Generative Adversarial Networks

These kittens are not real!

Image taken from https://github.com/aleju/cat-generator

Generative models

Given: A dataset containing samples from distribution p_{data}.
Goal: Find an approximation p_{model} of p_{data}.

Different flavours:

- density estimation,
- sampling,
- or both.

Note: We do not want to sample from the data, we want more kittens!

But why?

- better understand the data (model structure, parameters, etc.),
- learn useful embeddings,
- impute missing data,
- facilitate supervised learning,
- do cool stuff.

Generative Adversarial Networks (GANs)

Generative modelling is not a new idea, several approaches exist, notably Variational Auto Encoders.

We focus on GANs:

- for many exciting tasks they deliver state-of-the-art,
- attractive conceptually,
- extremely active area of research,
- no obvious limitations, except,
- notoriously hard to train (no, it’s a good thing!).

GANs - motivation

Setting:
Currency with notes that do not change over time.
Obvious target for counterfeiters.
Need to establish anti-counterfeit agency.
What happens in the long run?

Counterfeiters and agents improve simultaneously.

Eventually, counterfeiters produce perfect fakes and agents are helpless.

GANs

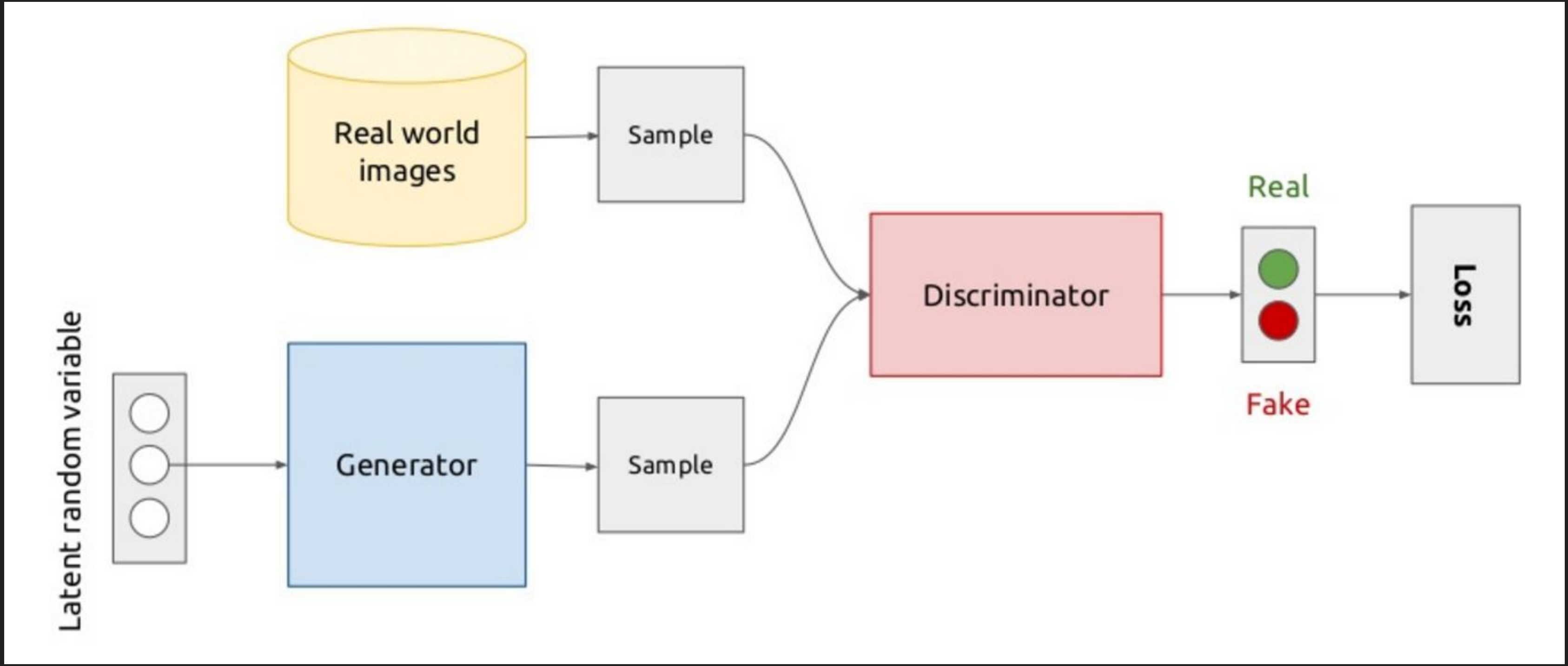


Image taken from http://imatge-upc.github.io/telecombcn-2016-dlcv/

GAN training

- Generate a batch of real images.
- Generate a batch of fake images from a batch of random vectors.
- Make a gradient step for discriminator on the two batches.
- Generate a batch of random vectors.
- Make a gradient step for generator to best fool the discriminator on these vectors.

Important consequence: the path from parameters to loss function needs to be differentiable -> no text generation.

Modern generator (DCGAN)

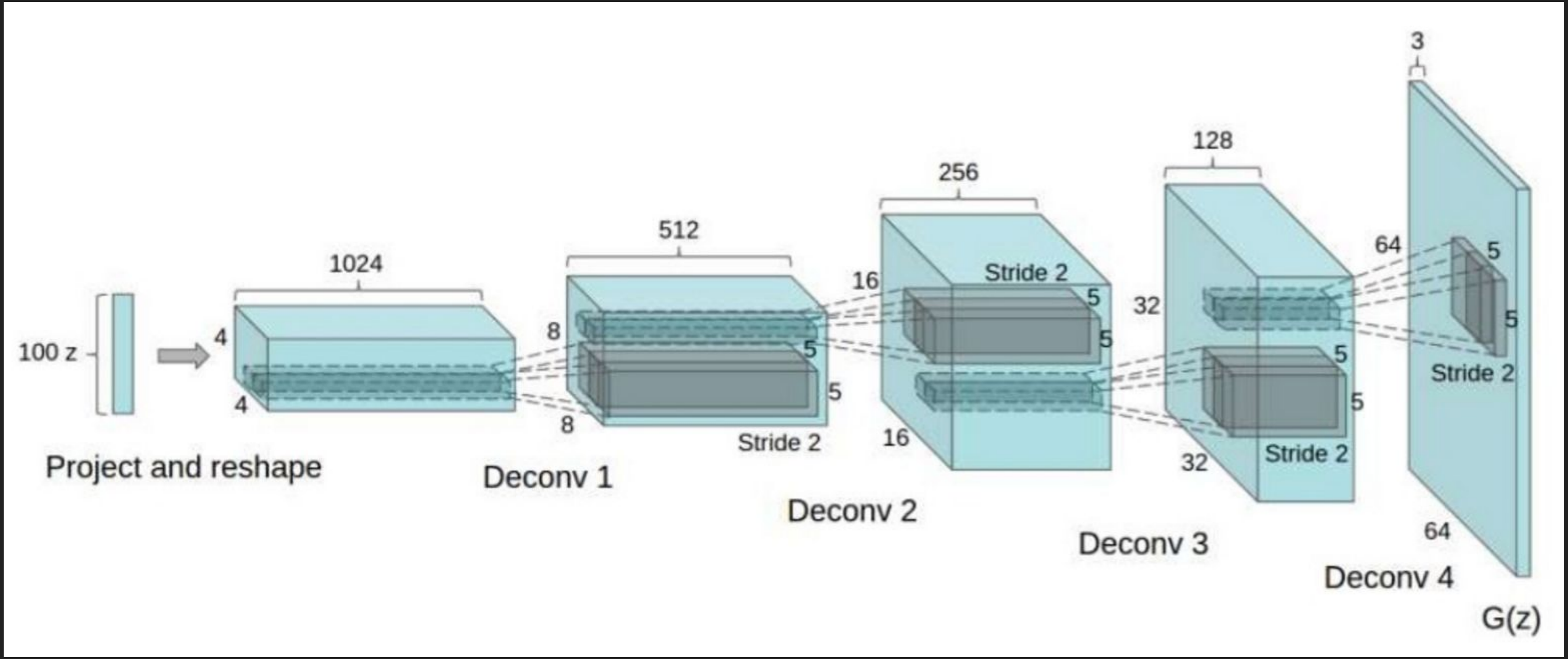


Image taken from https://arxiv.org/abs/1511.06434

Some applications

Generating bedrooms

Image taken from https://arxiv.org/abs/1511.06434

Interpolating bedrooms

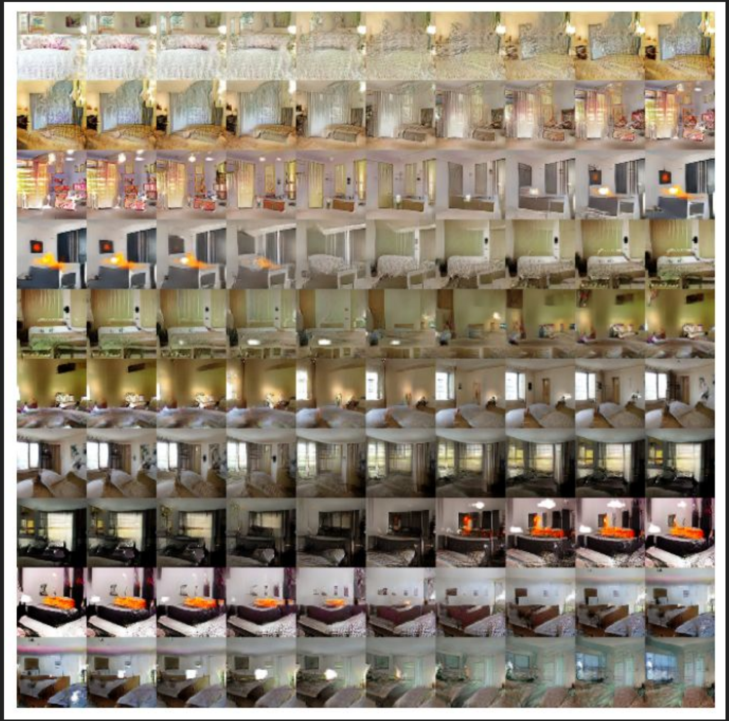


Image taken from https://arxiv.org/abs/1511.06434

Code arithmetic on faces

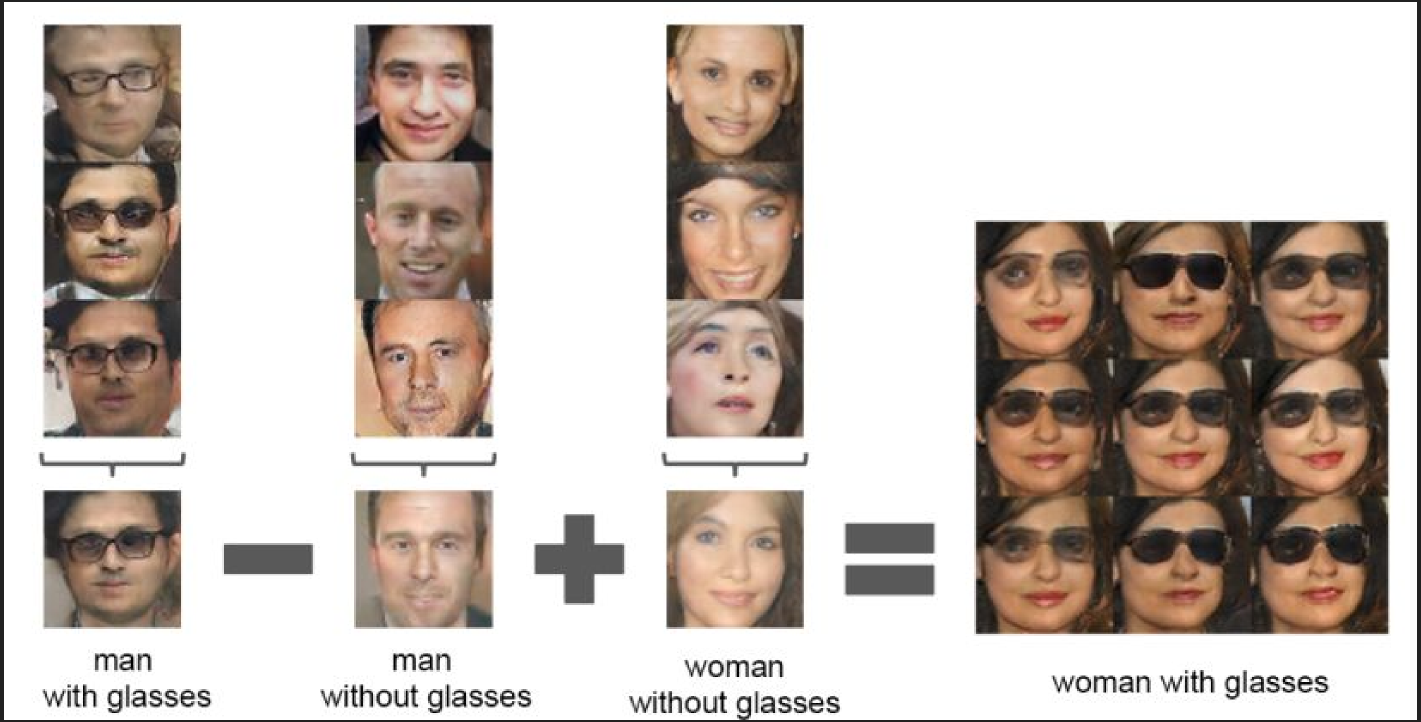


Image taken from https://arxiv.org/abs/1511.06434

Current state-of-the-art for ImageNet sampling

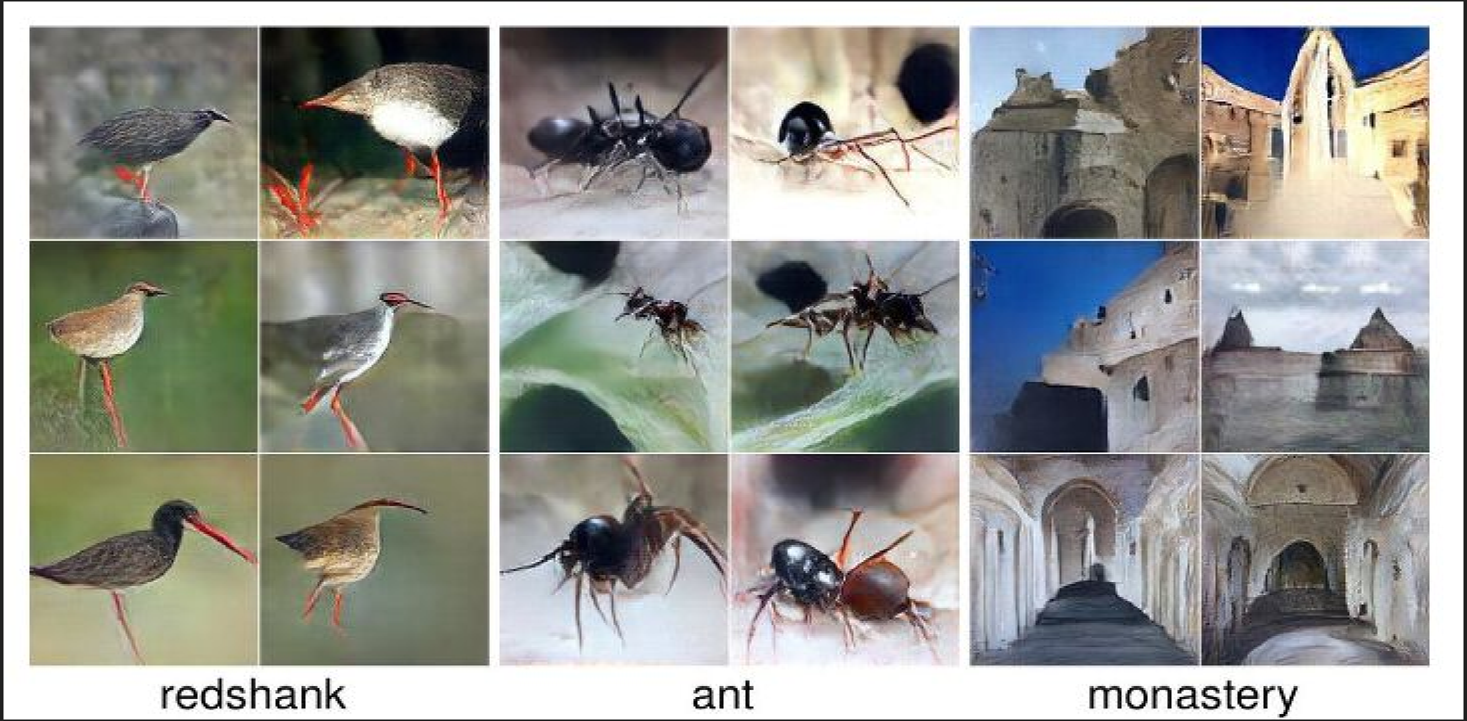


Image taken from http://www.evolvingai.org/ppgn

State-of-the-art for super-resolution

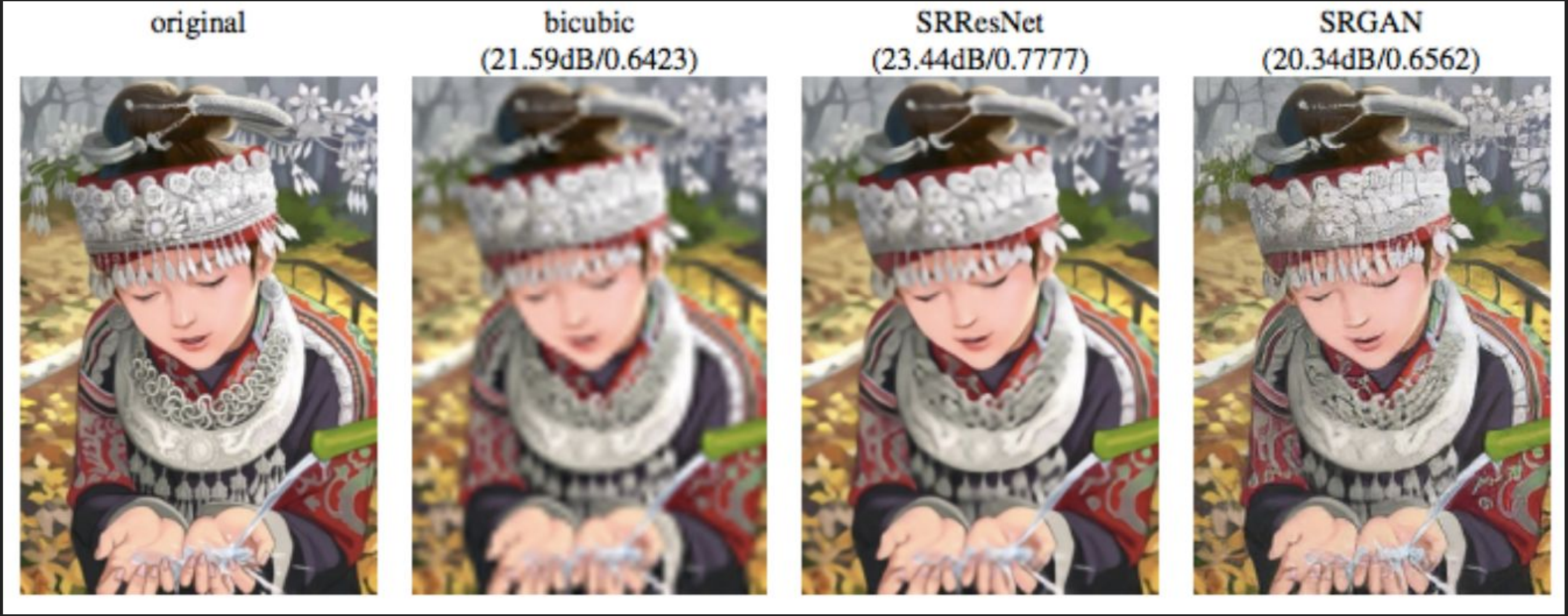
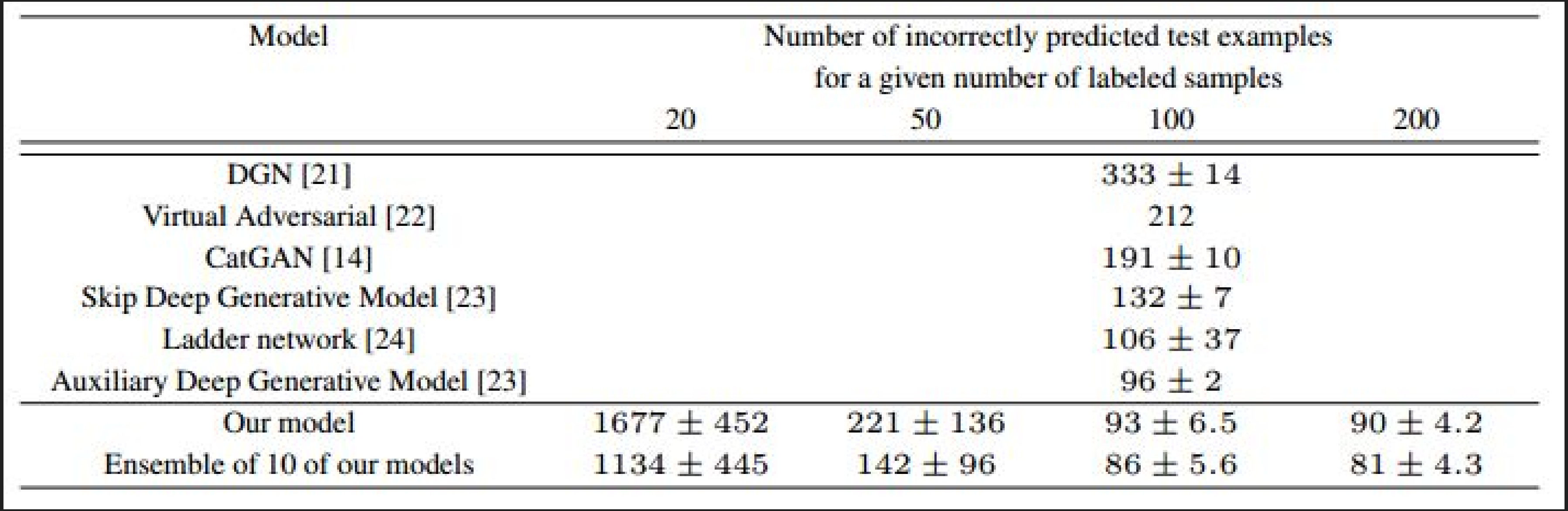


Image taken from https://arxiv.org/abs/1609.04802

Semi-supervised learning



Model	Number of incorrectly predicted test examples for a given number of labeled samples			
	20	50	100	200
DGN [21]			333 ± 14	
Virtual Adversarial [22]			212	
CatGAN [14]			191 ± 10	
Skip Deep Generative Model [23]			132 ± 7	
Ladder network [24]			106 ± 37	
Auxiliary Deep Generative Model [23]			96 ± 2	
Our model	1677 ± 452	221 ± 136	93 ± 6.5	90 ± 4.2
Ensemble of 10 of our models	1134 ± 445	142 ± 96	86 ± 5.6	81 ± 4.3

Discriminator predicts {fake,0,...,9}. We include class loss only if we know a label.

Image taken from https://arxiv.org/abs/1606.03498

Conditional GAN (CGAN)

One can condition the generator’s output on an arbitrary random variable y:

- append y to generator’s random input z.
- append y to discriminator’s input.
- the disciminator decides: does the image come from $p_{data}(\ |y)$.

Examples:

- sample a given digit from MNIST,
- sample a given class from ImageNet.

With conditional GANs you can do so much more!

Next frame prediction

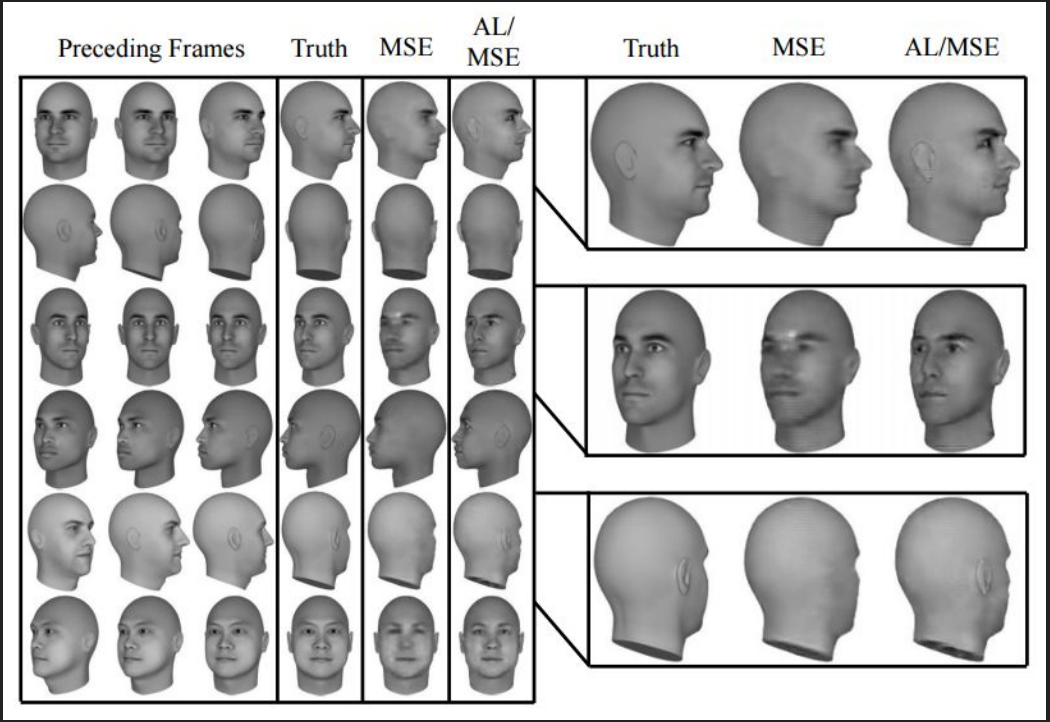


Image taken from https://arxiv.org/abs/1511.06380

Next frame prediction

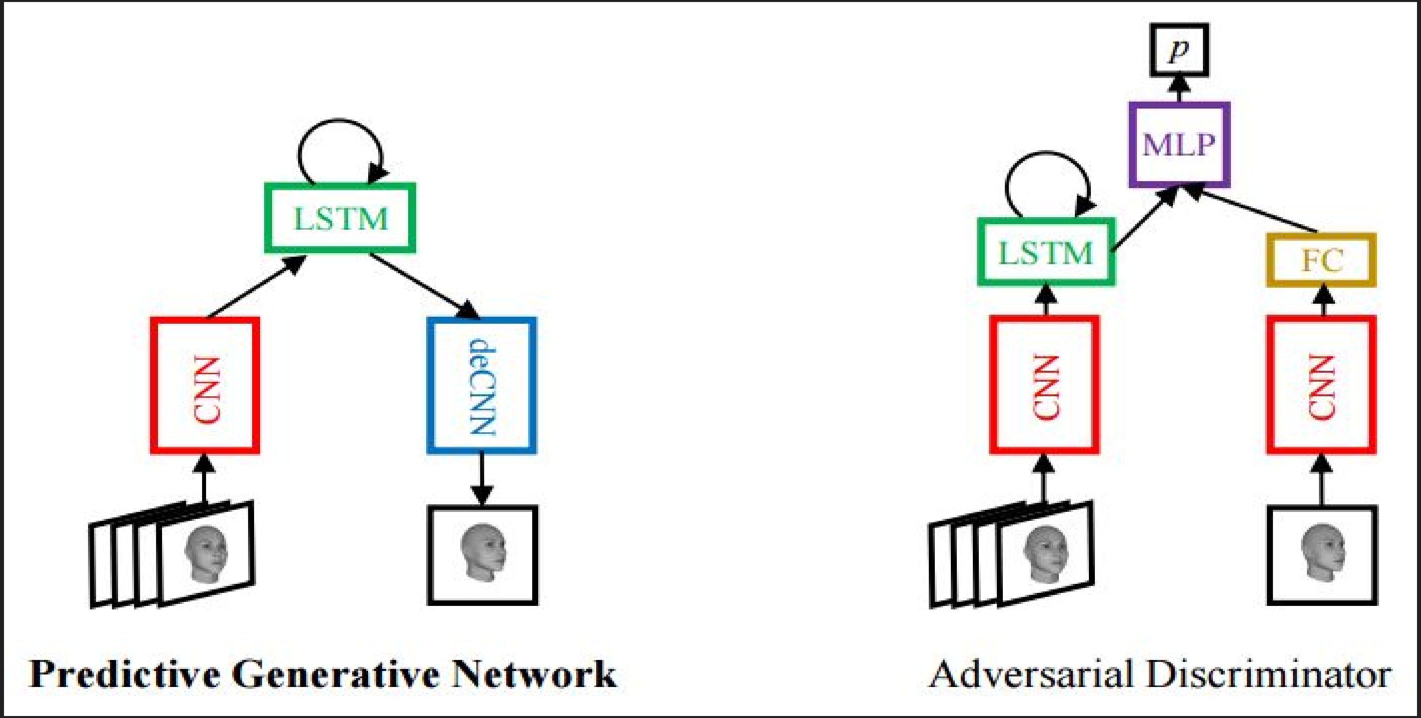


Image taken from https://arxiv.org/abs/1511.06380

Image to image translation

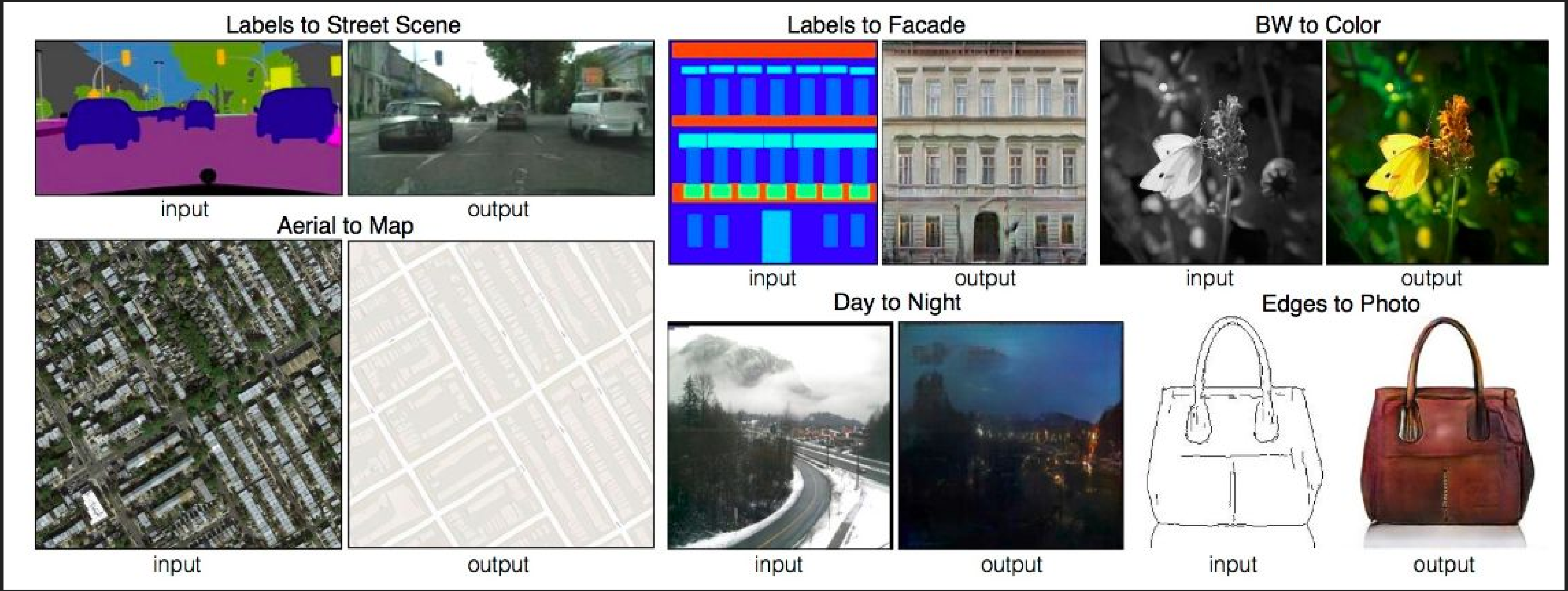


Image taken from https://github.com/phillipi/pix2pix

Real-time image from scribbles

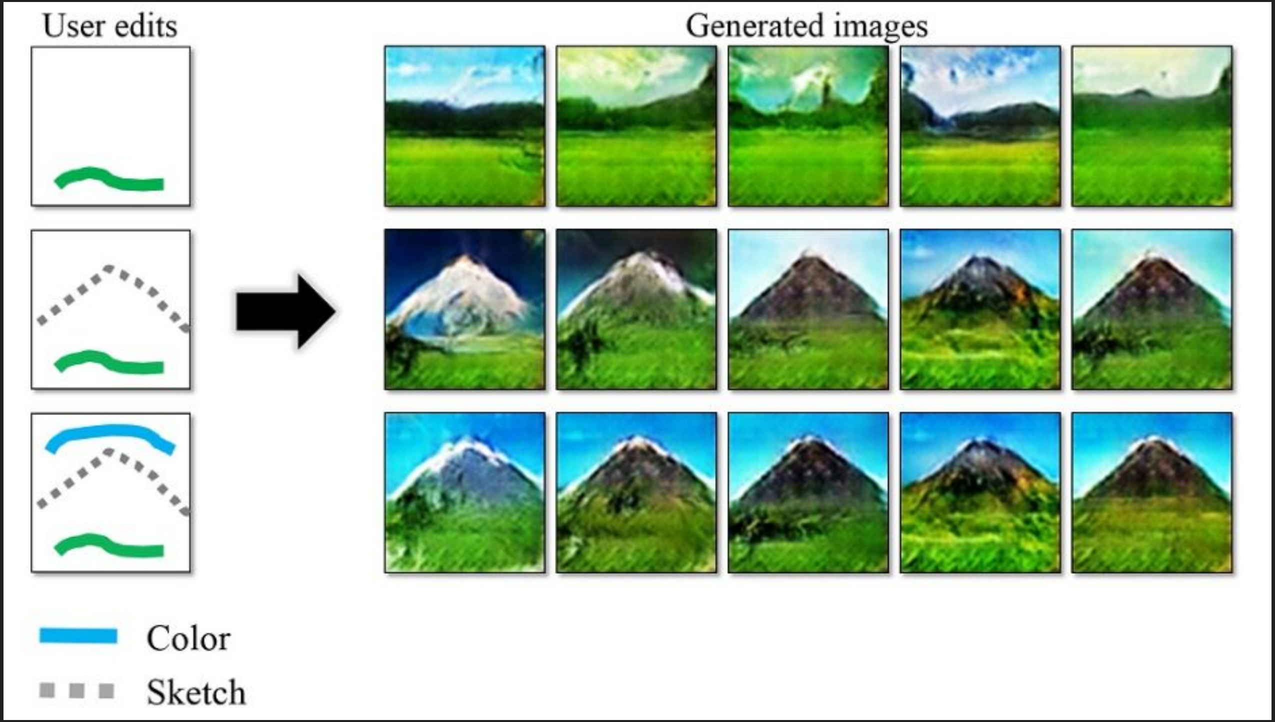


Image taken from http://people.eecs.berkeley.edu/~junyanz/projects/gvm/

How does it work?

Loss function

$$L_D = -E_x \log(D(x)) - E_z \log(1-D(G(z)))$$

$$L_G = -L_D$$

Generator and discriminator are “doing the same thing”.

Problem: gets stuck when generator is very bad.

Instead...

Loss function

$$L_D = -E_x \log(D(x)) - E_z \log(1-D(G(z)))$$

$$L_G = -E_z \log(D(G(z)))$$

Here, generator should make progress even if very bad.

Game theory

Very different setting from the usual. We are not directly optimizing any function.

Instead D and G are playing a game and looking for an equilibrium.

One can usually prove that this equilibrium is what we want, but does local best response dynamic guarantee convergence? Hard problem, some positive results.

Non-convergence

Consider $L_D = xy$ (and $L_G = -xy$), where G optimizes x and D optimizes y.

This can go very wrong. GD can enter a stable orbit and never converge to $(0,0)$ which is the equilibrium.

Similar behaviour has been observed for actual GANs.

Mode collapse

Question: Suppose that the discriminator is fixed. What should the generator do?

Answer: Output the same answer maximizing the score assigned by discriminator.

This actually happens in practice - mode collapse. Serious problem.

Discriminator eventually learns that this is a bad example, and generator moves to a different one. Generator entropy is lost.

Mode collapse

Question: Suppose that the discriminator is fixed. What should the generator do?

Answer: Output the same answer maximizing the score assigned by discriminator.

This actually happens in practice - mode collapse. Serious problem.

Discriminator eventually learns that this is a bad example, and generator moves to a different one. Generator entropy is lost.

How to train your GAN?

Several papers gives tips & tricks. No universal solution (many papers claim one, who knows…).

Simple examples:

- use batch-norm or other normalization like weight-norm,
- do not use fully connected layers
- use ReLU for the generator and LeakyReLU for discriminator

All of these are from https://arxiv.org/abs/1511.06434.

Also, use labels if you have them.

Advanced techniques

Feature matching: The generator is looking at an intermediate layer and trying to make his statistics on activations look like statistics on real data.

This often stabilizes the learning process.

Mini-batch GAN: Let the discriminator look at the whole mini-batch and use some measure of variation as feature.

This explicitly targets the mode collapse problem. In practice also leads to very “good looking” samples, e.g. MNIST indistinguishable from real for humans.

Both of these are from https://arxiv.org/abs/1606.03498.