min_{z \in \mathbb{R}^k} ||y - G(z)||_2$$

But I do not have the right weights w of the generator!

Denoising with Deep Image Prior



$$y = x + z$$

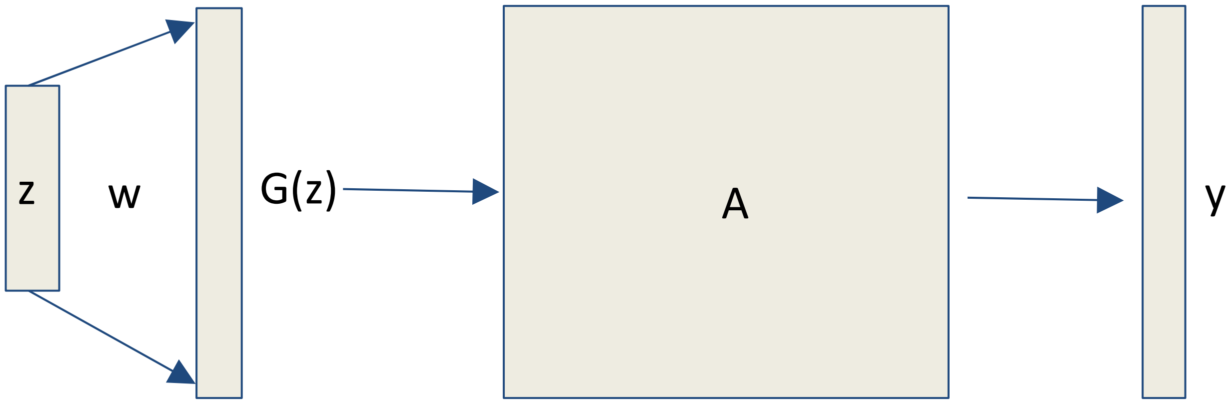
$$\min_{z \in \mathbb{R}^k} ||y - G(z)||_2$$

$$\min_{w \in \mathbb{R}^k} ||y - G(z_0, w)||_2$$

But I do not have the right weights w of the generator!

Train over weights w. Keep random z_0

Denoising with Deep Image Prior



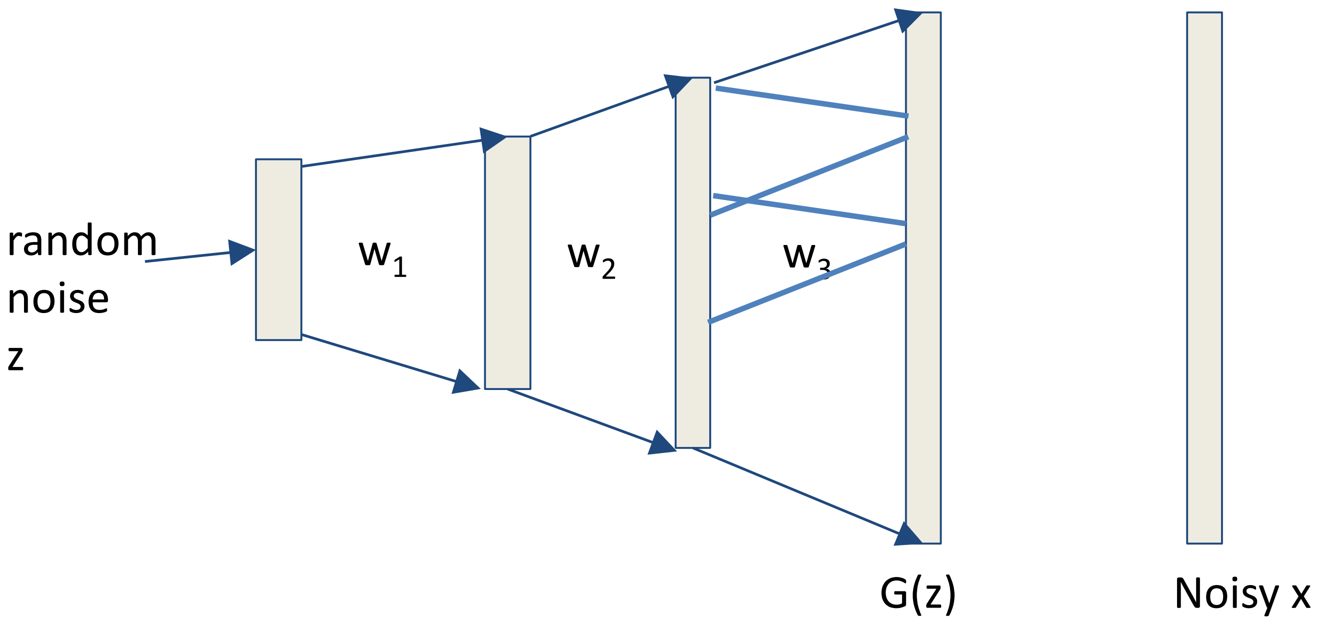
mi
$z\in\mathbb{R}$

$$\min_{w\in\mathbb{R}^k} ||y - G(z_0, w)||_2$$

E … ts w of the generator!

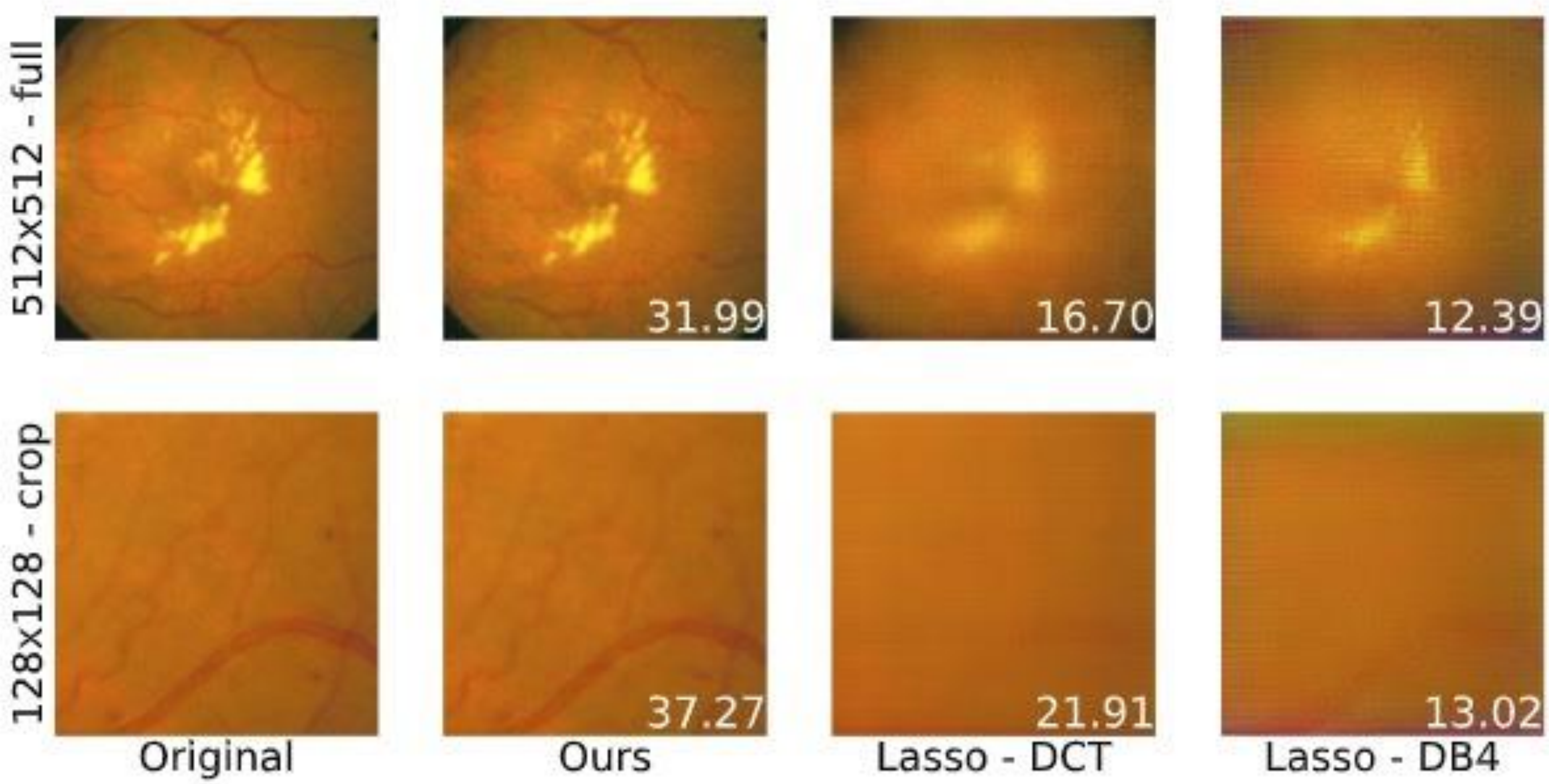
T … m z_0

Deep image prior



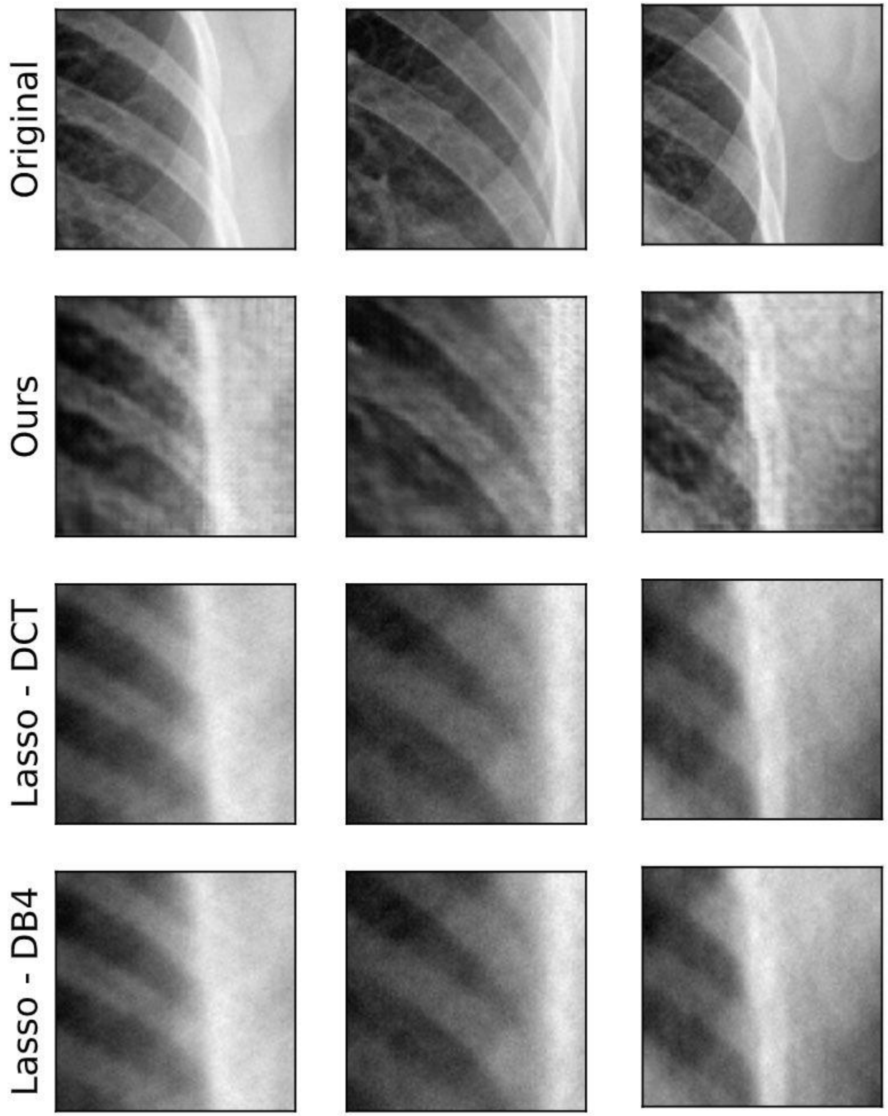
The fact that an image can be generated by convolutional weights applied to some random noise, makes it natural

Can be applied to any dataset



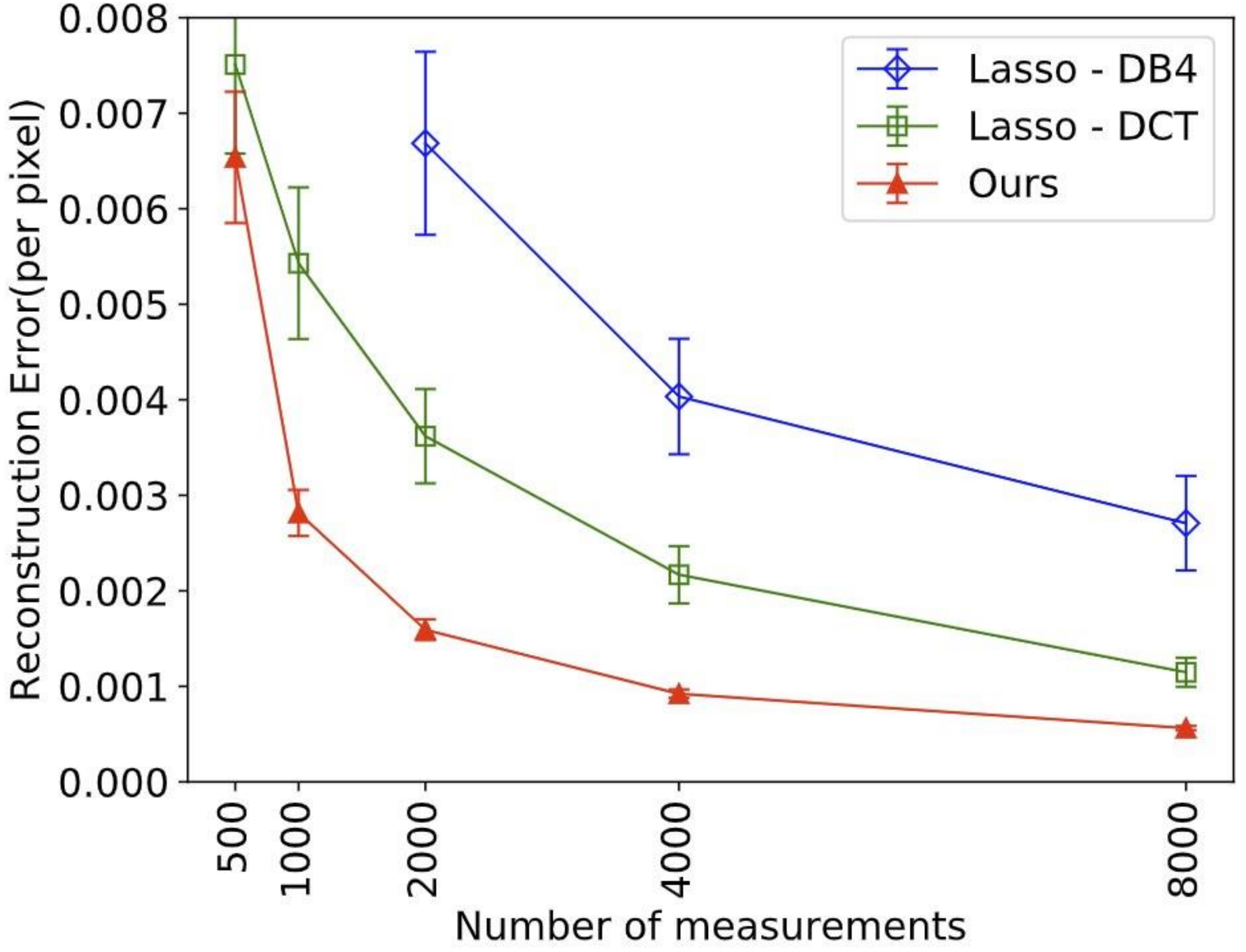
From our recent preprint:
Compressed Sensing with Deep Image Prior and Learned Regularization

Can be applied to any dataset



From our recent preprint:
Compressed Sensing with Deep Image Prior and Learned Regularization

DIP-CS vs Lasso



From our recent preprint:
Compressed Sensing with Deep Image Prior and Learned Regularization

Conclusions and outlook

- Defined compressed sensing for images coming from generative models
- Performs very well for few measurements. Lasso is more accurate for many measurements.
- Ideas: Better loss functions, combination with lasso, using discriminator in reconstruction.
- Theory of compressed sensing nicely extends to S-REC and recovery approximation bounds.
- Algorithm can be applied to non-linear measurements. Can solve general inverse problems for differentiable measurements.
- Plug and play different differentiable boxes !
- Better generative models (eg for MRI datasets) can be useful.
- Deep Image prior can be applied even without a pre-trained GAN
- Idea of differentiable compression seems quite general.

- Code and pre-trained models:
- https://github.com/AshishBora/csgm
- https://github.com/davevanveen/compsensing_dip

fin

Main results

Theorem 1.2. Let $G : \mathbb{R}^k \to \mathbb{R}^n$ be an L-Lipschitz function. Let $A \in \mathbb{R}^{m \times n}$ be a random Gaussian matrix for $m = O(k \log \frac{Lr}{\delta})$, scaled so $A_{i,j} \sim N(0, 1/m)$. For any $x^* \in \mathbb{R}^n$ and any observation $y = Ax^* + \eta$, let $\widehat{z}$ minimize $\|y - AG(z)\|_2$ to within additive ϵ of the optimum over vectors with $\|\widehat{z}\|_2 \le r$. Then with $1 - e^{-\Omega(m)}$ probability,

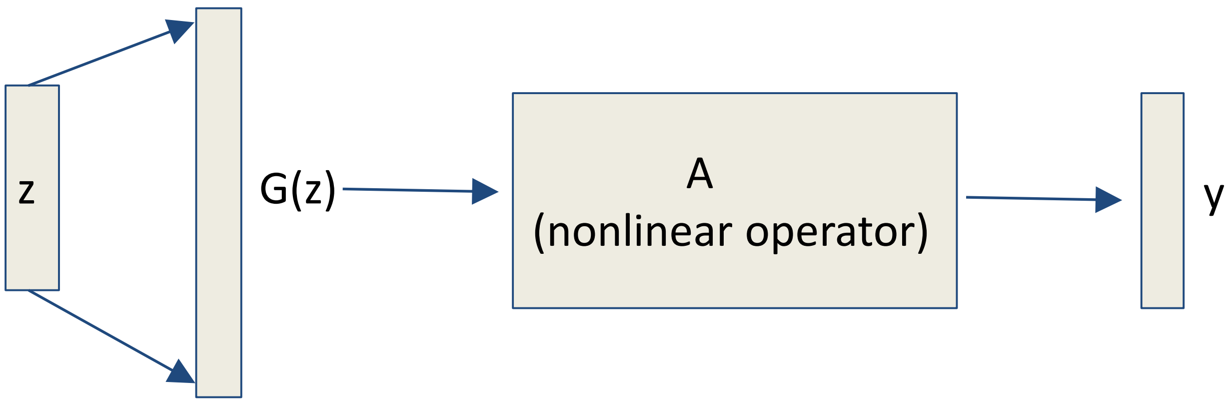
$$\|G(\widehat{z}) - x^*\|_2 \le 6 \min_{\substack{z^* \in \mathbb{R}^k \\ \|z^*\|_2 \le r}} \|G(z^*) - x^*\|_2 + 3\|\eta\|_2 + 2\epsilon + 2\delta.$$

- For general L-Lipschitz functions.
- Minimize only over z vectors within a ball.
- Assuming poly(n) bounded weights: $L= n^{O(d)}$,$\delta= 1/n^{O(d)}$

Intermezzo

Our algorithm works
even for non-linear measurements.

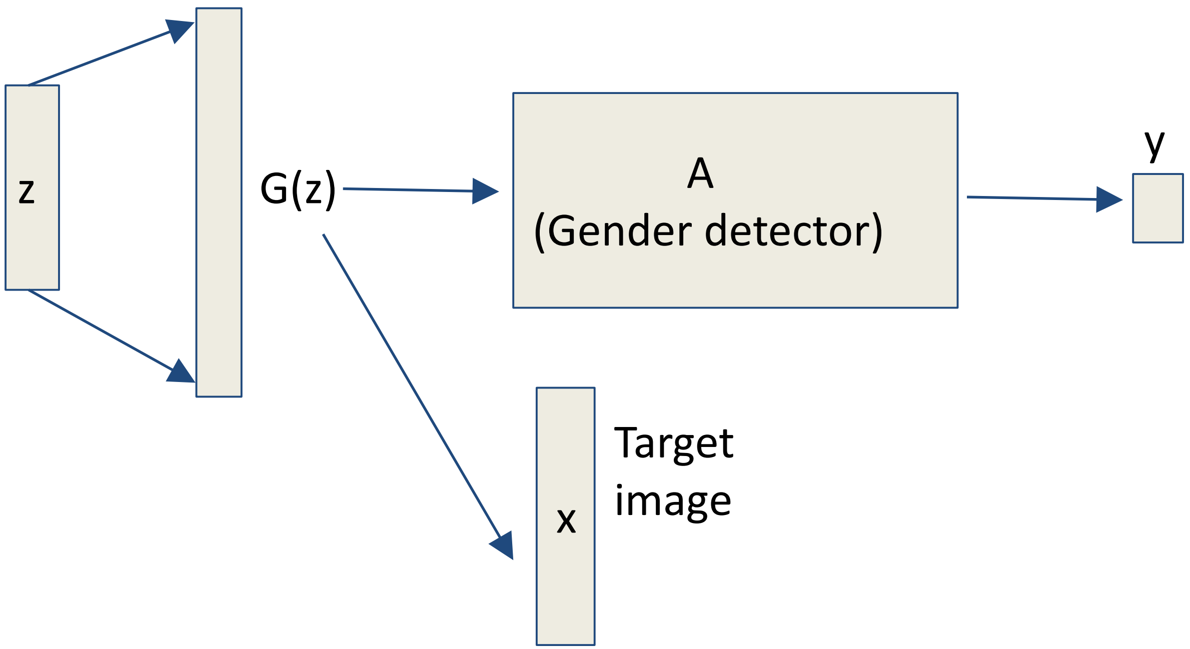
Recovery from nonlinear measurements



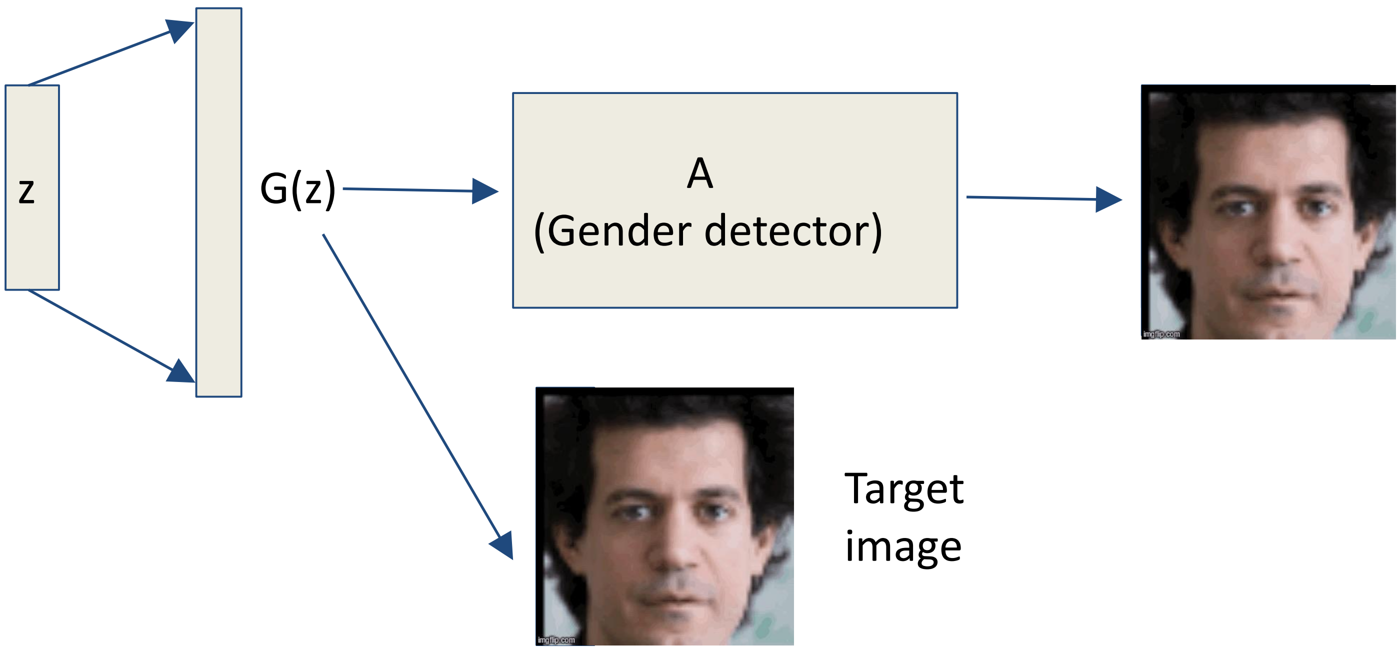
$$\min_{z \in \mathbb{R}^k} ||y - AG(z)||_2$$

- This recovery method can be applied even for any non-linear measurement **differentiable** box A.
- Even a mixture of losses: approximate my face but also amplify a mustache detector loss.

Using nonlinear measurements

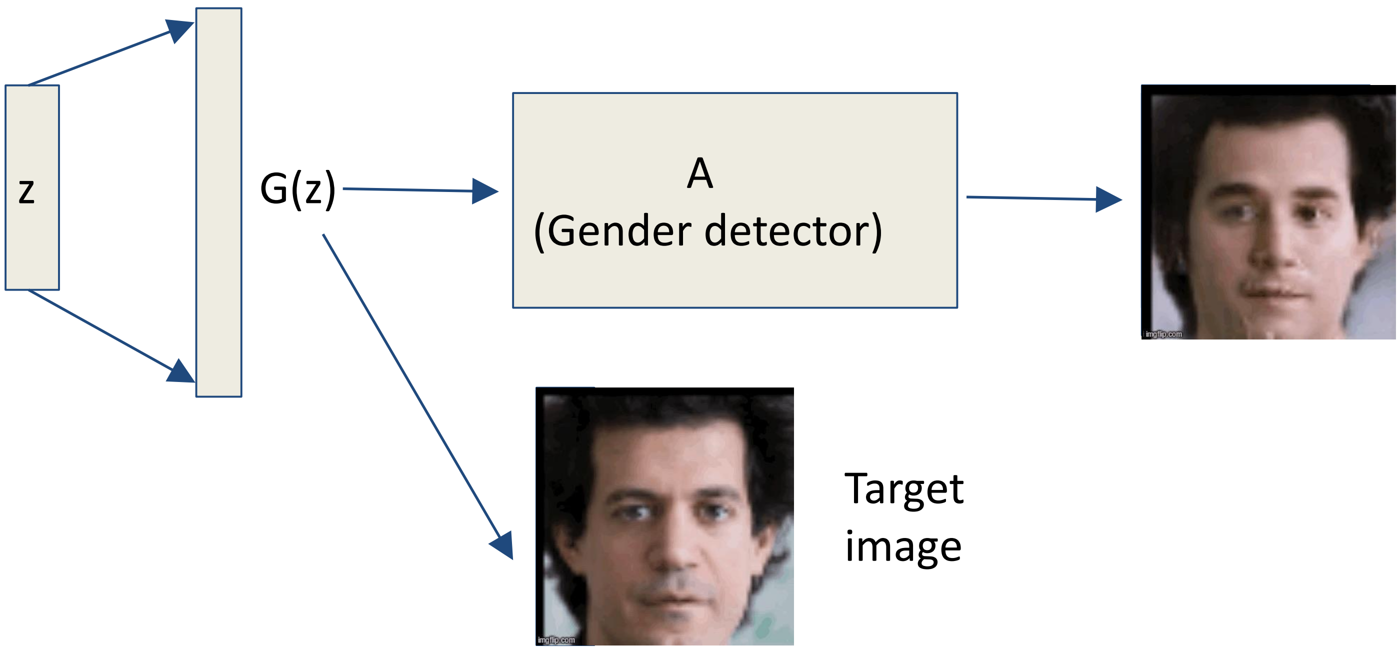


Using nonlinear measurements



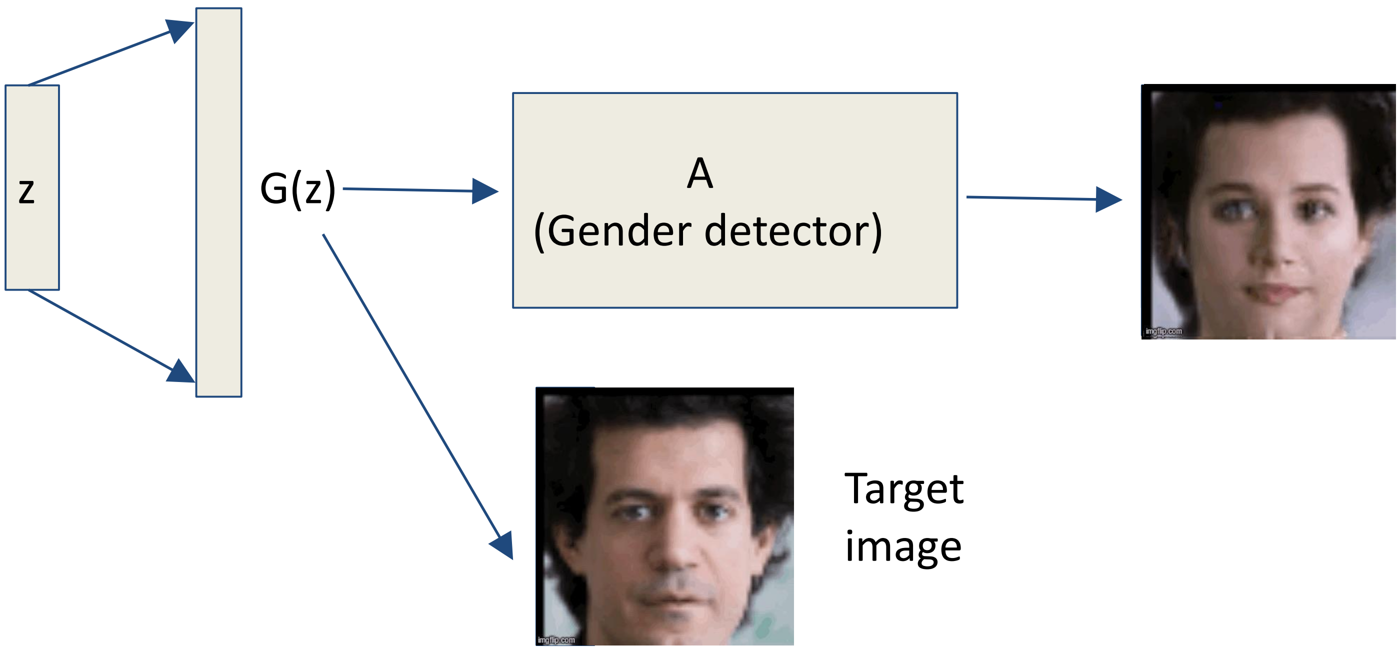
$$\min_{z \in \mathbb{R}^k} ||G(z) - x_{\text{Costis}}|| + \lambda \texttt{MaleProb}(G(z))$$

Using nonlinear measurements



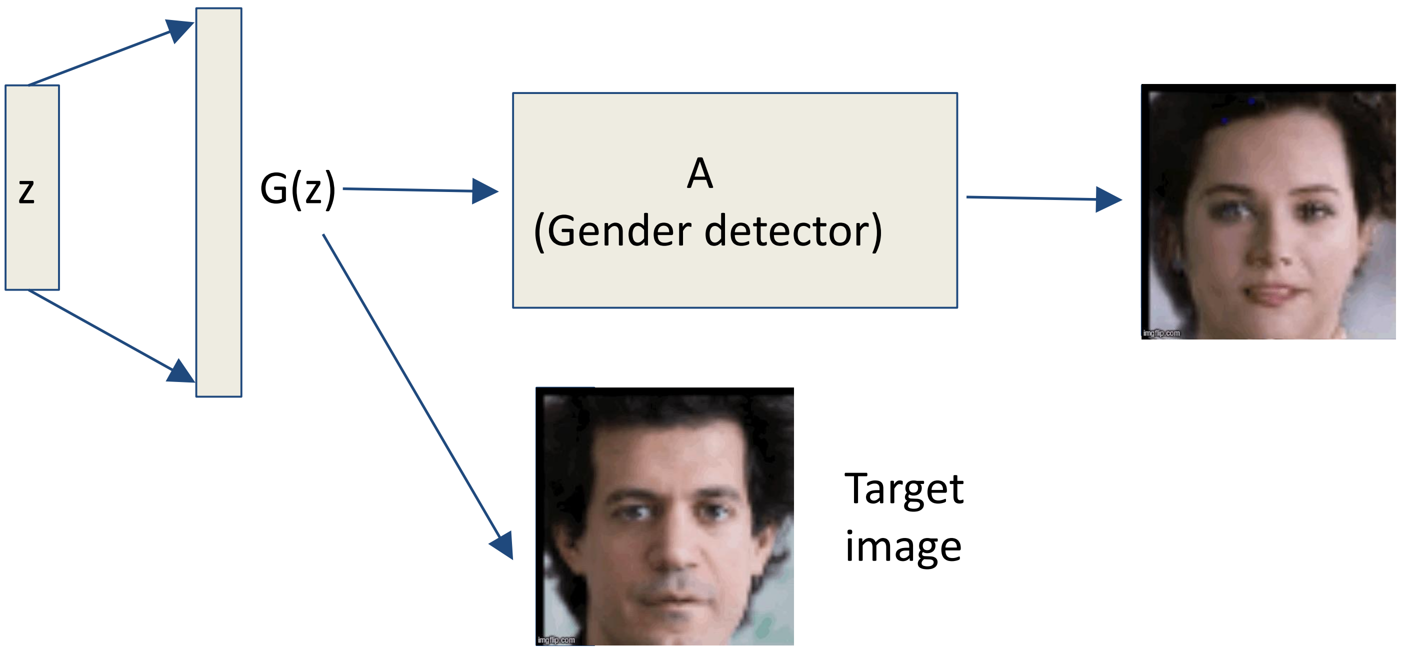
$$\min_{z \in \mathbb{R}^k} ||G(z) - x_{\text{Costis}}|| + \lambda \texttt{MaleProb}(G(z))$$

Using nonlinear measurements



$$\min_{z \in \mathbb{R}^k} ||G(z) - x_{\text{Costis}}|| + \lambda \texttt{MaleProb}(G(z))$$

Using nonlinear measurements



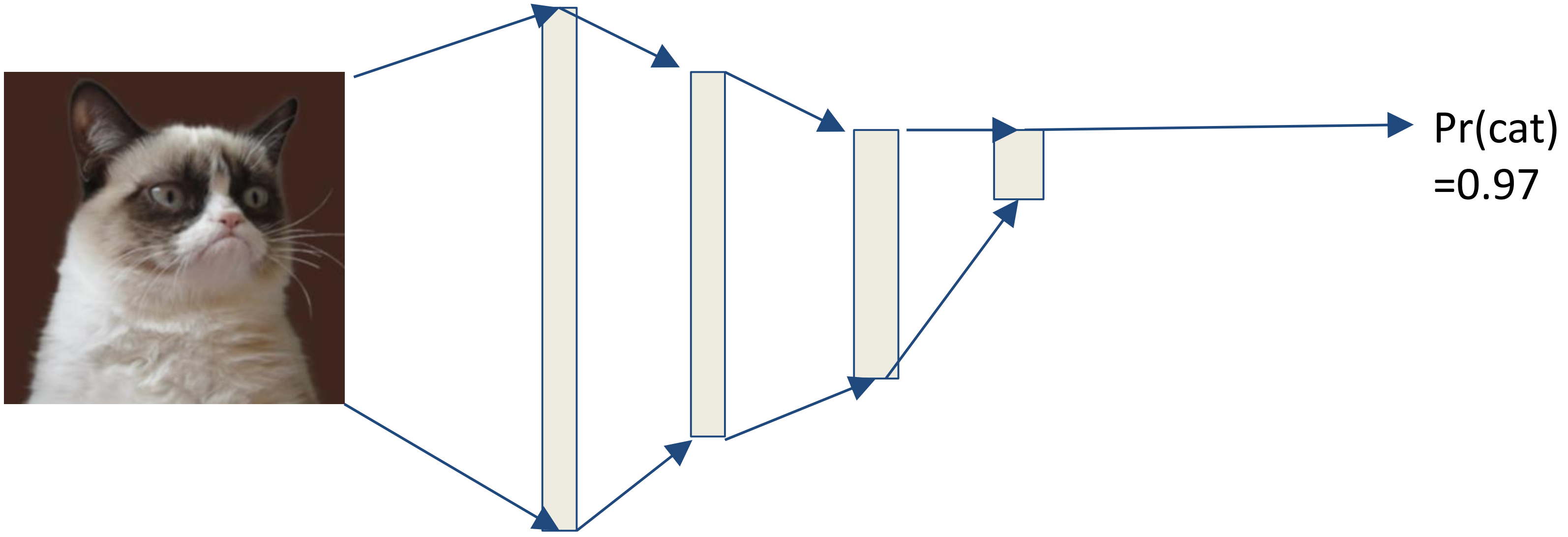
$$\min_{z \in \mathbb{R}^k} ||G(z) - x_{\text{Costis}}|| + \lambda \texttt{MaleProb}(G(z))$$

Part 4: Dessert

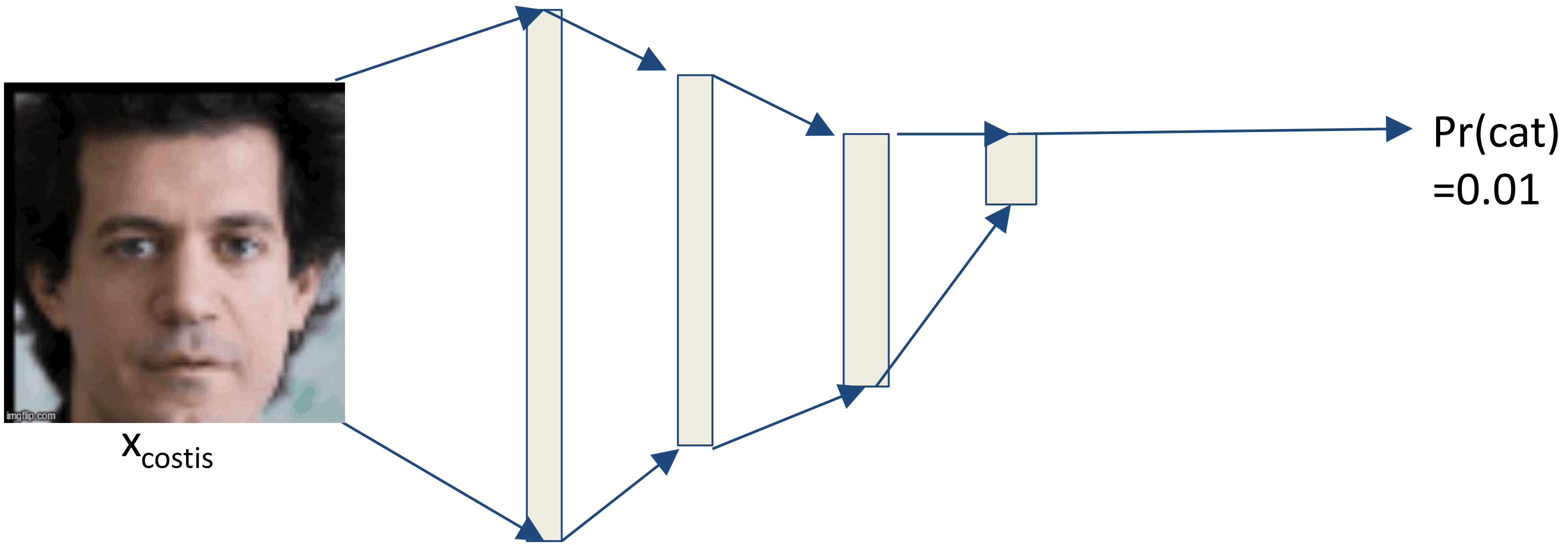
Adversarial examples in ML

Using the idea of compressed sensing to defend from adversarial attacks.

Lets start with a good cat classifier

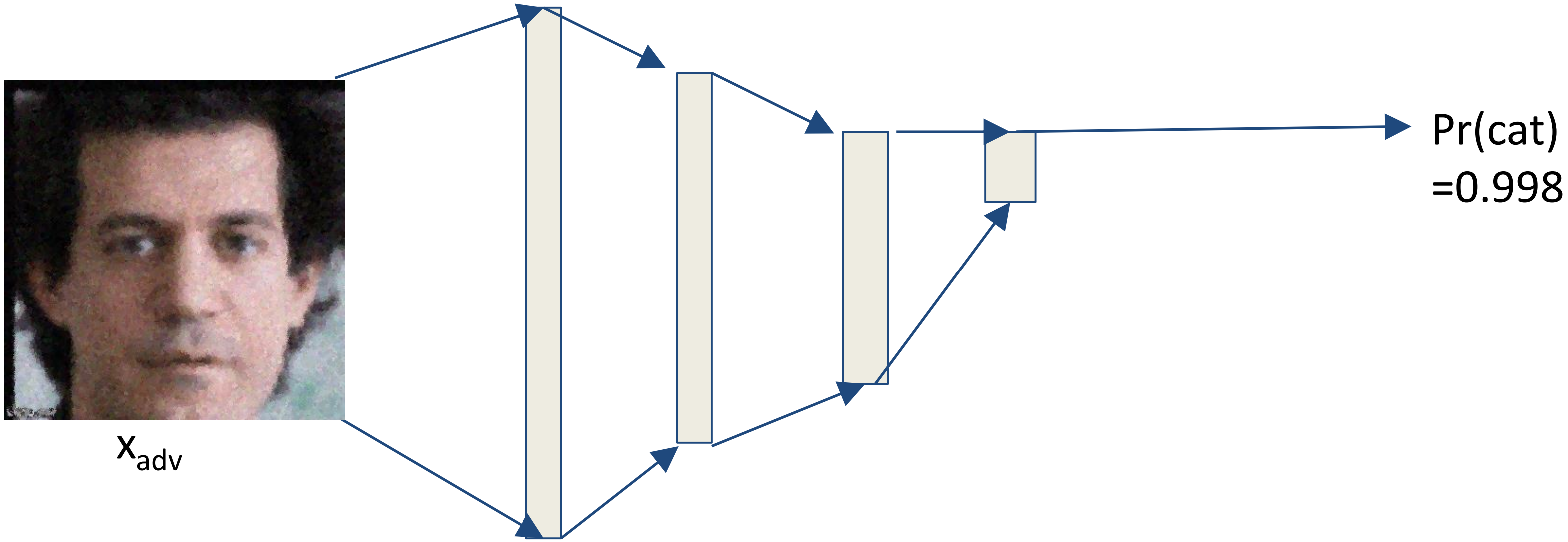


Modify image slightly to maximize $P_{cat}(x)$



Move x input to maximize 'catness' of x while keeping it close to x_{costis}

Adversarial examples

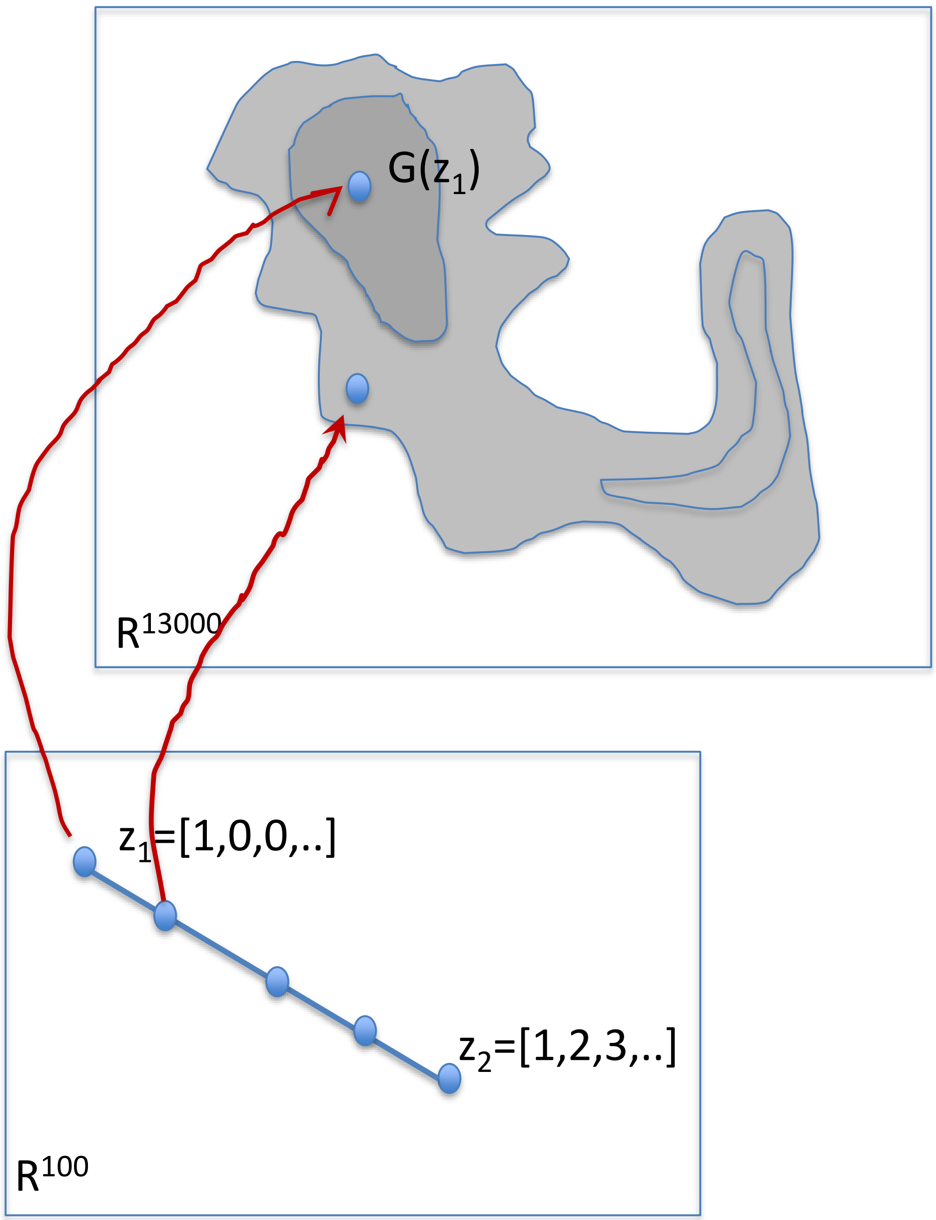


x_{adv}

Move x input to maximize 'catness' of x while keeping it close to x_{costis}

1. Moved in the direction pointed by cat classifier
2. Left the manifold of natural images

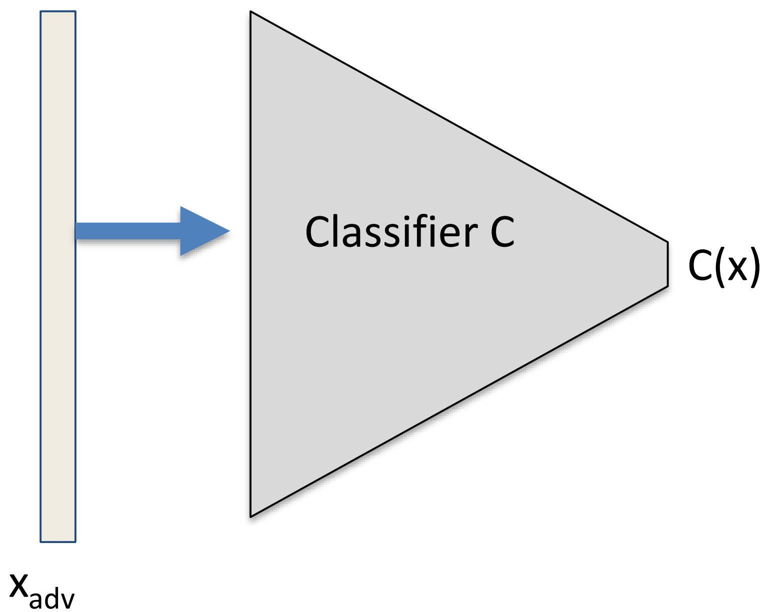
Difference from before?



In our previous work we were doing gradient descent in z-space so staying in the **range** of the Generator.

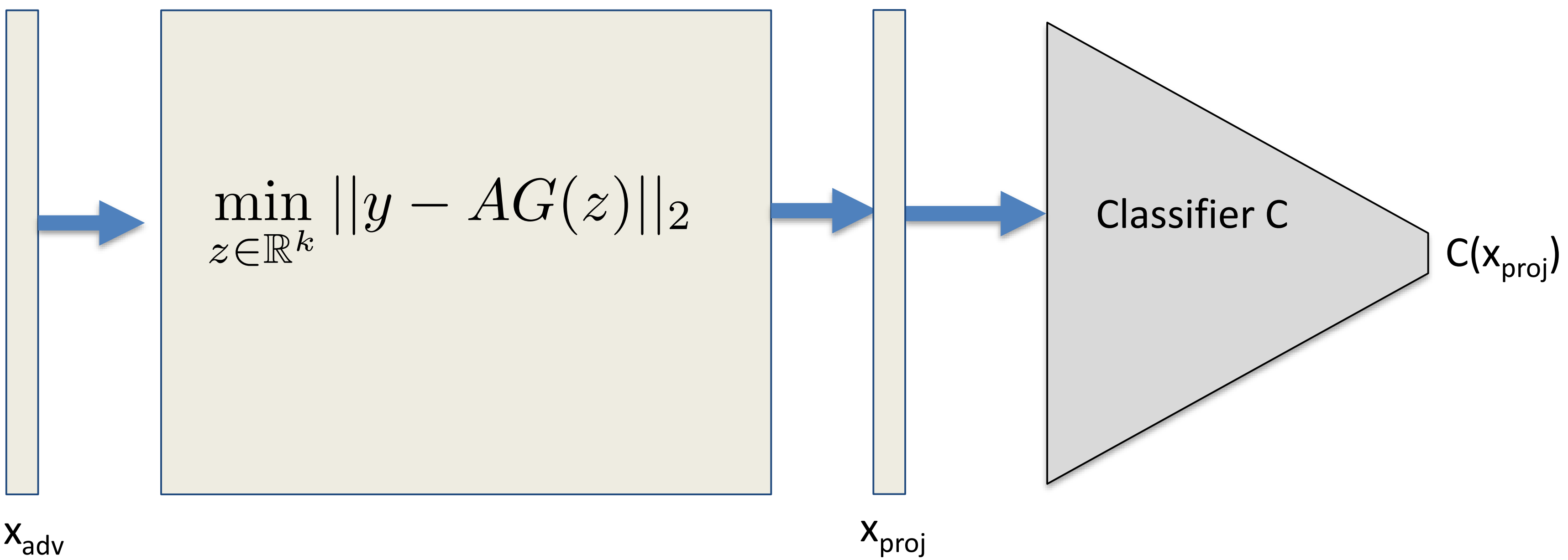
- **Suggests** that there are no adversarial examples in the range of the generator
- Shows a way to defend classifiers if we have a GAN for the domain: simply project on the range before classifying.
- (we have a preprint on that).

Defending using a classifier using a GAN



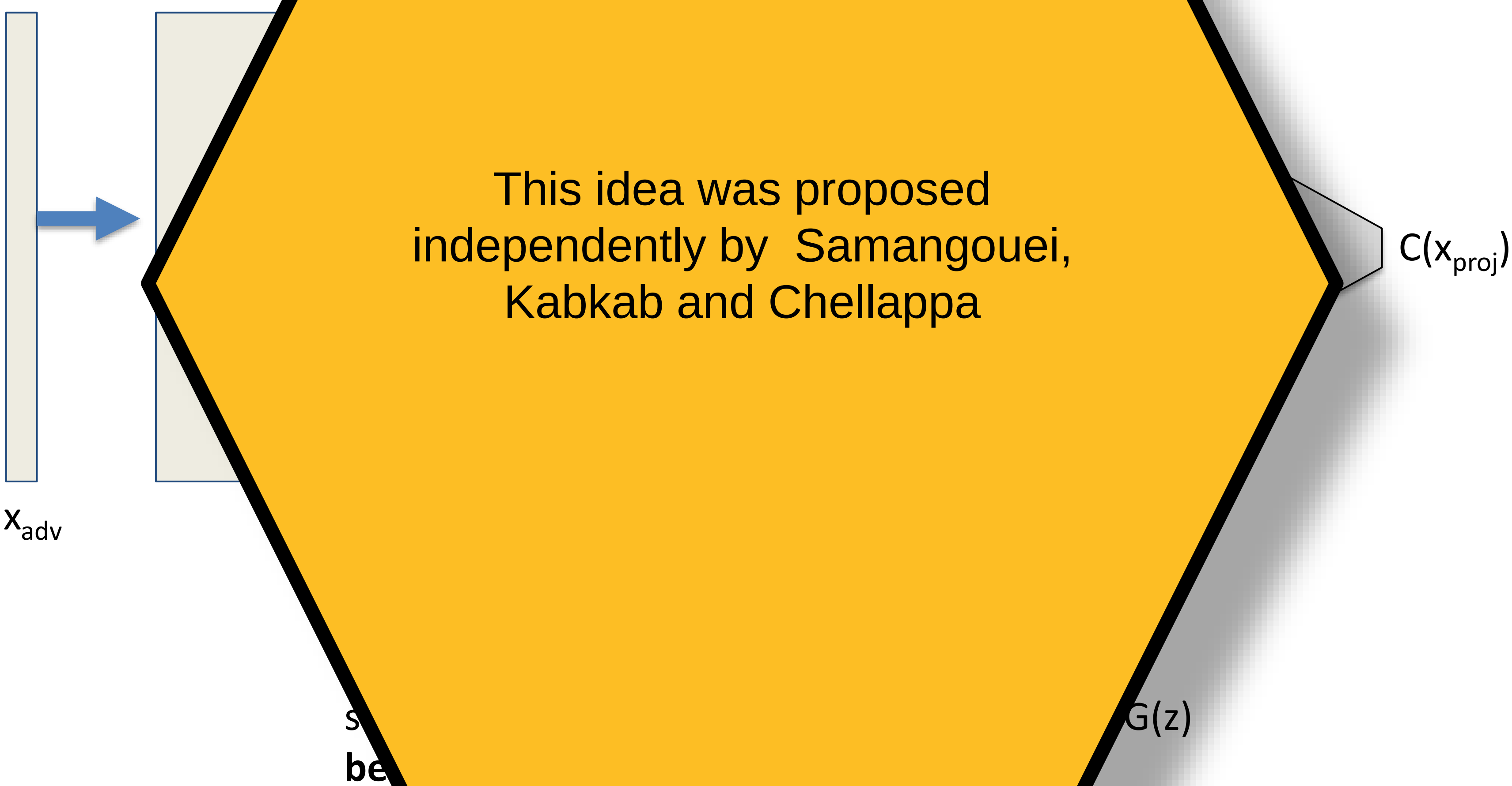
Unprotected classifier with input x_{adv}

Defending using a classifier using a GAN



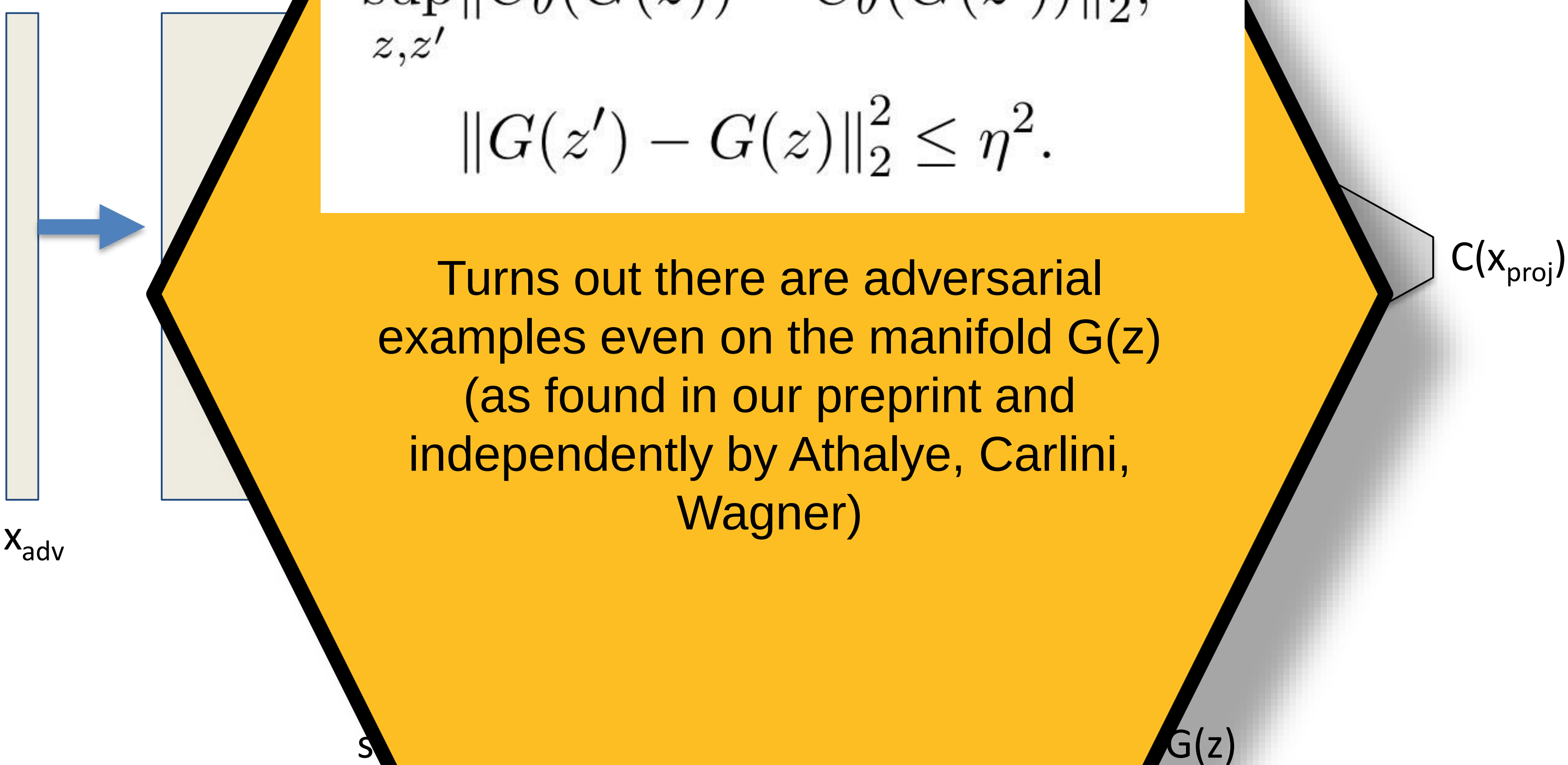
Treating x_{adv} as noisy nonlinear compressed sensing observations. Projecting on manifold G(z) **before feeding in classifier.**

Defending using a classifier



This idea was proposed independently by Samangouei, Kabkab and Chellappa

Defending using a classifier

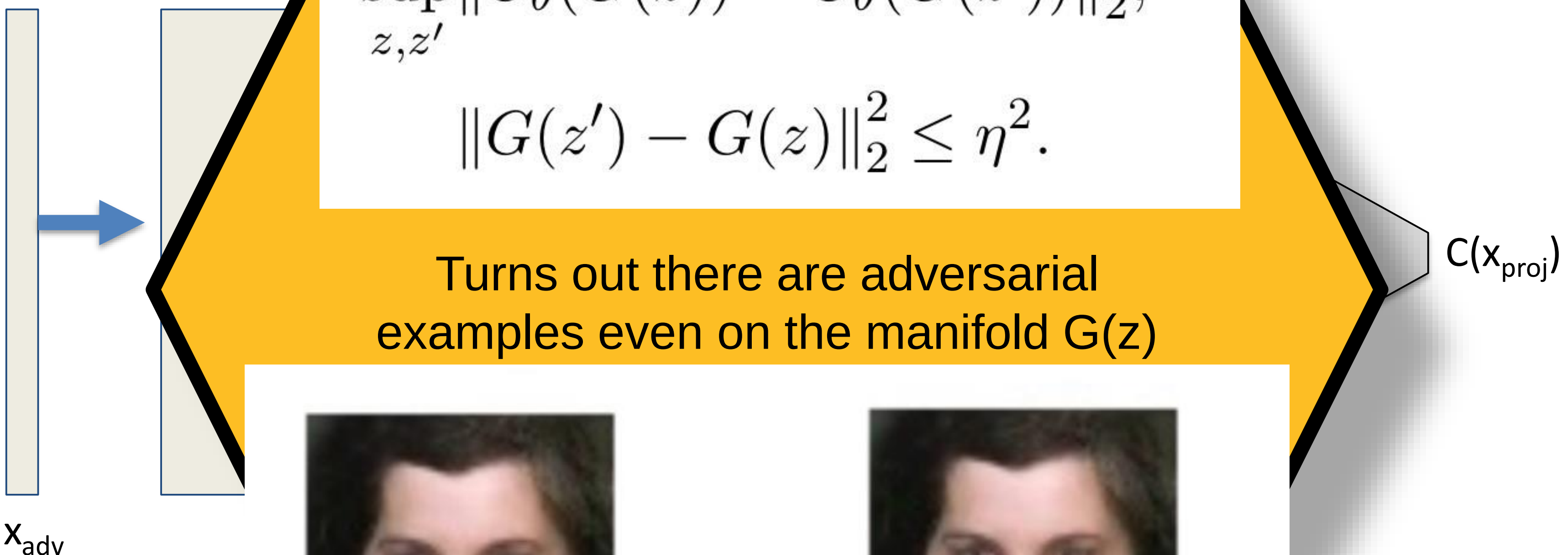


$$\sup_{z,z'} \|C_\theta(G(z)) - C_\theta(G(z'))\|_2^2,$$

$$\|G(z') - G(z)\|_2^2 \leq \eta^2.$$

Turns out there are adversarial examples even on the manifold G(z) (as found in our preprint and independently by Athalye, Carlini, Wagner)

Defending using a classifier



$$\sup_{z,z'} \|C_\theta(G(z)) - C_\theta(G(z'))\|_2^2,$$

$$\|G(z') - G(z)\|_2^2 \leq \eta^2.$$

Turns out there are adversarial examples even on the manifold G(z)

P(man)=0.99 P(woman)=0.99

Defending using a classifier

This idea was proposed independently by Samangouei, Kabkab and Chellappa

Turns out there are adversarial examples even on the manifold G(z) (as found in our preprint and independently by Athalye, Carlini, Wagner)

Can be made robust using adversarial training on the manifold: Robust Manifold Defense.

The Robust Manifold Defense (Arxiv paper)
Blog post on *Approximately Correct* on using GANs for defense

CausalGAN

work with Murat Kocaoglu and Chris Snyder,

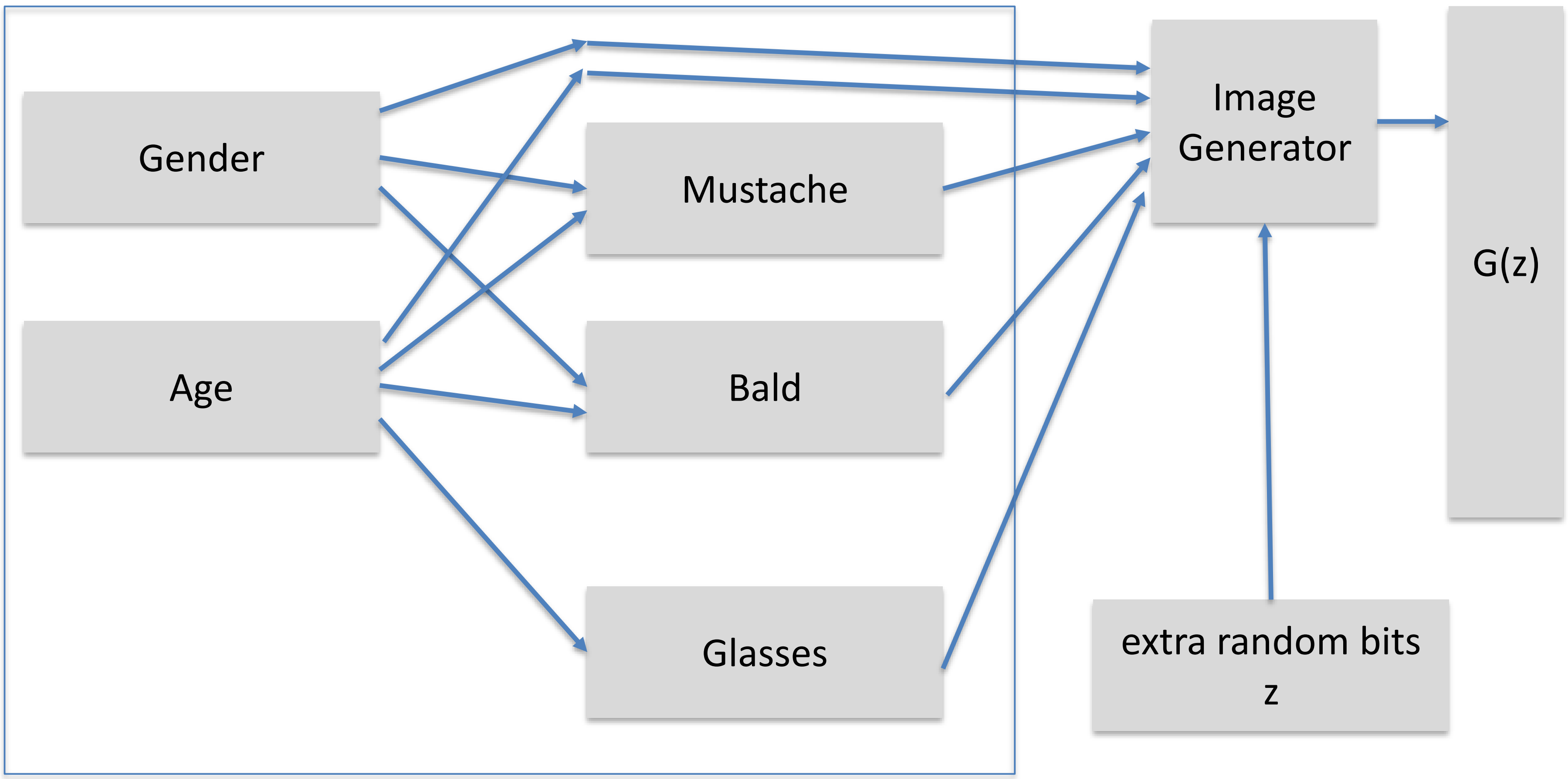
Postulate a causal structure on attributes (gender, mustache, long hair, etc)

Create a machine that can sample conditional and interventional samples: we call that an **implicit causal generative model.**

Adversarial training.

The causal generator seems to allow configurations never seen in the dataset (e.g. women with mustaches)

CausalGAN



Gender
Age
Mustache
Bald
Glasses
Image Generator
extra random bits z
G(z)

CausalGAN

Conditioning on Bald=1 vs Intervention (Bald=1)

CausalGAN

Conditioning on Bald=1 vs Intervention (Bald=1)

CausalGAN

Conditioning on Mustache=1 vs Intervention (Mustache=1)

CausalGAN

Conditioning on Mustache=1 vs Intervention (Mustache=1)