



mpi max planck institut
informatik



UNIVERSITÄT
DES
SAARLANDES

High Level Computer Vision

Adversarial Networks & Applications

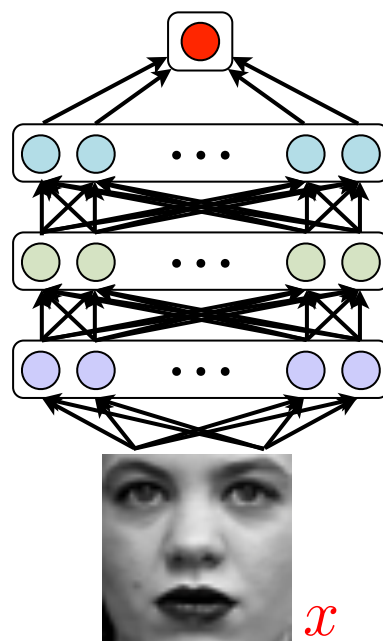
Bernt Schiele - schiele@mpi-inf.mpg.de

Mario Fritz - mfritz@mpi-inf.mpg.de

<https://www.mpi-inf.mpg.de/hlcv>

Discriminative deep learning

- Recipe for success



Generative modeling

- Have training examples $\mathbf{x} \sim \mathbf{p}_{\text{data}}(\mathbf{x})$
- Want a model that can draw samples: $\mathbf{x} \sim \mathbf{p}_{\text{model}}(\mathbf{x})$
- Where $\mathbf{p}_{\text{model}} \approx \mathbf{p}_{\text{data}}$



$\mathbf{x} \sim \mathbf{p}_{\text{data}}(\mathbf{x})$



$\mathbf{x} \sim \mathbf{p}_{\text{model}}(\mathbf{x})$

Generative Adversarial Networks (GANs)

Ian Goodfellow, OpenAI Research Scientist

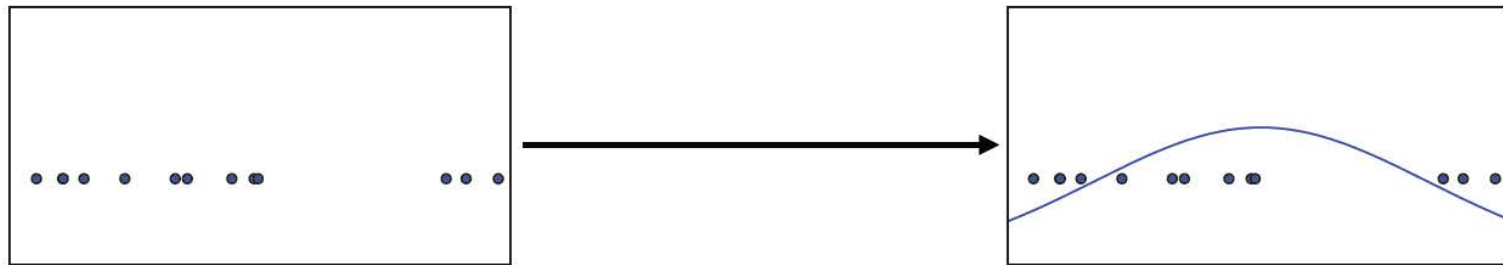
NIPS 2016 tutorial

Barcelona, 2016-12-4

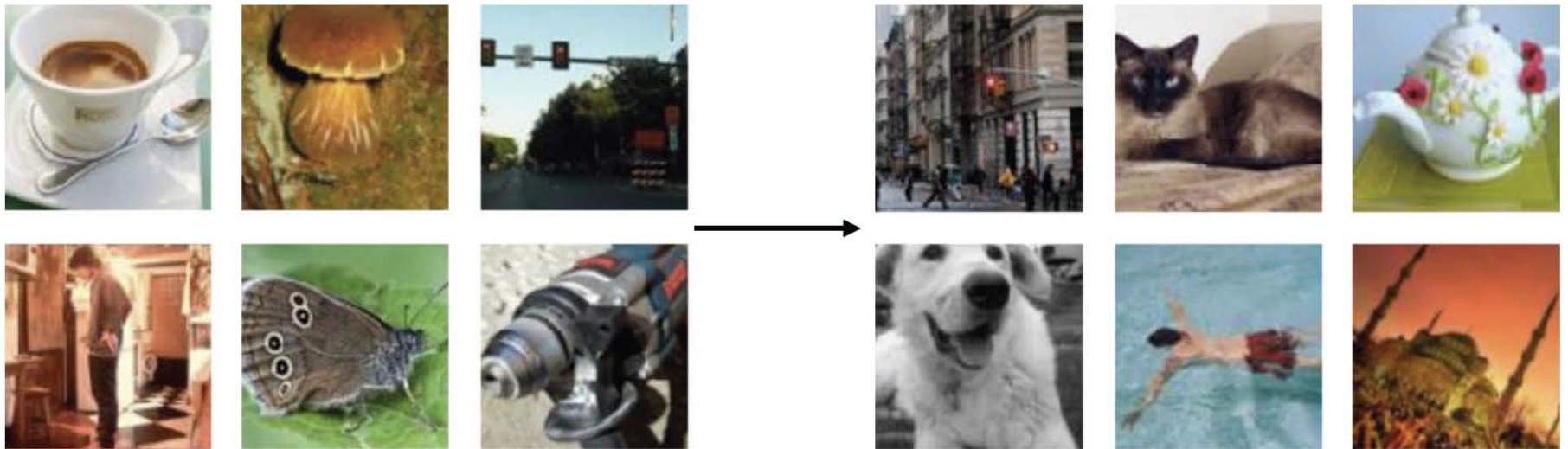
OpenAI

Generative Modeling

- Density estimation



- Sample generation



Training examples

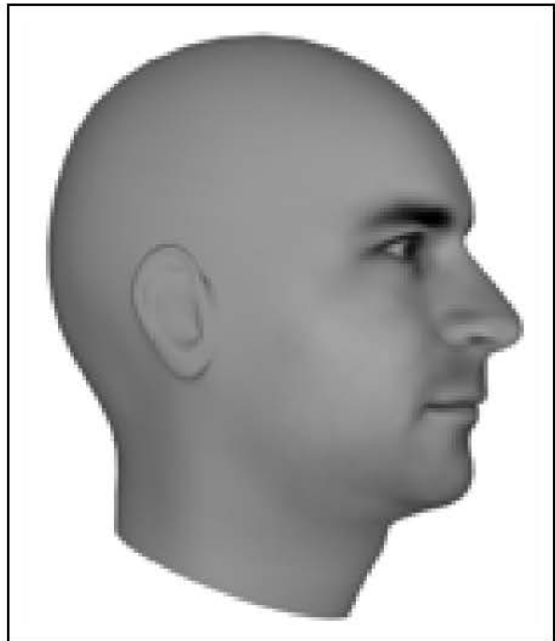
Model samples

Why study generative models?

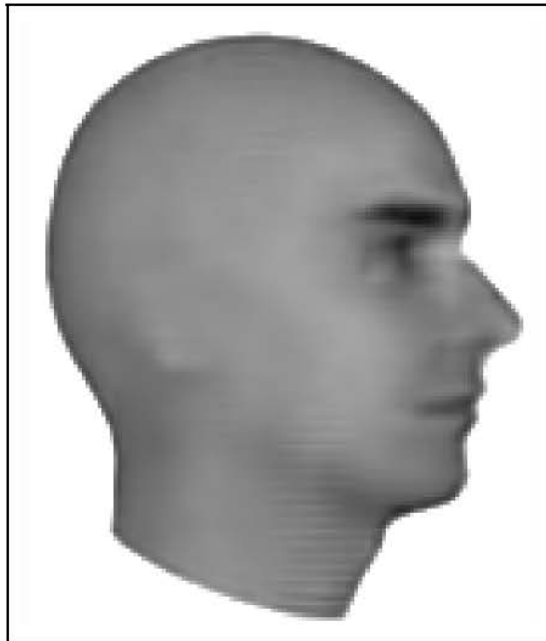
- Excellent test of our ability to use high-dimensional, complicated probability distributions
- Simulate possible futures for planning or simulated RL
- Missing data
 - Semi-supervised learning
- Multi-modal outputs
- Realistic generation tasks

Next Video Frame Prediction

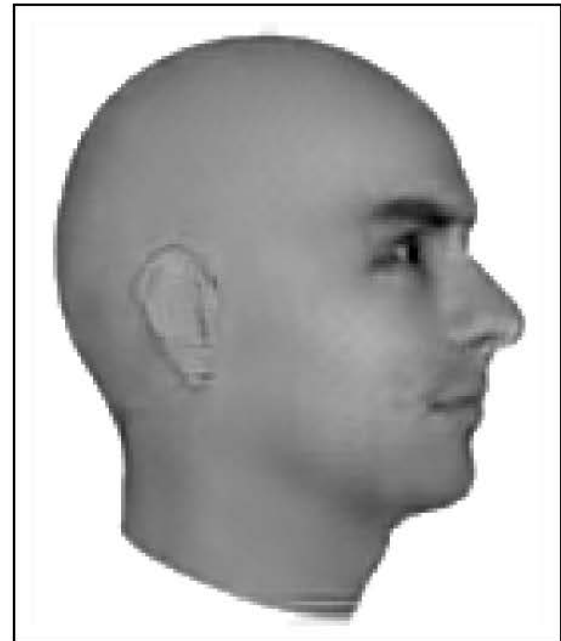
Ground Truth



MSE



Adversarial



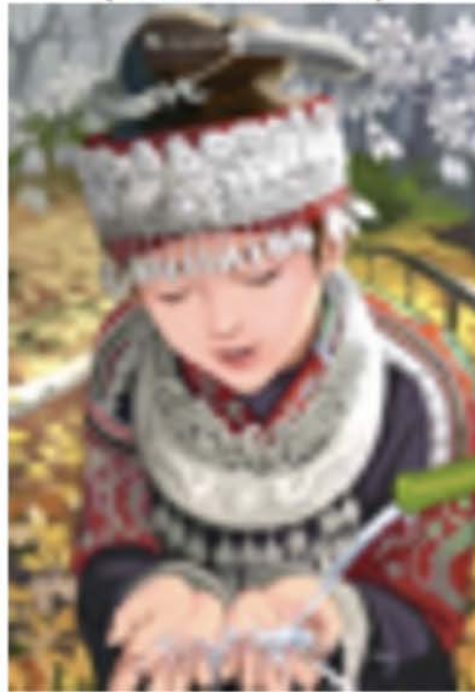
(Lotter et al 2016)

Single Image Super-Resolution

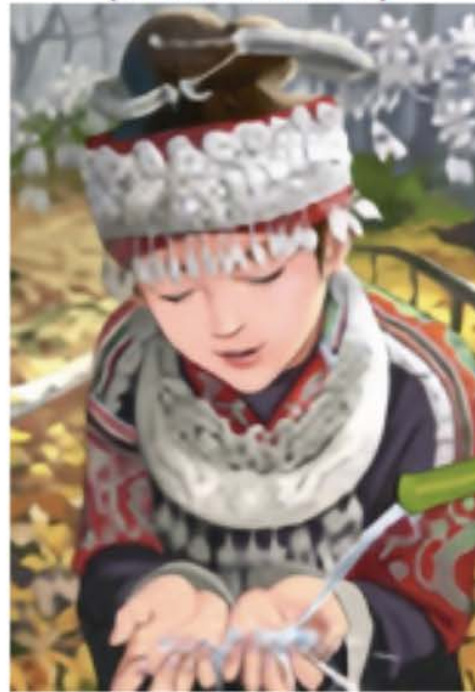
original



bicubic
(21.59dB/0.6423)



SRResNet
(23.44dB/0.7777)

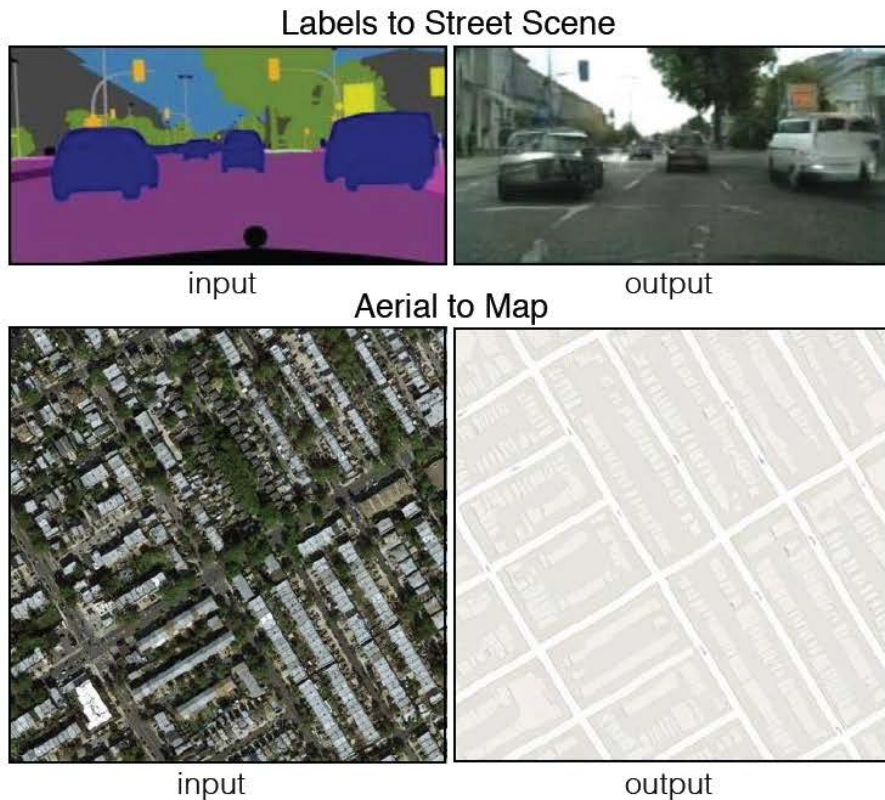


SRGAN
(20.34dB/0.6562)



(Ledig et al 2016)

Image to Image Translation



(Isola et al 2016)

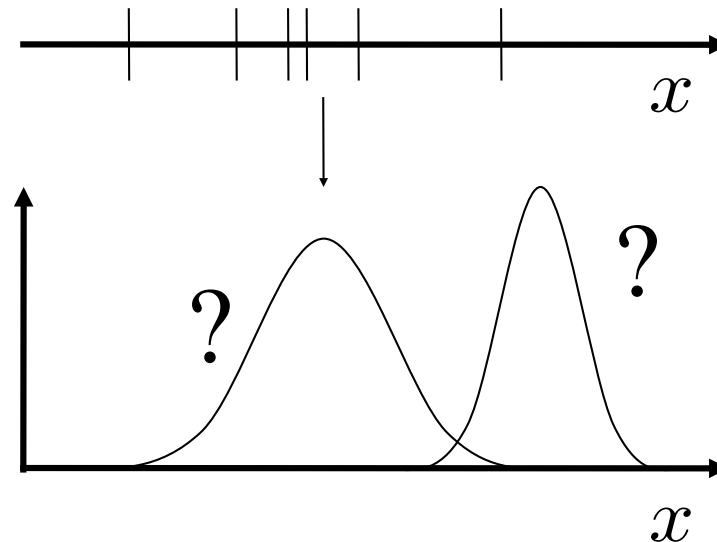
Roadmap

- Why study generative modeling?
- How do generative models work? How do GANs compare to others?
- How do GANs work?
- Tips and tricks
- Research frontiers
- Combining GANs with other methods

Maximum Likelihood Method

- Learning = Estimation of parameter θ (given data X)

$$X = \{x_1, x_2, x_3, \dots, x_N\}$$



- Likelihood of θ
 - ▶ defined as the probability of the data X has been generated from the distribution with parameter θ

- ▶ Likelihood $L(\theta)$:

$$L(\theta) = p(X|\theta)$$

Maximum Likelihood Method

- Calculation of Likelihood

- ▶ a single datapoint: $p(x_n | \theta)$
- ▶ assume: all N data points are independent
 - data points are i.i.d = independent identically distributed

$$L(\theta) = p(X | \theta) = \prod_{n=1}^N p(x_n | \theta)$$

- often used is **log-likelihood**:

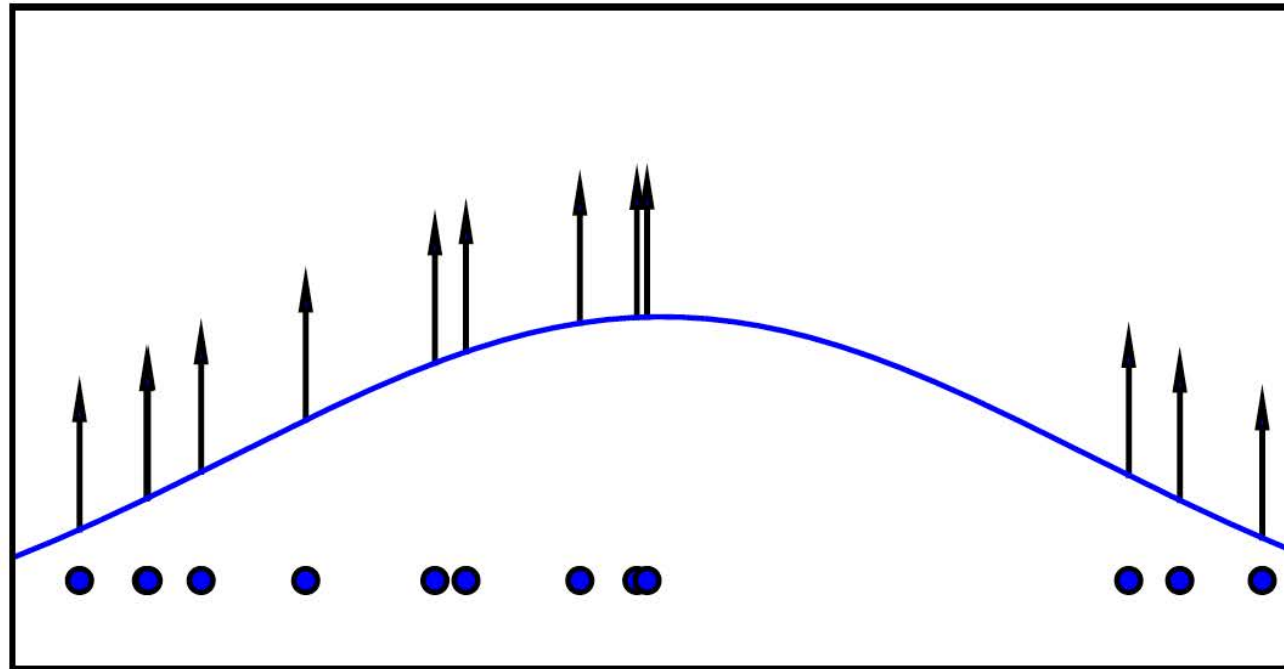
- ▶ often easier to calculate and manipulate

$$E = -\ln L(\theta) = -\sum_{n=1}^N \ln p(x_n | \theta)$$

- parameter estimation= learning

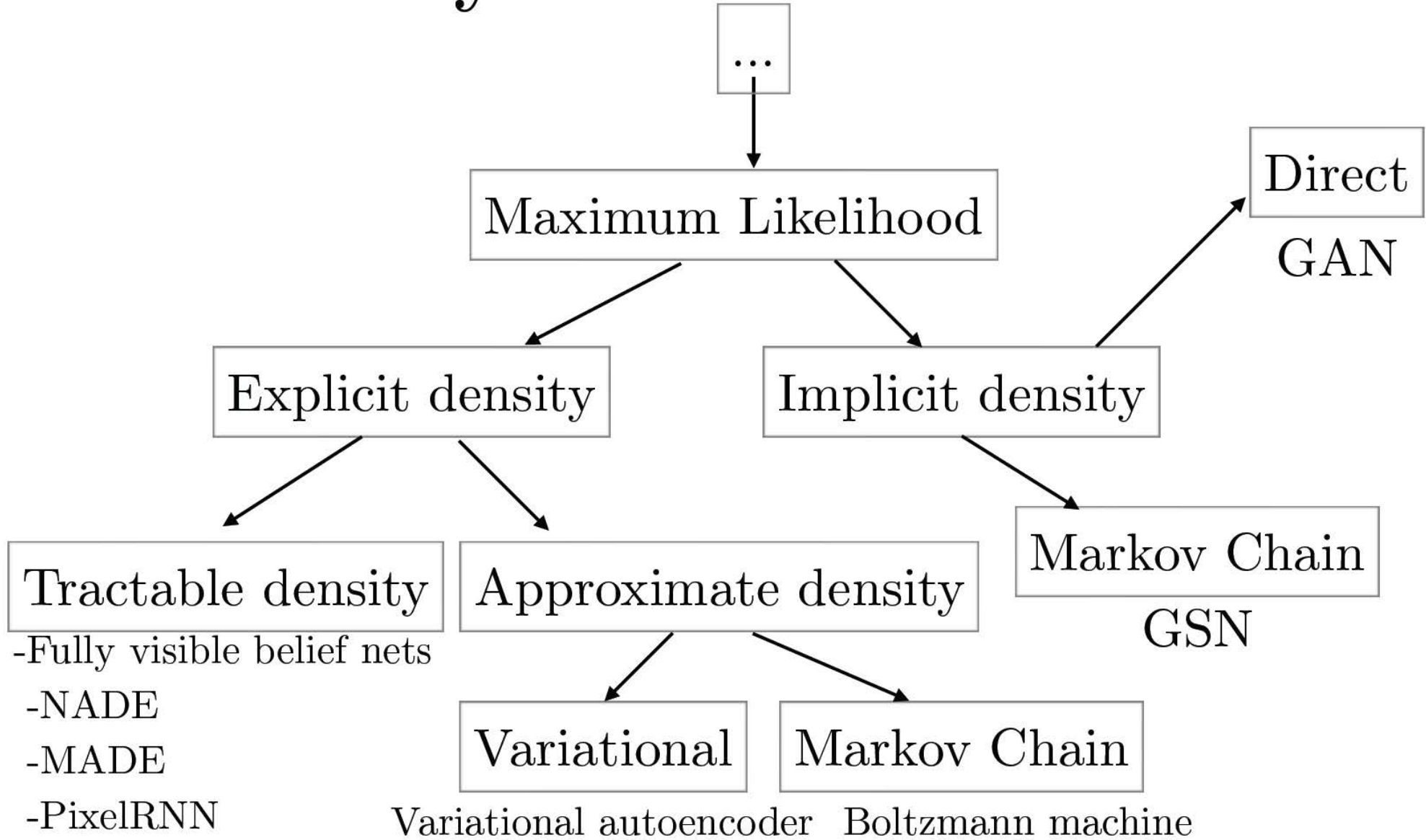
- ▶ maximize likelihood or log-likelihood or
- ▶ minimize negative log-likelihood

Maximum Likelihood



$$\theta^* = \arg \max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log p_{\text{model}}(\mathbf{x} \mid \theta)$$

Taxonomy of Generative Models



-Fully visible belief nets
-NADE
-MADE
-PixelRNN
-Change of variables
models (nonlinear ICA)

Variational autoencoder Boltzmann machine

Fully Visible Belief Nets

- Explicit formula based on chain (Frey et al, 1996)

rule:

$$p_{\text{model}}(\mathbf{x}) = p_{\text{model}}(x_1) \prod_{i=2}^n p_{\text{model}}(x_i \mid x_1, \dots, x_{i-1})$$

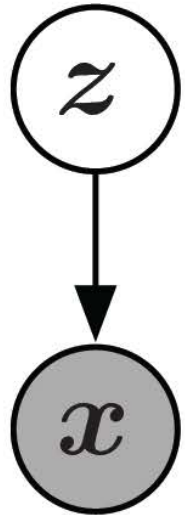
- Disadvantages:
 - $O(n)$ sample generation cost
 - Generation not controlled by a latent code



PixelCNN elephants
(van den Ord et al 2016)

Variational Autoencoder

(Kingma and Welling 2013, Rezende et al 2014)



$$\log p(\mathbf{x}) \geq \log p(\mathbf{x}) - D_{\text{KL}}(q(\mathbf{z}) \| p(\mathbf{z} | \mathbf{x}))$$
$$= \mathbb{E}_{\mathbf{z} \sim q} \log p(\mathbf{x}, \mathbf{z}) + H(q)$$



CIFAR-10 samples
(Kingma et al 2016)

Disadvantages:

- Not asymptotically consistent unless q is perfect
- Samples tend to have lower quality

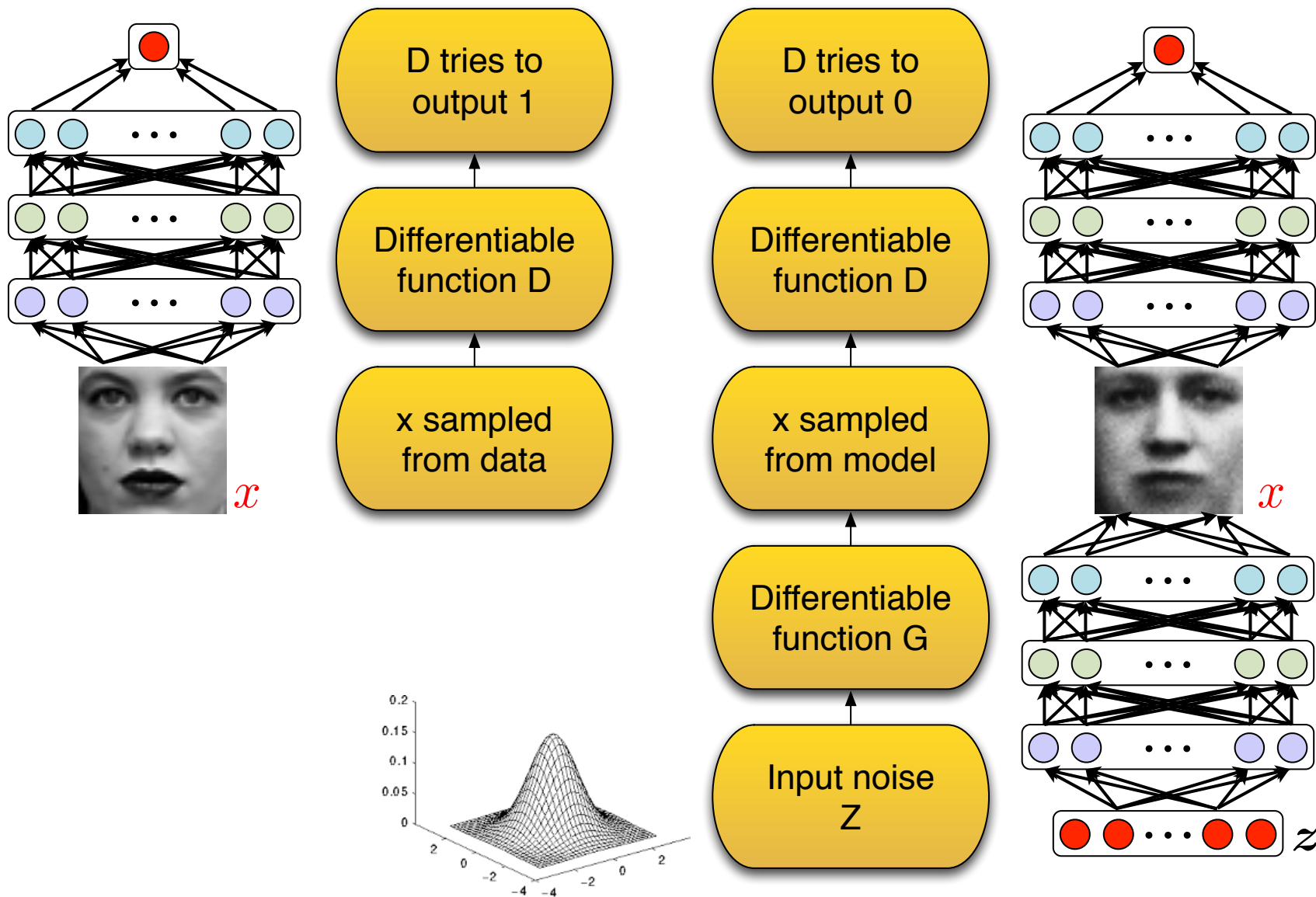
GANs

- Use a latent code
- Asymptotically consistent (unlike variational methods)
- No Markov chains needed
- Often regarded as producing the best samples
 - No good way to quantify this

Roadmap

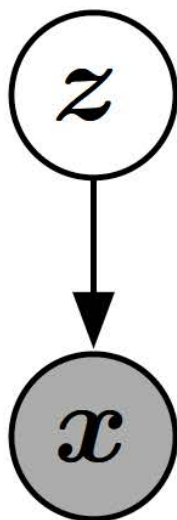
- Why study generative modeling?
- How do generative models work? How do GANs compare to others?
- How do GANs work?
- Tips and tricks
- Research frontiers
- Combining GANs with other methods

Adversarial nets framework



Generator Network

$$x = G(z; \theta^{(G)})$$



- Must be differentiable
- No invertibility requirement
- Trainable for any size of z
- Some guarantees require z to have higher dimension than x
- Can make x conditionally Gaussian given z but need not do so

Training Procedure

- Use SGD-like algorithm of choice (Adam) on two minibatches simultaneously:
 - A minibatch of training examples
 - A minibatch of generated samples
- Optional: run k steps of one player for every step of the other player.

Minimax Game

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\log D(\mathbf{x}) - \frac{1}{2}\mathbb{E}_{\mathbf{z}}\log(1 - D(G(\mathbf{z})))$$

$$J^{(G)} = -J^{(D)}$$

- Equilibrium is a saddle point of the discriminator loss
- Resembles Jensen-Shannon divergence
- Generator minimizes the log-probability of the discriminator being correct

Exercise 1

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\log D(\mathbf{x}) - \frac{1}{2}\mathbb{E}_{\mathbf{z}}\log(1 - D(G(\mathbf{z})))$$

$$J^{(G)} = -J^{(D)}$$

- What is the solution to $D(x)$ in terms of p_{data} and $p_{\text{generator}}$?
- What assumptions are needed to obtain this solution?

Solution

- Assume both densities are nonzero everywhere
 - If not, some input values x are never trained, so some values of $D(x)$ have undetermined behavior.
- Solve for where the functional derivatives are zero:

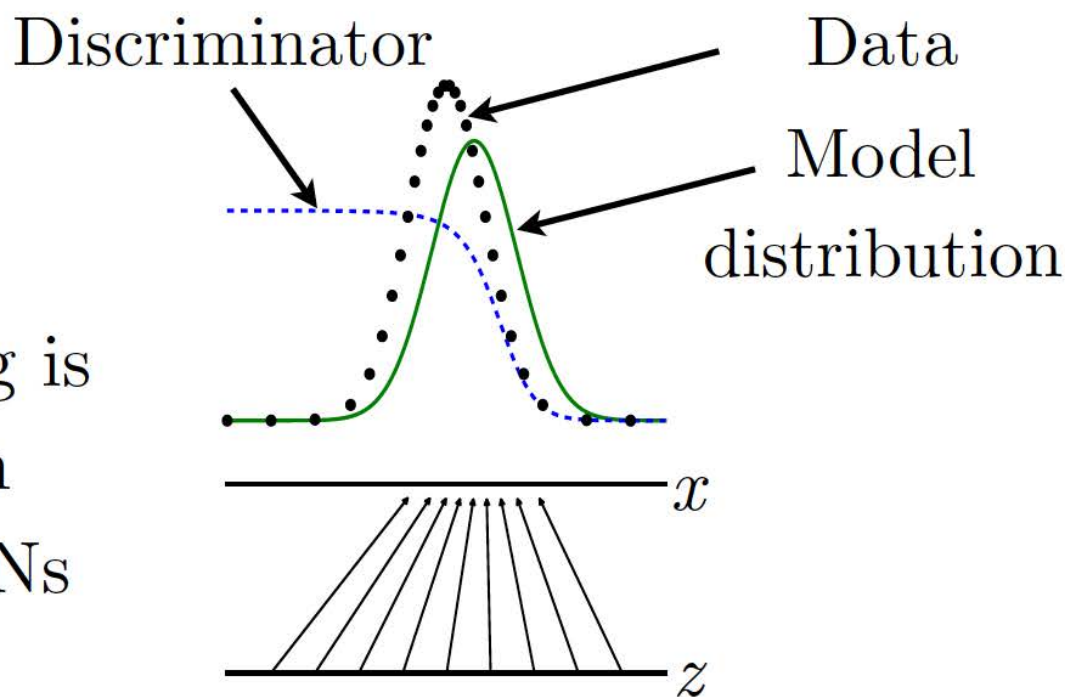
$$\frac{\delta}{\delta D(\mathbf{x})} J^{(D)} = 0$$

Discriminator Strategy

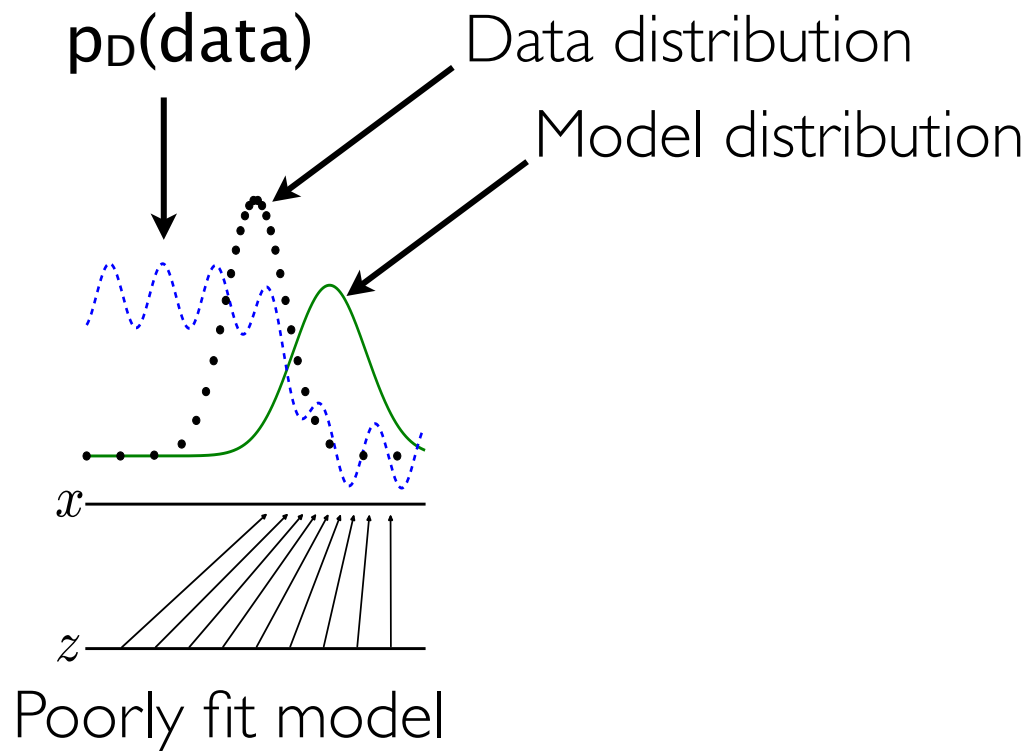
Optimal $D(\mathbf{x})$ for any $p_{\text{data}}(\mathbf{x})$ and $p_{\text{model}}(\mathbf{x})$ is always

$$D(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_{\text{model}}(x)}$$

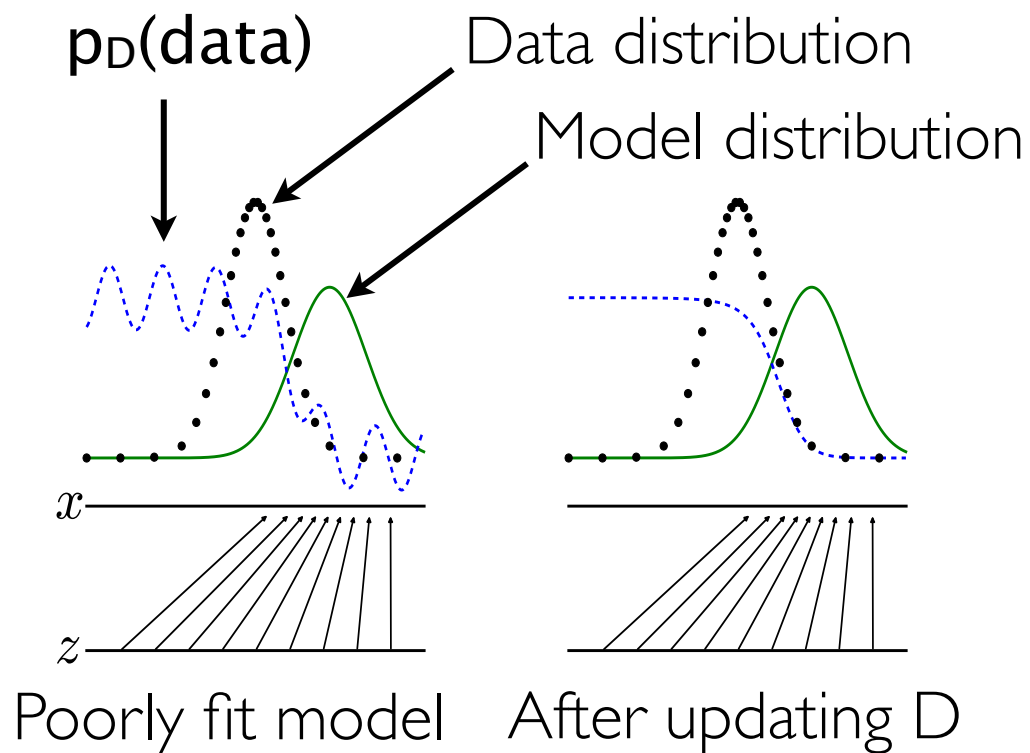
Estimating this ratio using supervised learning is the key approximation mechanism used by GANs



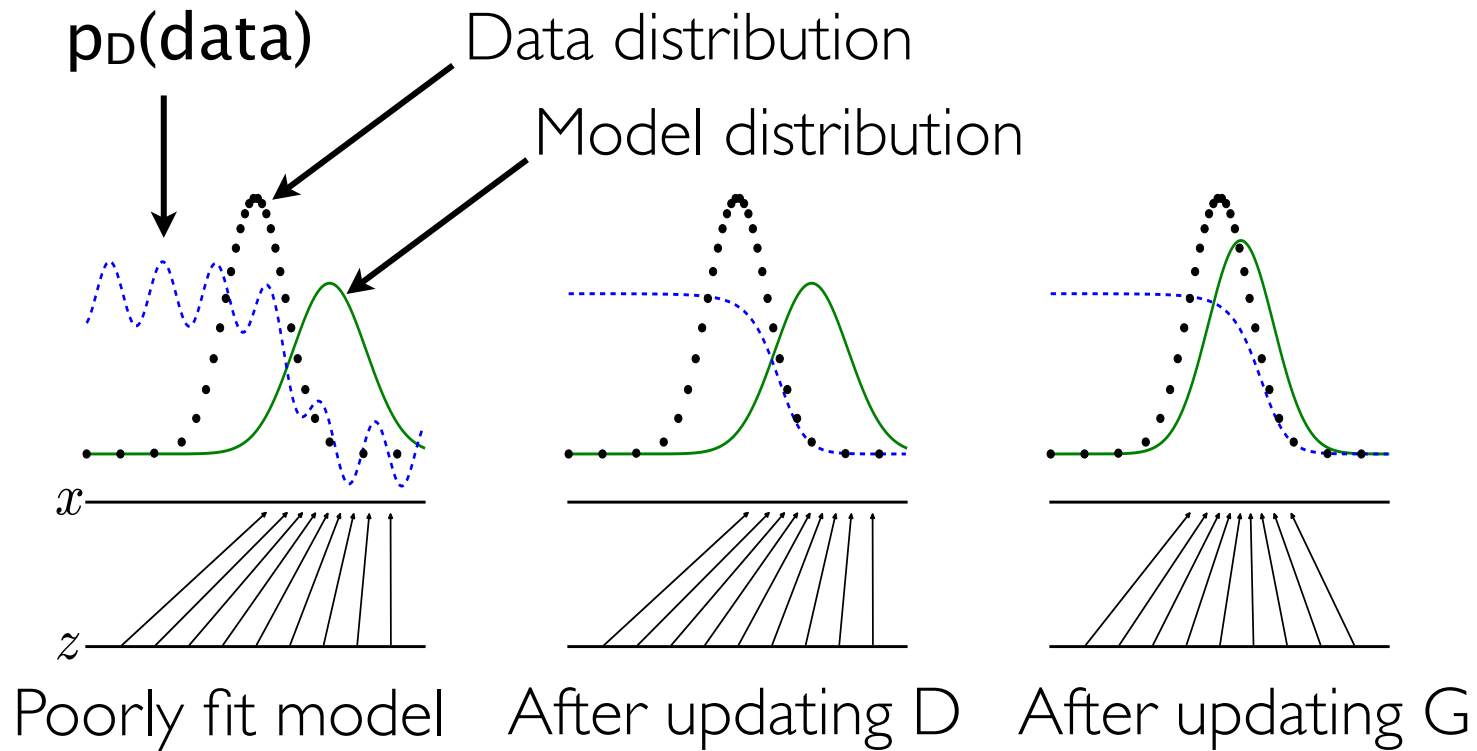
Learning process



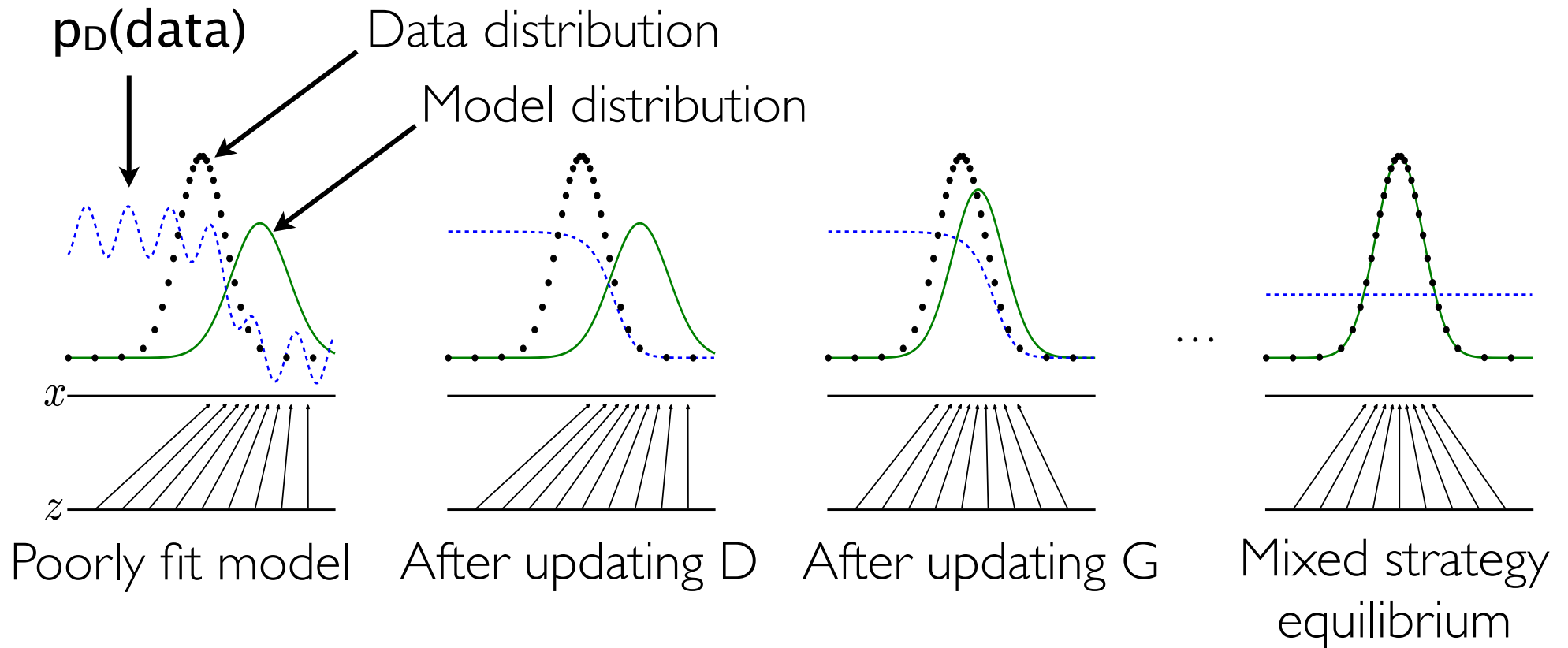
Learning process



Learning process



Learning process



Non-Saturating Game

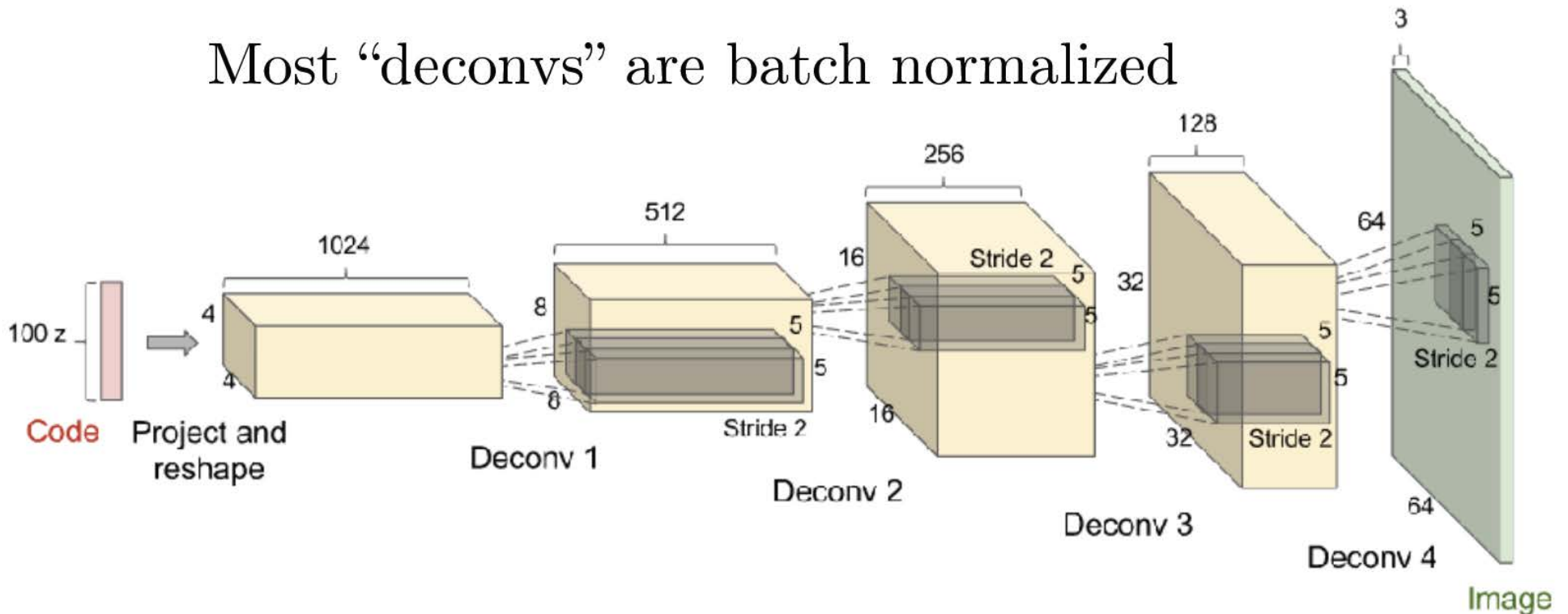
$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\log D(\mathbf{x}) - \frac{1}{2}\mathbb{E}_{\mathbf{z}}\log(1 - D(G(\mathbf{z})))$$

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\mathbf{z}}\log D(G(\mathbf{z}))$$

- Equilibrium no longer describable with a single loss
- Generator maximizes the log-probability of the discriminator being mistaken
- Heuristically motivated; generator can still learn even when discriminator successfully rejects all generator samples

DCGAN Architecture

Most “deconvs” are batch normalized



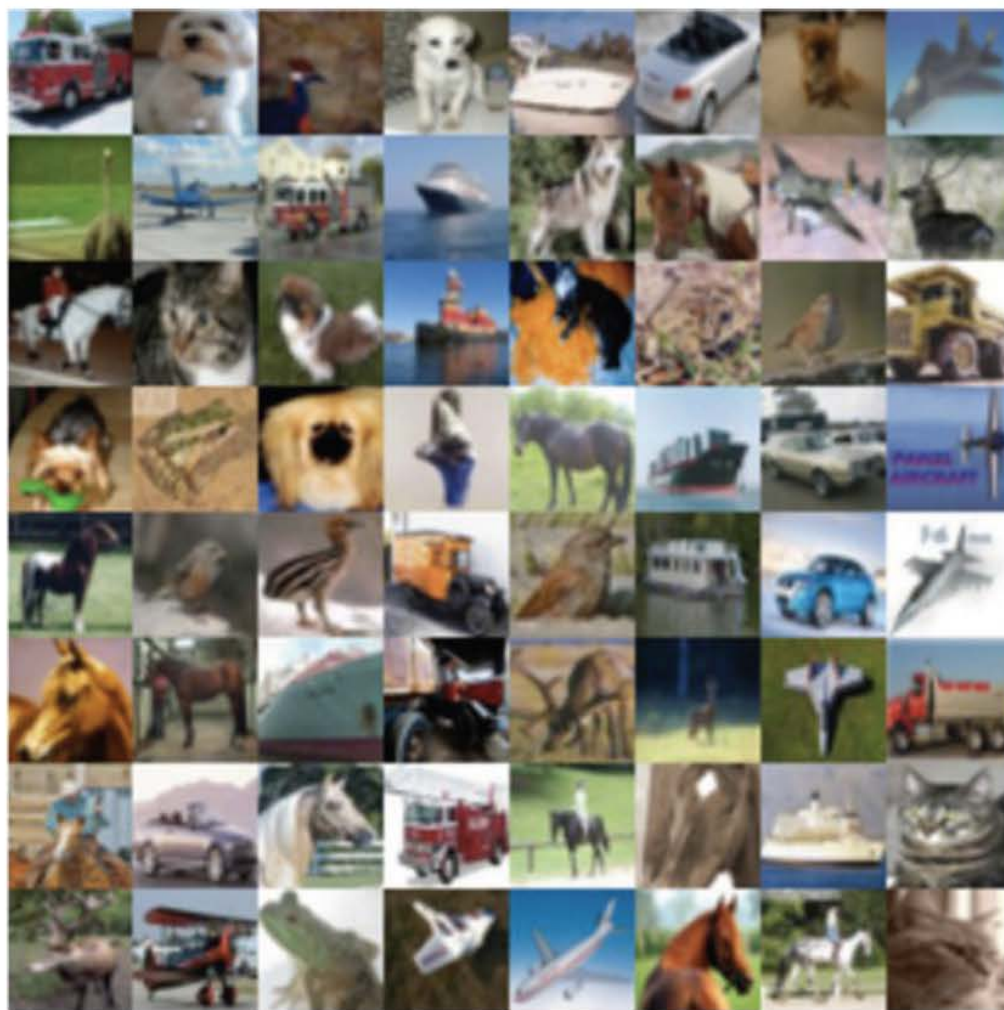
(Radford et al 2015)

DCGANs for LSUN Bedrooms

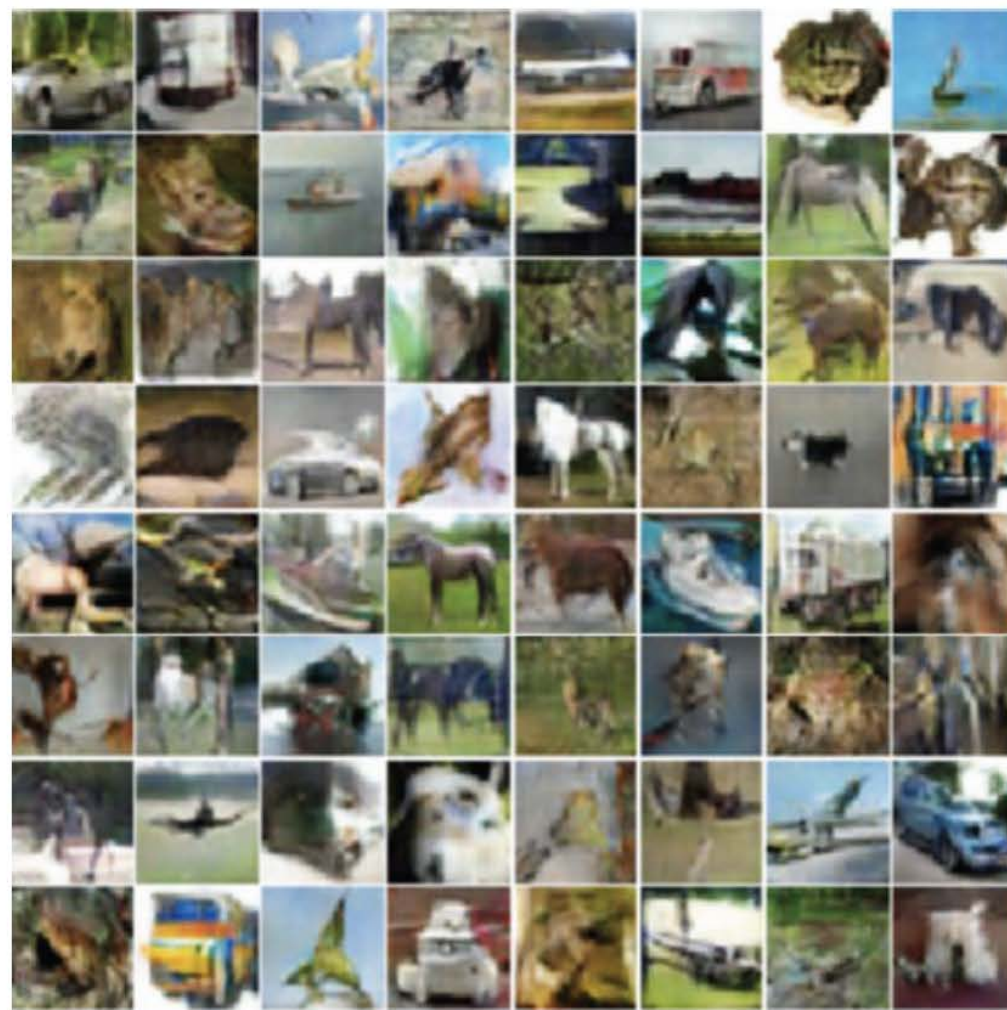


(Radford et al 2015)

Minibatch GAN on CIFAR



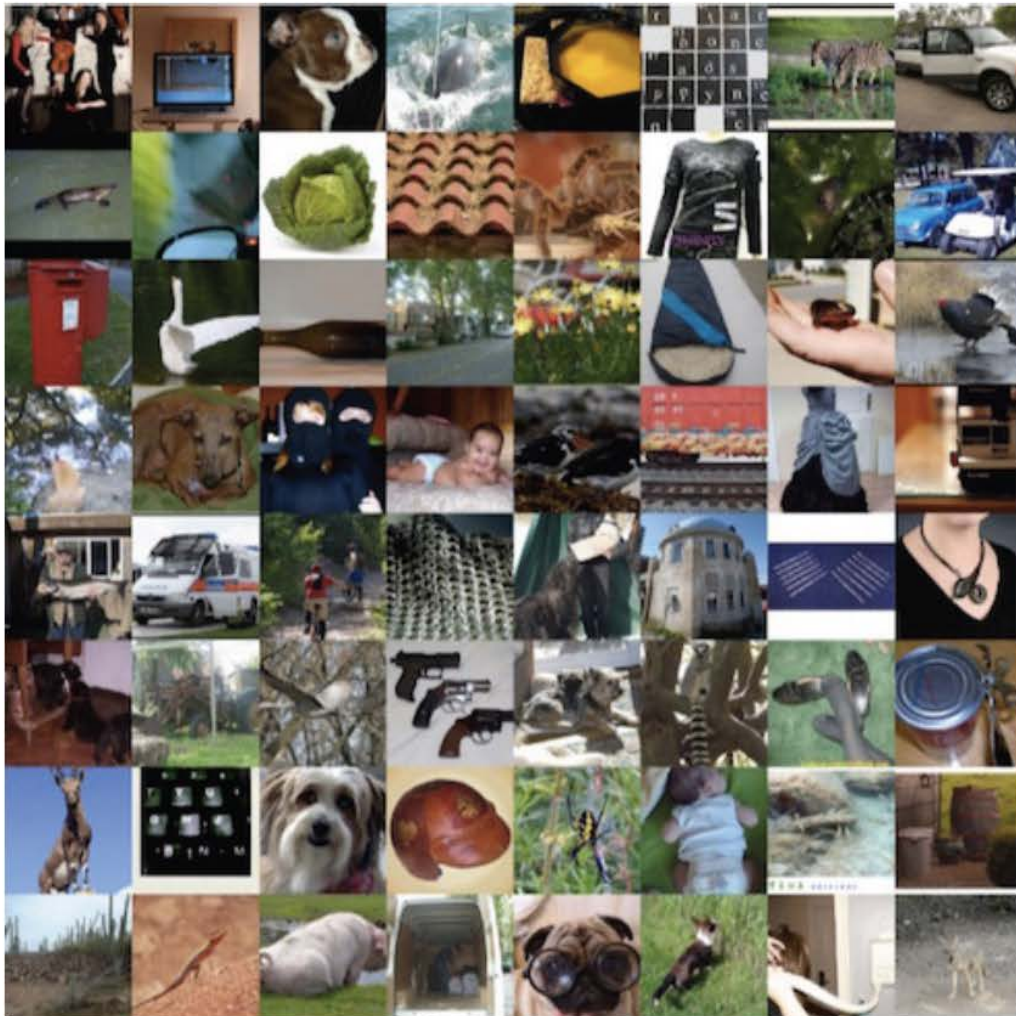
Training Data



Samples

(Salimans et al 2016)

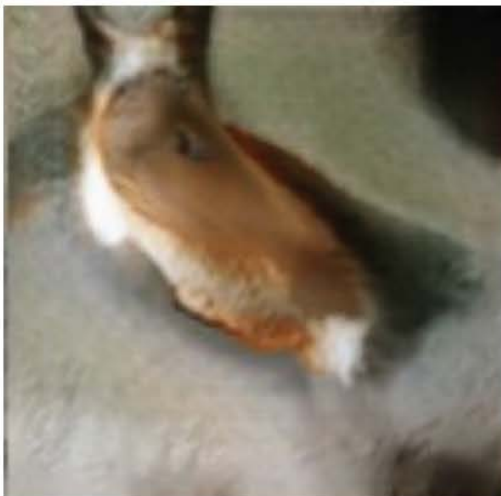
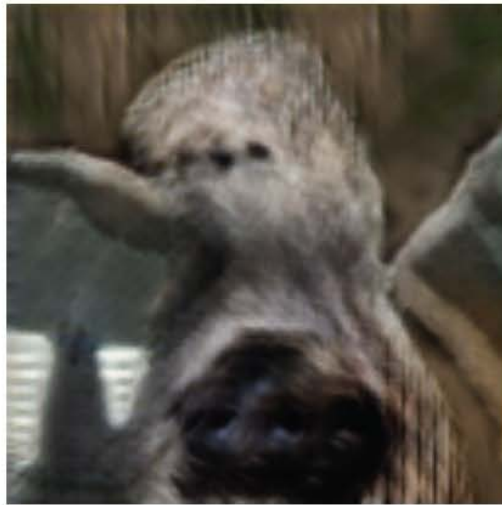
Minibatch GAN on ImageNet



(Salimans et al 2016)

(Goodfellow 2016)

Cherry-Picked Results



Problems with Counting



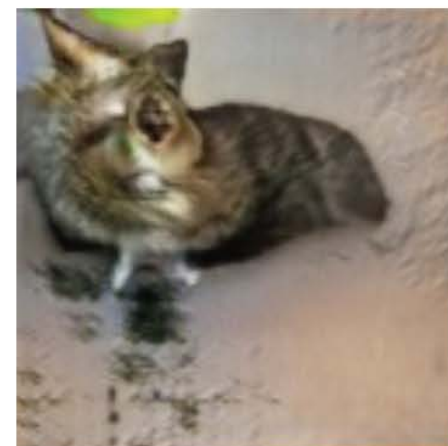
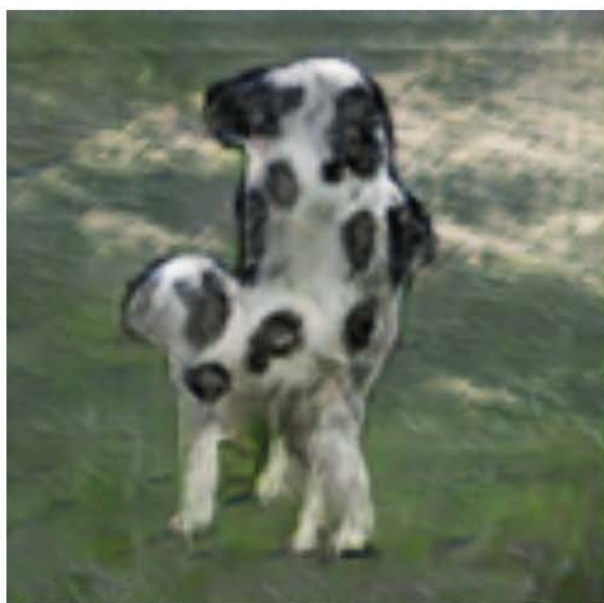
Problems with Perspective



This one is real



Problems with Global Structure



Plug and Play Generative Models

- New state of the art generative model (Nguyen et al 2016) released days before NIPS
- Generates 227×227 realistic images from all ImageNet classes
- Combines adversarial training, moment matching, denoising autoencoders, and Langevin sampling

PPGN Samples



redshank

ant

monastery



volcano

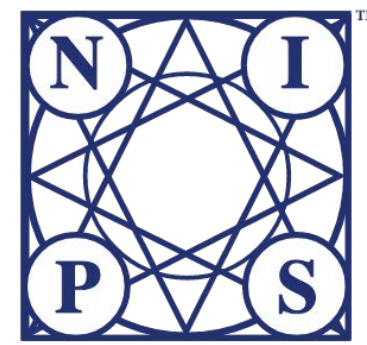
(Nguyen et al 2016)

Basic idea

- Langevin sampling repeatedly adds noise and gradient of $\log p(x,y)$ to generate samples (Markov chain)
- Denoising autoencoders estimate the required gradient
- Use a special denoising autoencoder that has been trained with multiple losses, including a GAN loss, to obtain best results

Conclusion

- GANs are generative models that use supervised learning to approximate an intractable cost function
- GANs can simulate many cost functions, including the one used for maximum likelihood
- Finding Nash equilibria in high-dimensional, continuous, non-convex games is an important open research problem
- GANs are a key ingredient of PPGNs, which are able to generate compelling high resolution samples from diverse image classes



Learning What and Where to Draw

Scott Reed^{1,3}, Zeynep Akata², Santosh Mohan¹,
Samuel Tenka¹, Bernt Schiele², Honglak Lee¹



1



2



3



Motivation

GT

GAN Samples

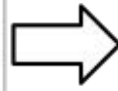
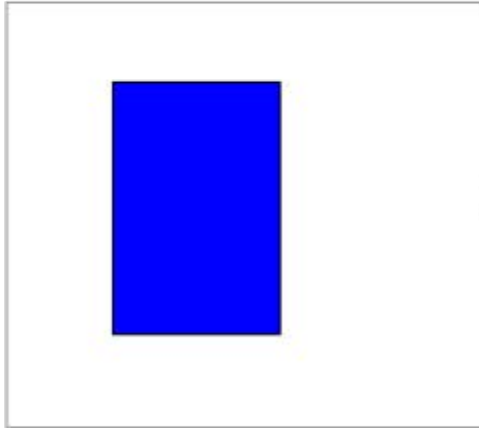
a pitcher is
about to throw
the ball to the
batter.



What object is meant to be drawn here?
Can we control its location?

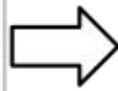
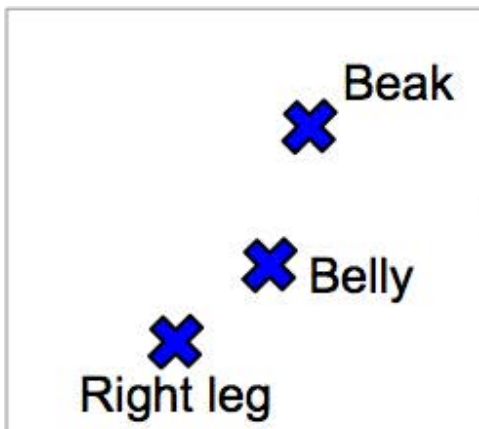
Idea: condition on location in addition to text

1. Bounding box



This bird is completely black.

1. Keypoints, e.g. 15 parts of a bird



This bird is bright blue.

Background: Generative Adversarial Networks

[1] Goodfellow, Ian, et al. "Generative adversarial nets." Advances in Neural Information Processing Systems. 2014.

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

- The discriminator D tries to distinguish real training data from synthetic images.

[1] Goodfellow, Ian, et al. "Generative adversarial nets." Advances in Neural Information Processing Systems. 2014.

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

- The generator G tries to fool D .

[1] Goodfellow, Ian, et al. "Generative adversarial nets." Advances in Neural Information Processing Systems. 2014.

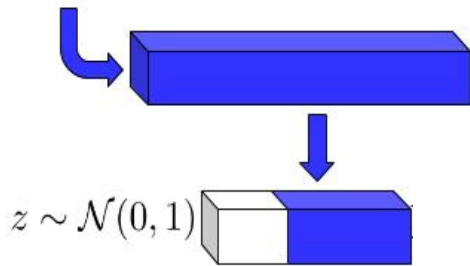
Text-conditional GAN

$$\min_G \max_D V(D, G) = \mathbb{E}_{x, \mathbf{t} \sim p_{data}(x, \mathbf{t})} [\log D(x, \mathbf{t})] + \mathbb{E}_{z \sim p_z(z), \mathbf{t} \sim p_{data}(t)} [\log(1 - D(G(z, \mathbf{t})))]$$

- The discriminator D tries to distinguish real **(text, image) pairs** from synthetic.
- The generator G tries to fool D .

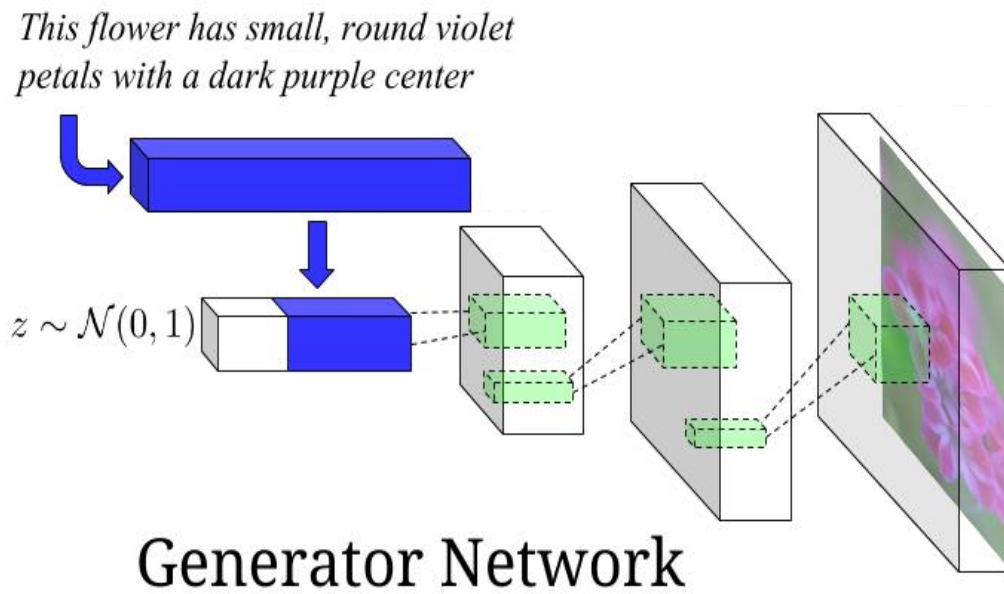
Text-conditional GAN

*This flower has small, round violet
petals with a dark purple center*



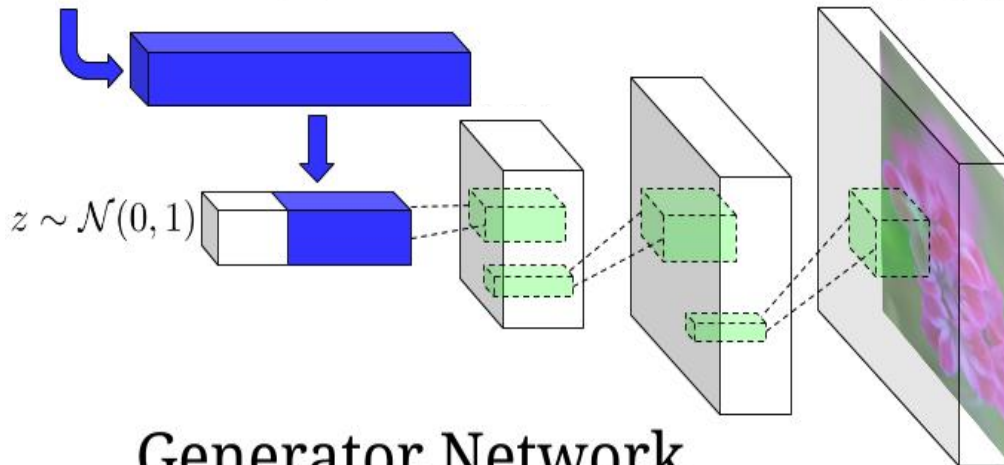
Generator Network

Text-conditional GAN



Text-conditional GAN

This flower has small, round violet petals with a dark purple center



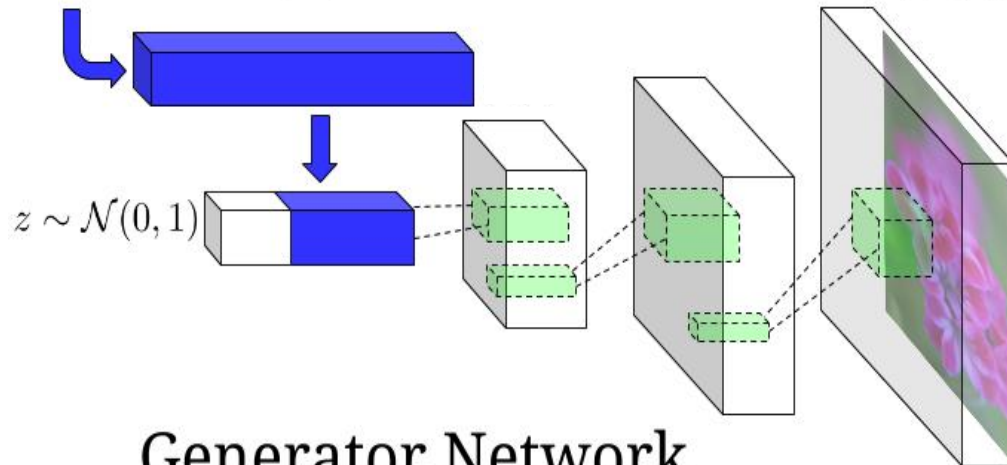
Generator Network

This flower has small, round violet petals with a dark purple center

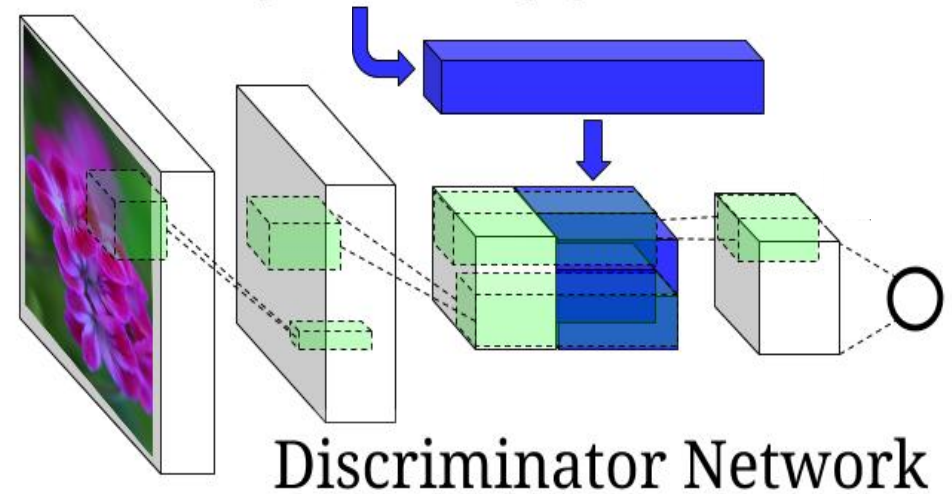


Text-conditional GAN

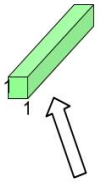
This flower has small, round violet petals with a dark purple center



This flower has small, round violet petals with a dark purple center



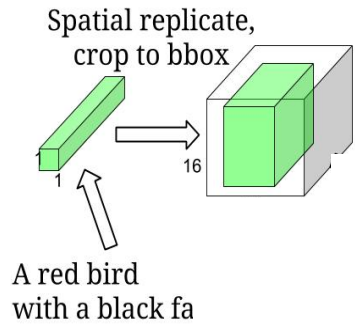
Conditioning on bounding box



A red bird
with a black face

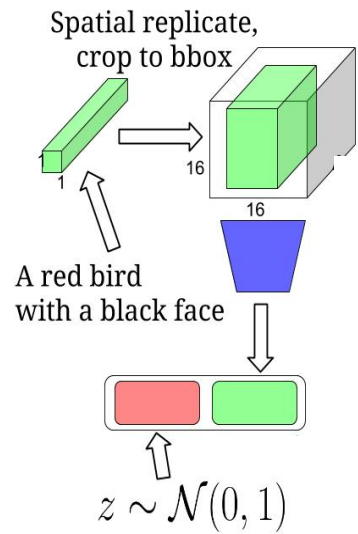
Generator Network

Conditioning on bounding box



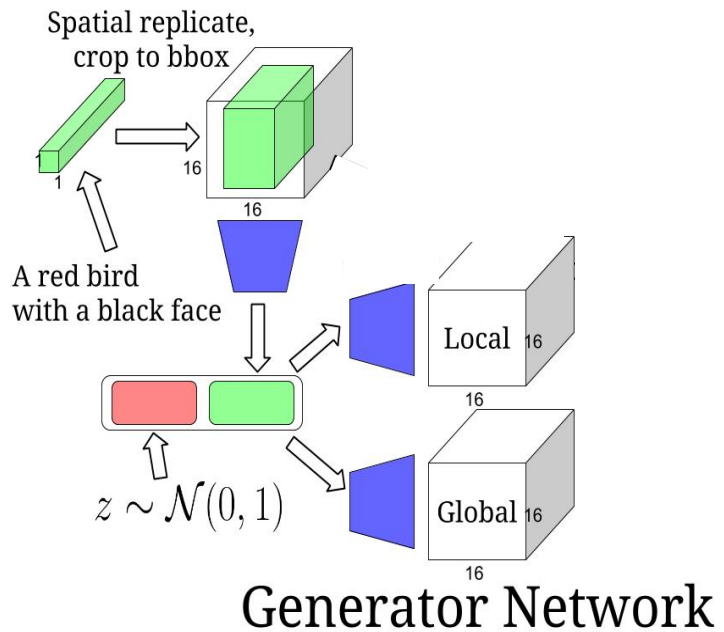
Generator Network

Conditioning on bounding box

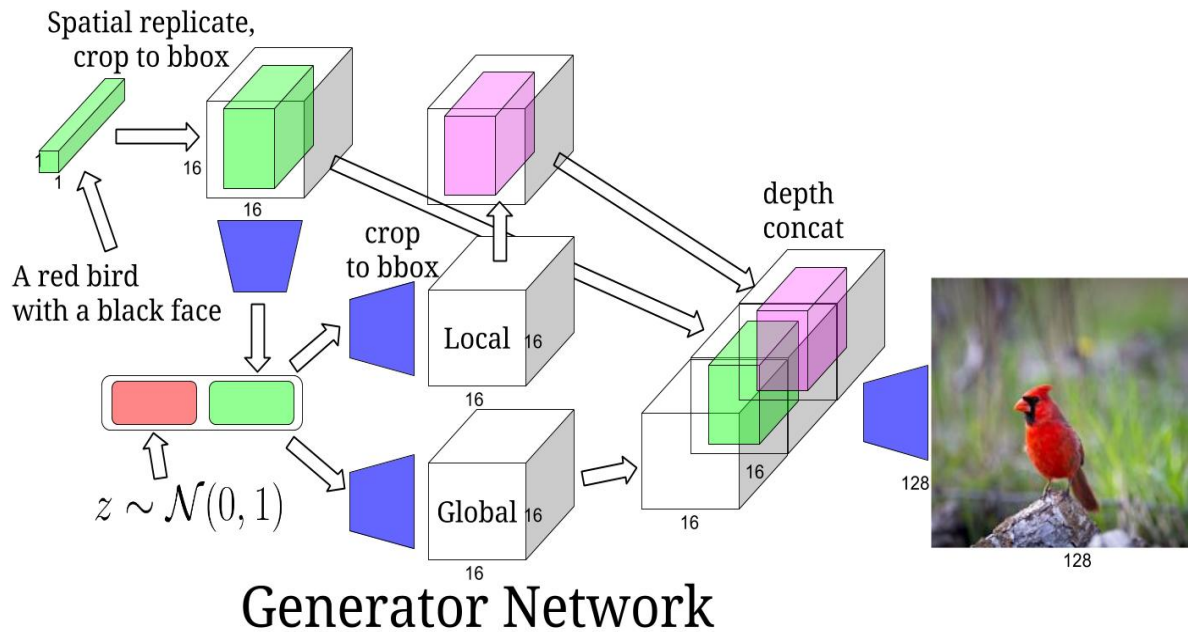


Generator Network

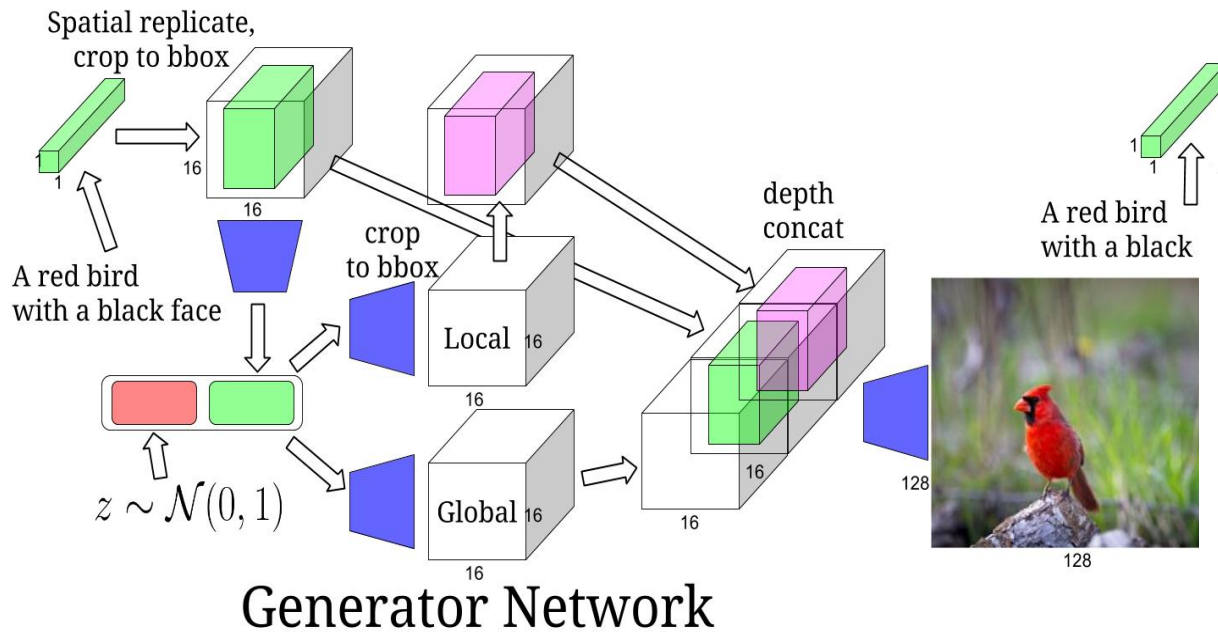
Conditioning on bounding box



Conditioning on bounding box

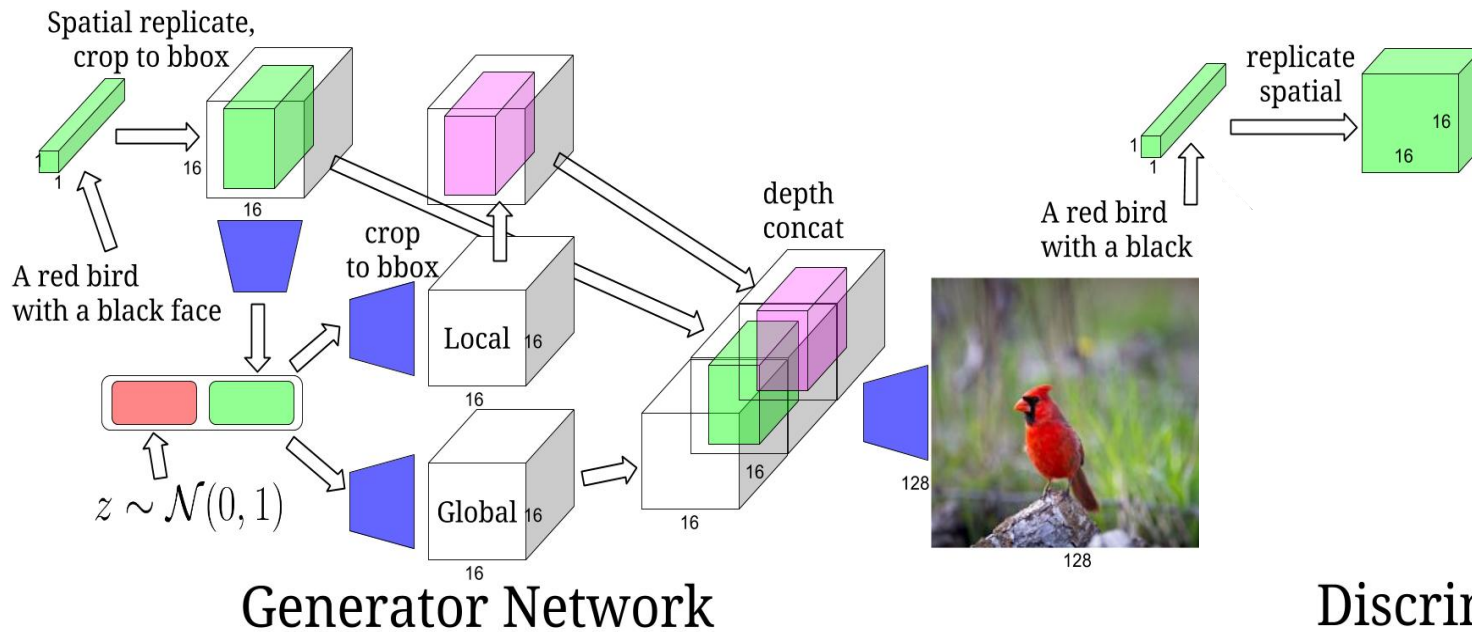


Conditioning on bounding box

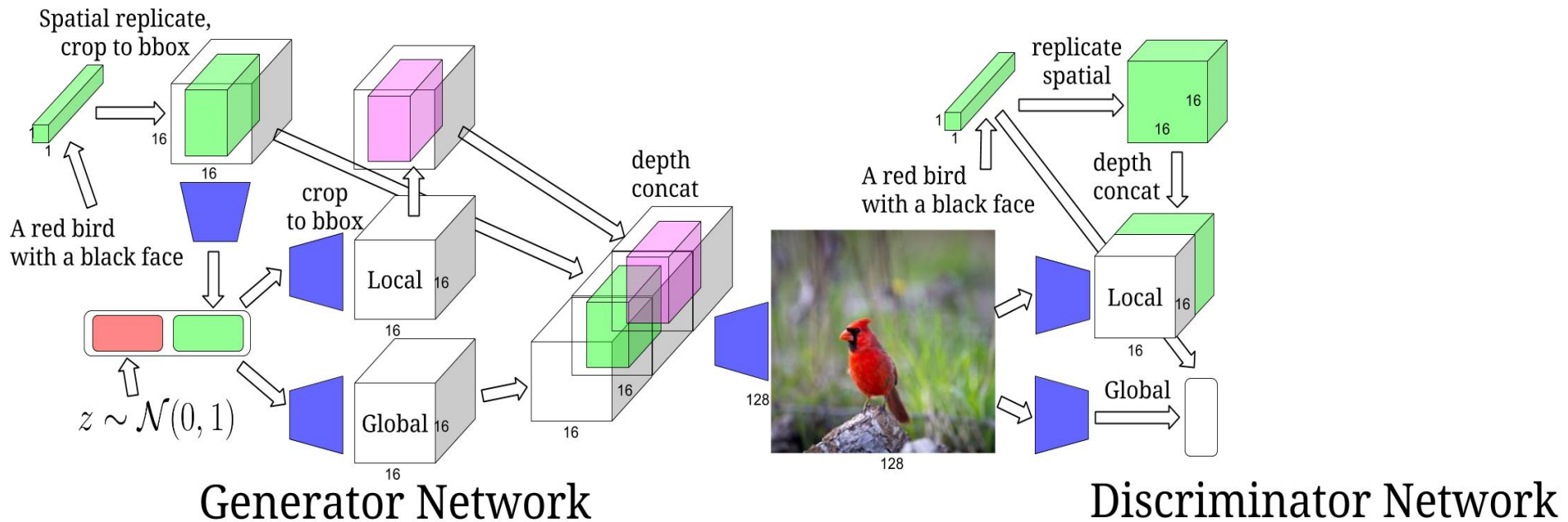


Discriminator Network

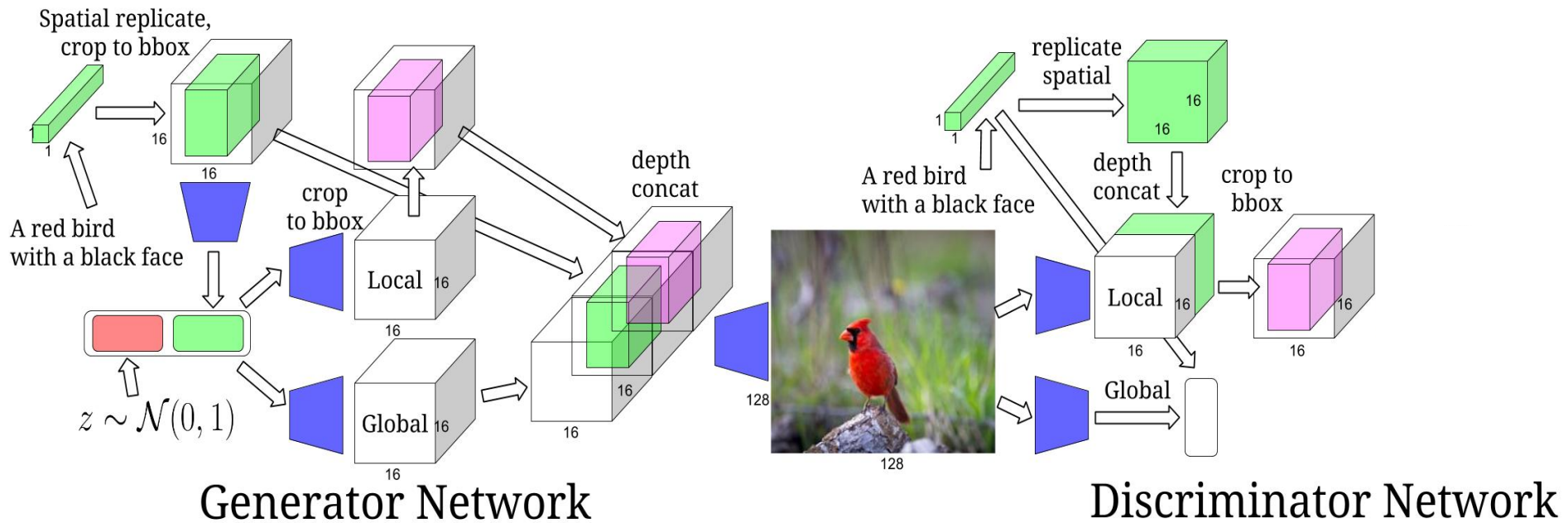
Conditioning on bounding box



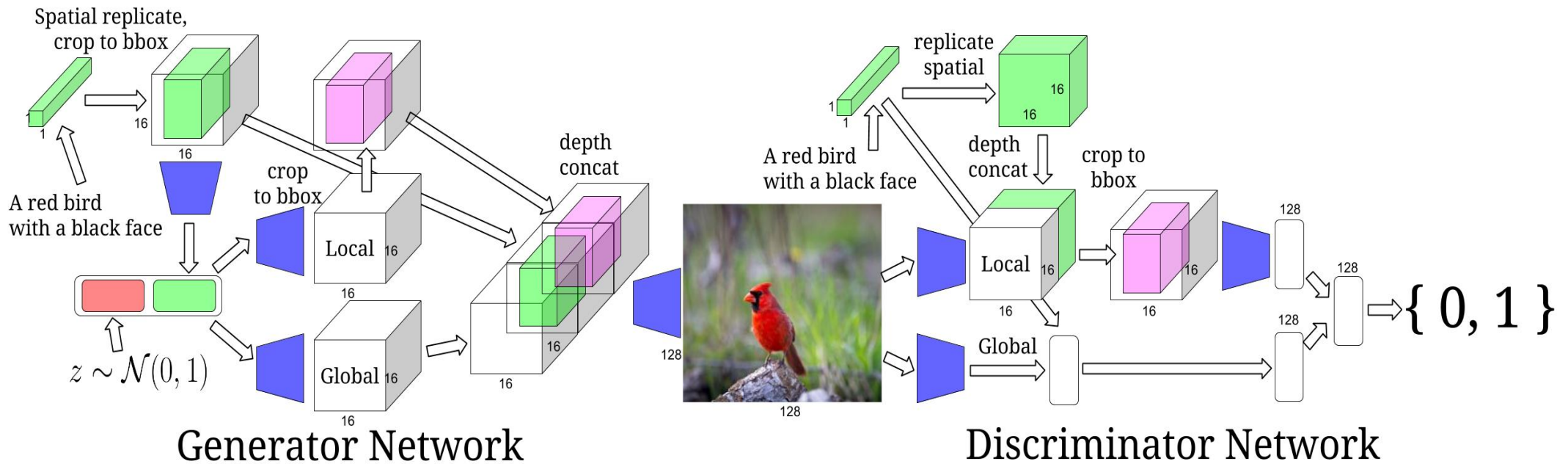
Conditioning on bounding box



Conditioning on bounding box



Conditioning on bounding box



Moving the bird around with bounding box (noize z fixed)

Caption

This bird has a black head, a long orange beak and yellow body

GT



Moving the bird around with bounding box (noize z fixed)

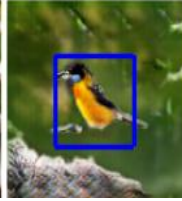
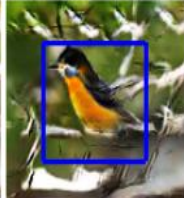
Caption

This bird has a black head, a long orange beak and yellow body

GT



Shrinking



Moving the bird around with bounding box (noize z fixed)

Caption

This bird has a black head, a long orange beak and yellow body

GT



Translation



Moving the bird around with bounding box (noize z fixed)

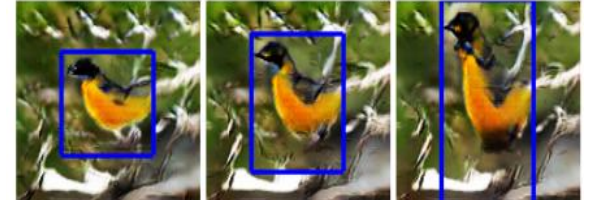
Caption

This bird has a black head, a long orange beak and yellow body

GT



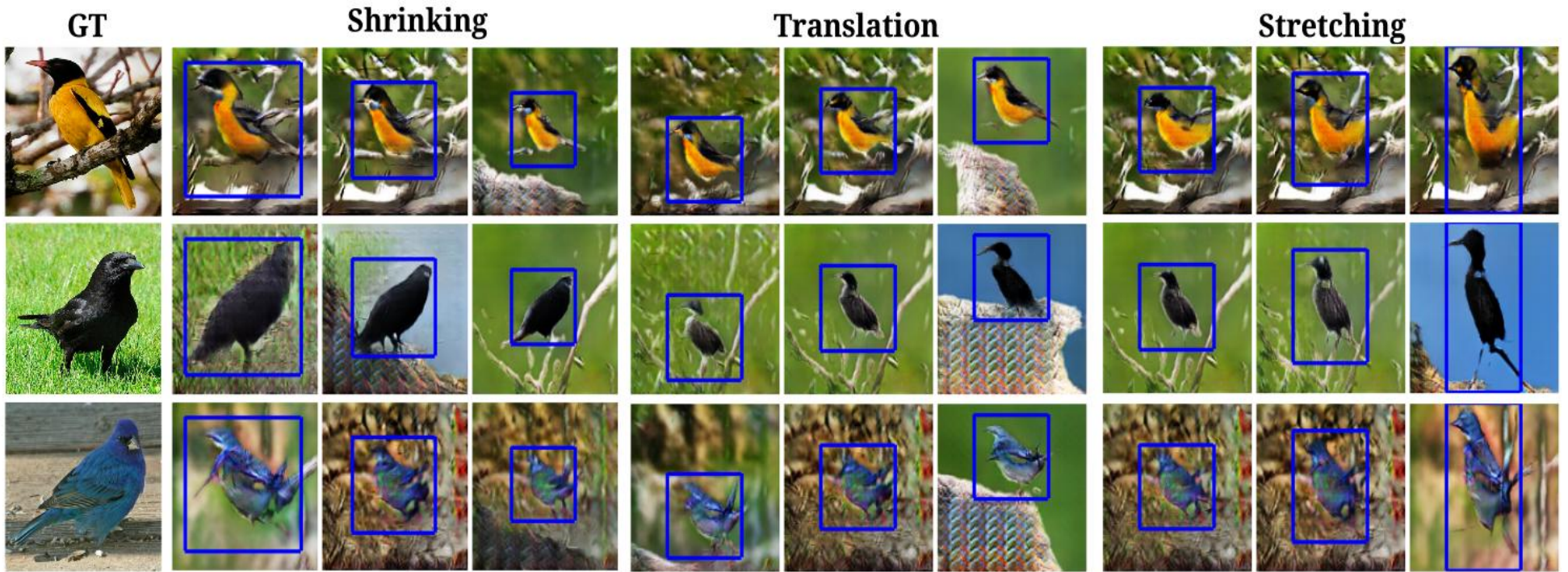
Stretching



Moving the bird around with bounding box (noize z fixed)

Caption

This bird has a black head, a long orange beak and yellow body



This large black bird has a pointy beak and black eyes

This small blue bird has a short pointy beak and brown patches on its wings

Moving the bird around with key points (noize z fixed)

Caption

This bird has a black head, a long orange beak and yellow body

GT



Shrinking



Moving the bird around with key points (noize z fixed)

Caption

This bird has a black head, a long orange beak and yellow body

GT



Translation



Moving the bird around with key points (noize z fixed)

Caption

This bird has a black head, a long orange beak and yellow body































GT



Stretching



Moving the bird around with key points (noize z fixed)

Caption	GT	Shrinking			Translation			Stretching		
This bird has a black head, a long orange beak and yellow body										
This large black bird has a pointy beak and black eyes										
This small blue bird has a short pointy beak and brown patches on its wings										

Comparison to text-only conditional GAN:

Ground-truth image and text caption



A small sized bird that has tones of brown and dark red with a short stout bill



This bird has a yellow breast and a dark grey face



The bird is solid black with white eyes and a black beak.

Comparison to text-only conditional GAN:

Ground-truth image and text caption



A small sized bird that has tones of brown and dark red with a short stout bill



This bird has a yellow breast and a dark grey face



The bird is solid black with white eyes and a black beak.

GAN-INT-CLS (Reed et. al, 2016b)



^ Text-only 64 x 64 GAN samples.

Comparison to text-only conditional GAN:

Ground-truth image and text caption



A small sized bird that has tones of brown and dark red with a short stout bill

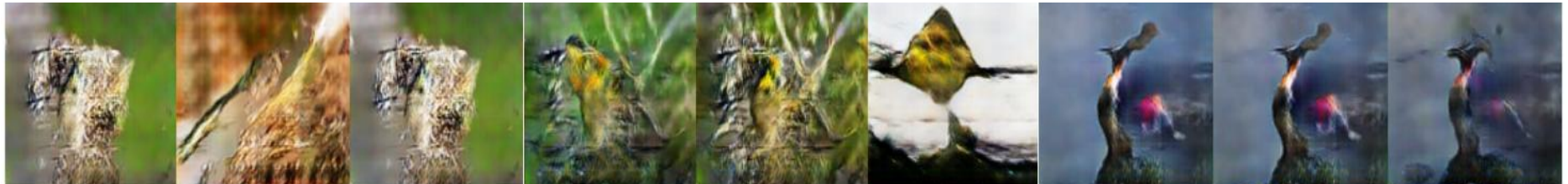


This bird has a yellow breast and a dark grey face



The bird is solid black with white eyes and a black beak.

GAWWN trained without key points



^ Text-only 128 x 128 GAN samples.

Comparison to text-only conditional GAN:

Ground-truth image and text caption



A small sized bird that has tones of brown and dark red with a short stout bill



This bird has a yellow breast and a dark grey face



The bird is solid black with white eyes and a black beak.

GAWWN
Key points given



^ Ours, with fixed keypoints.

Comparison to text-only conditional GAN:

Ground-truth image and text caption



A small sized bird that has tones of brown and dark red with a short stout bill



This bird has a yellow breast and a dark grey face



The bird is solid black with white eyes and a black beak.

GAWWN
Key points
generated



^ Ours, with generated keypoints.

Comparison to text-only conditional GAN:

Ground-truth image and text caption



A small sized bird that has tones of brown and dark red with a short stout bill



This bird has a yellow breast and a dark grey face

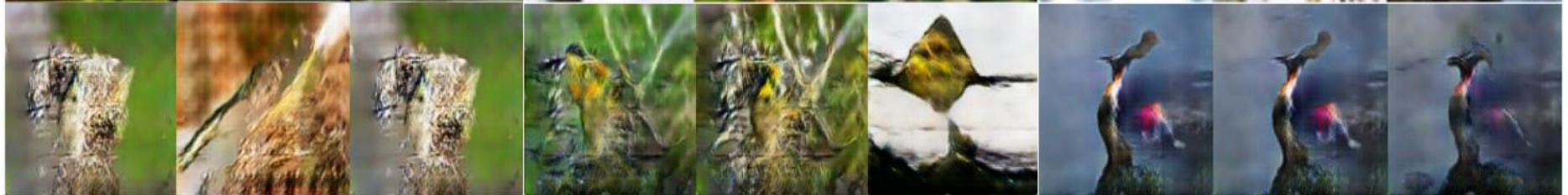


The bird is solid black with white eyes and a black beak.

GAN-INT-CLS (Reed et. al, 2016b)



GAWWN trained without key points



GAWWN Key points given



GAWWN Key points generated



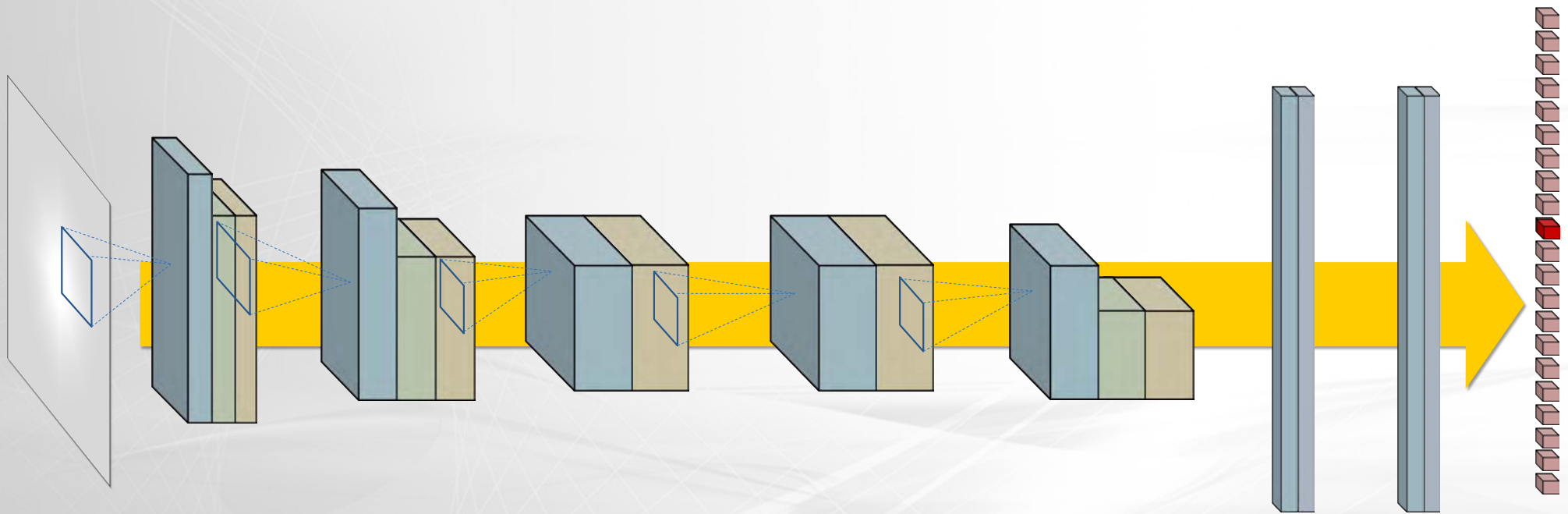
Deep Model Adaptation using Domain Adversarial Training

Victor Lempitsky,
joint work with Yaroslav Ganin



*Skolkovo Institute of Science and Technology (Skoltech)
Moscow region, Russia*

Deep supervised neural networks



- are a “big thing” in computer vision and beyond
- **are hungry for labeled data**



Where to get the data?

Lots of modalities do not have large labeled data sets:

- Biomedical
- Unusual cameras / image types
- Videos
- Data with expert-level annotation (not mTurkable)
-

Surrogate training data often available:

- Borrow from adjacent modality
- Generate synthetic imagery (computer graphics)
- Use data augmentation to amplify data (*image-based rendering, morphing, re-synthesis,....*)

Resulting training data are shifted. *Domain adaptation* needed.

Example: Internet images -> Webcam sensor



[Saenko et al. ECCV2010]

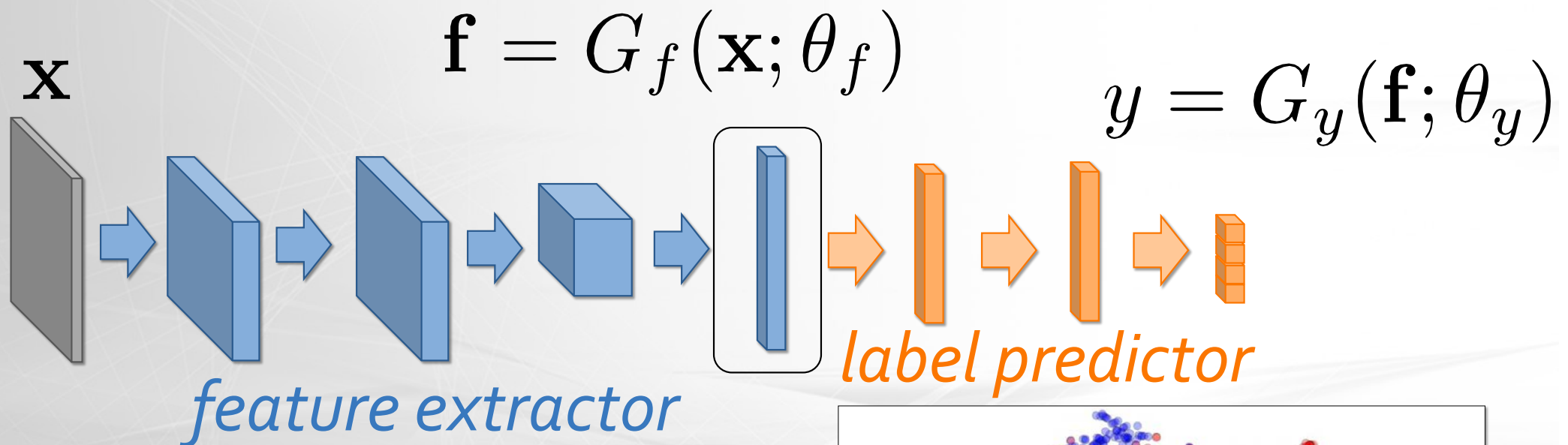


Assumptions and goals

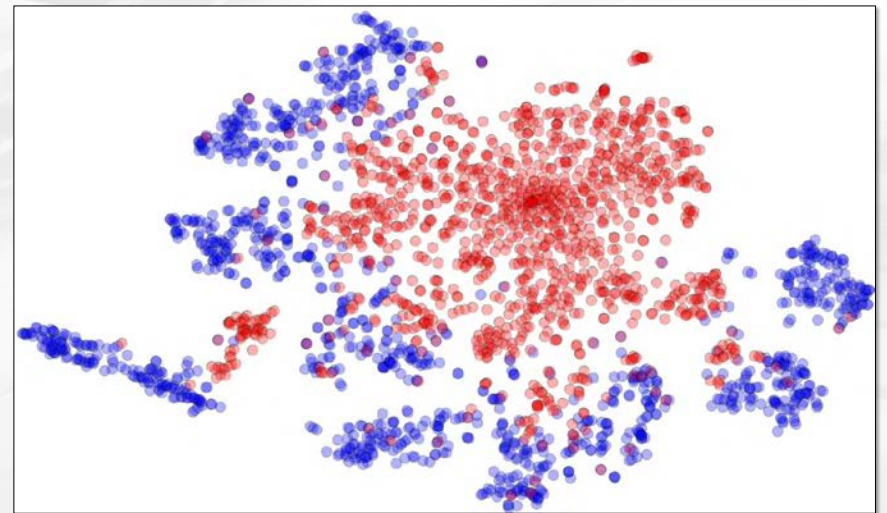
- Lots of *labeled* data in the *source domain* (e.g. synthetic images)
- Lots of *unlabeled* data in the *target domain* (e.g. real images)
- **Goal:** train a **deep neural net** that does well on the **target domain**

Large-scale deep unsupervised domain adaptation

Domain shift in a deep architecture



When trained on source only, feature distributions do not match:

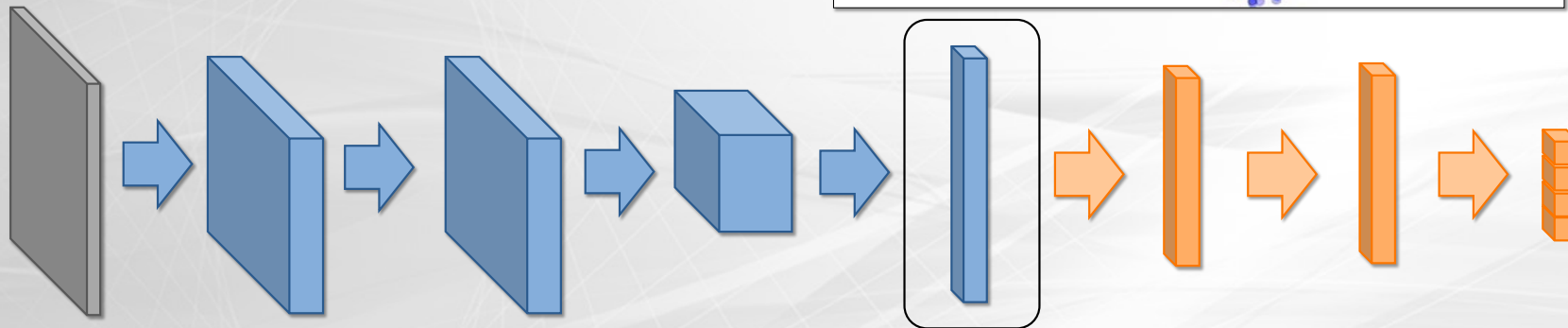
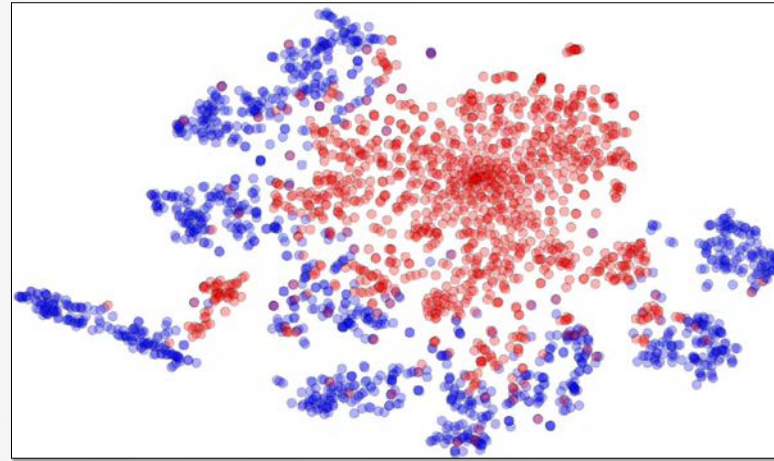


$$S(\mathbf{f}) = \{G_f(\mathbf{x}; \theta_f) \mid \mathbf{x} \sim S(\mathbf{x})\}$$

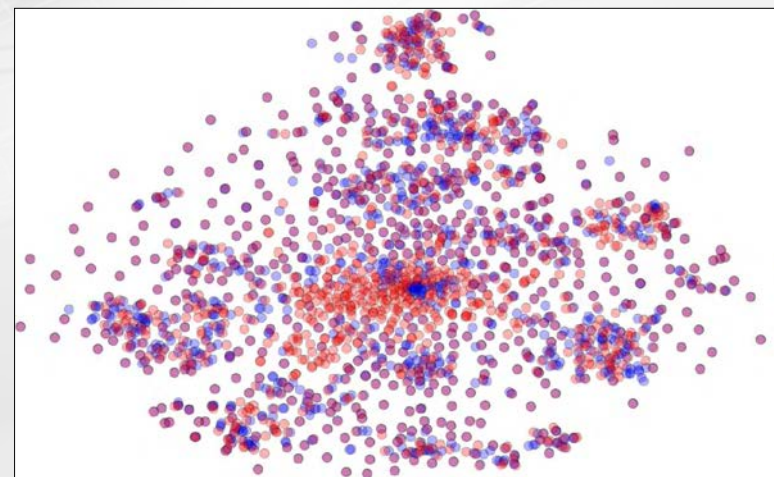
$$T(\mathbf{f}) = \{G_f(\mathbf{x}; \theta_f) \mid \mathbf{x} \sim T(\mathbf{x})\}$$

Idea 1: domain-invariant features wanted

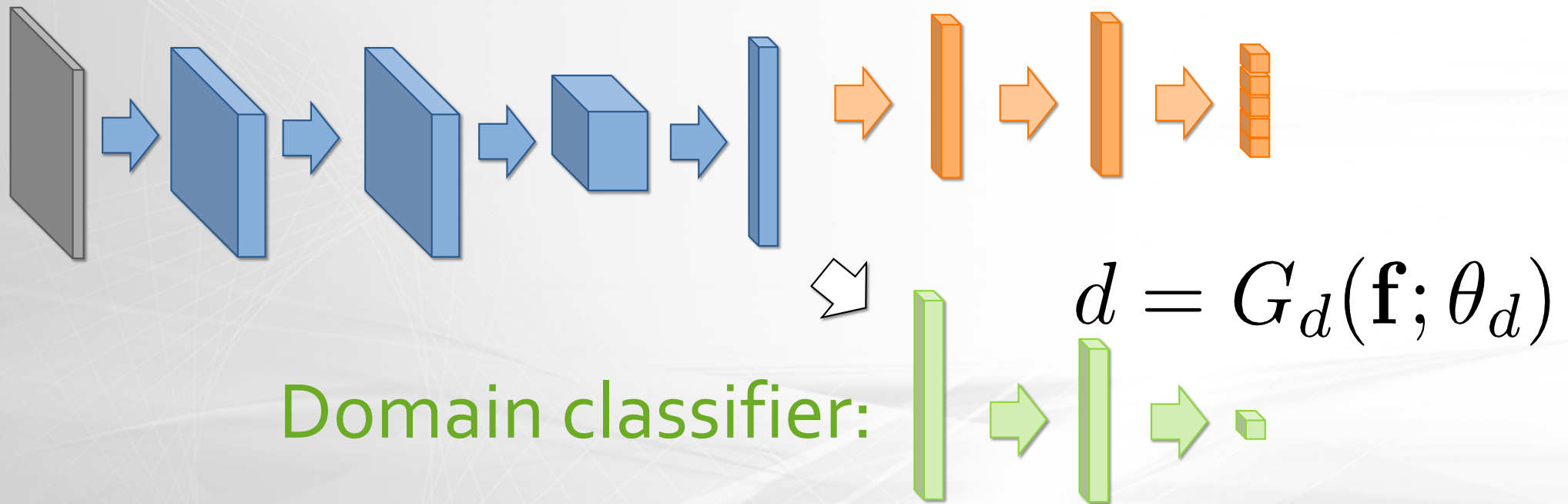
Feature distribution without adaptation:



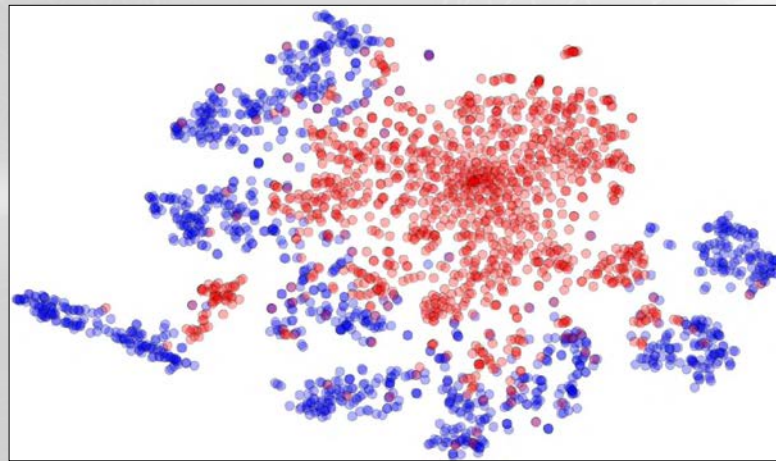
Our goal (after adaptation):



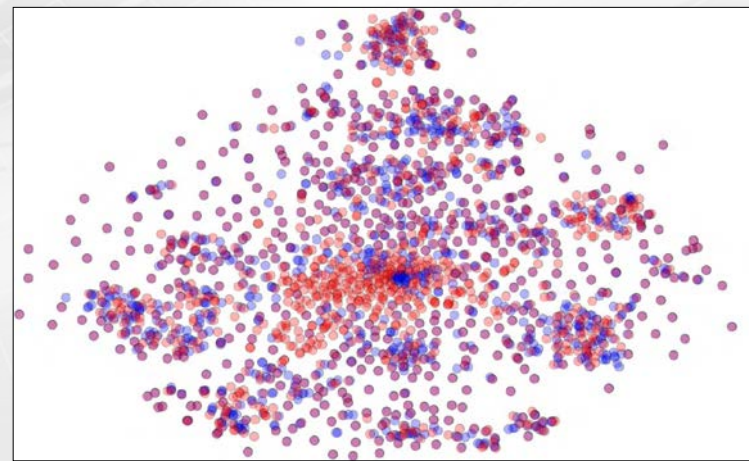
Idea 2: measuring domain shift



Domain classifier:

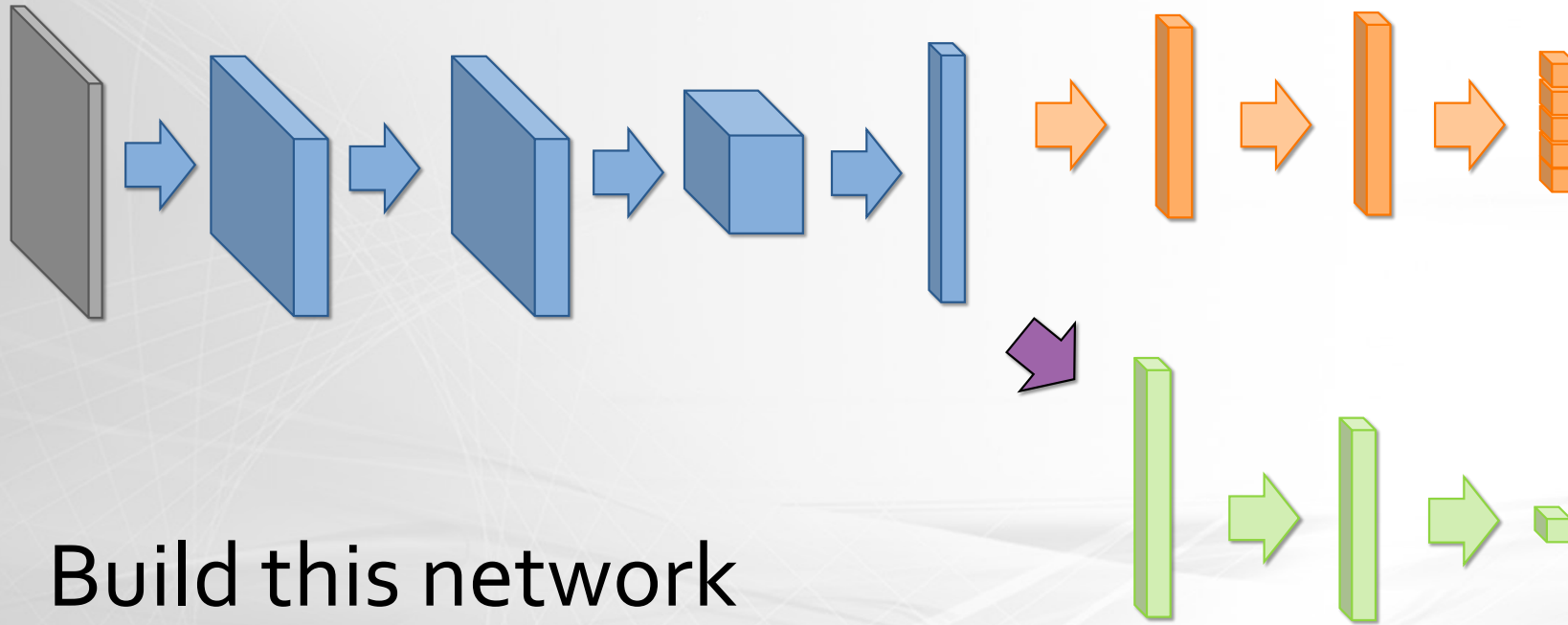


Domain loss low



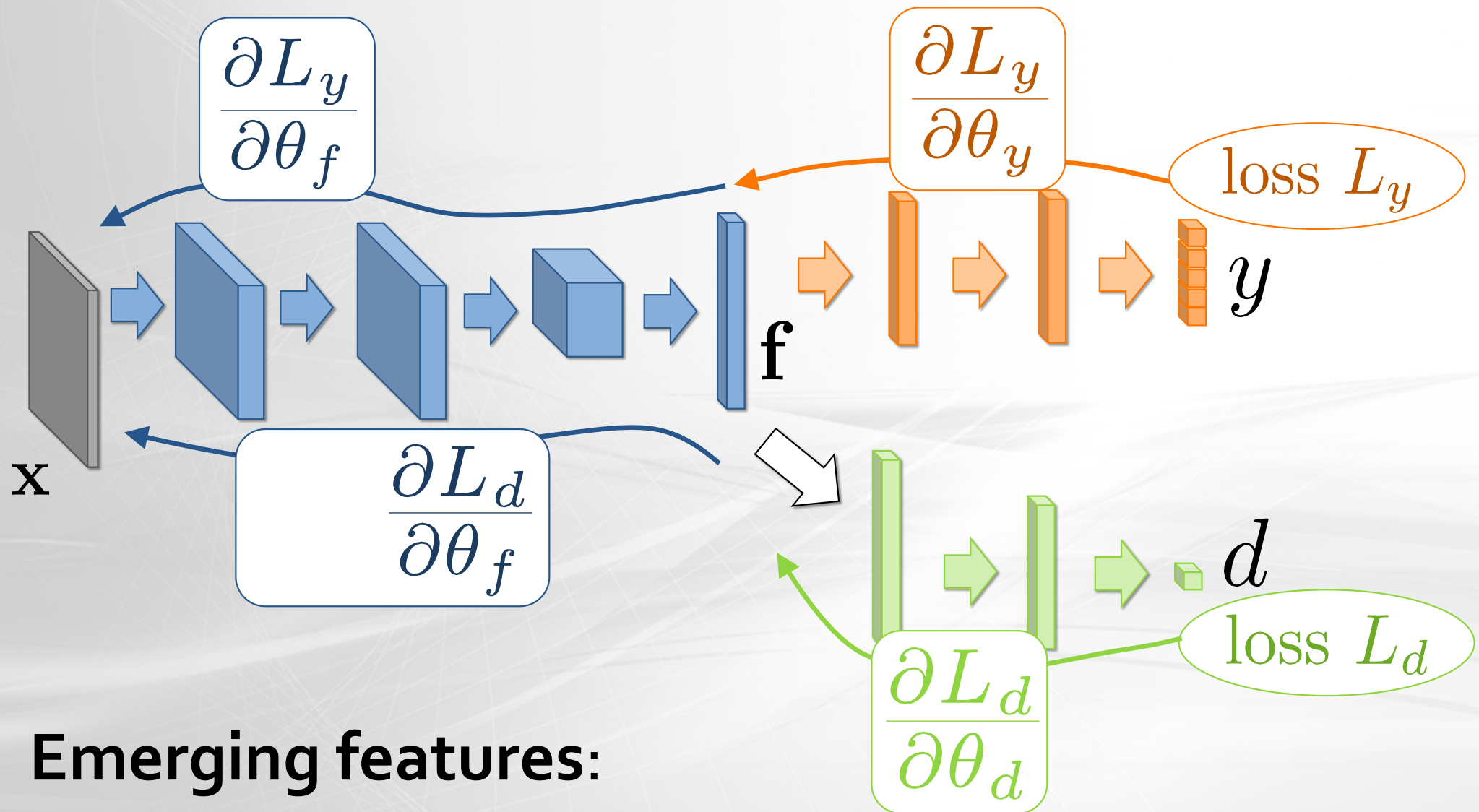
Domain loss high

Learning with adaptation



1. Build this network
2. Train **feature extractor** + **class predictor** on source data
3. Train **feature extractor** + **domain classifier** on source+target data
4. Use **feature extractor** + **class predictor** at test time

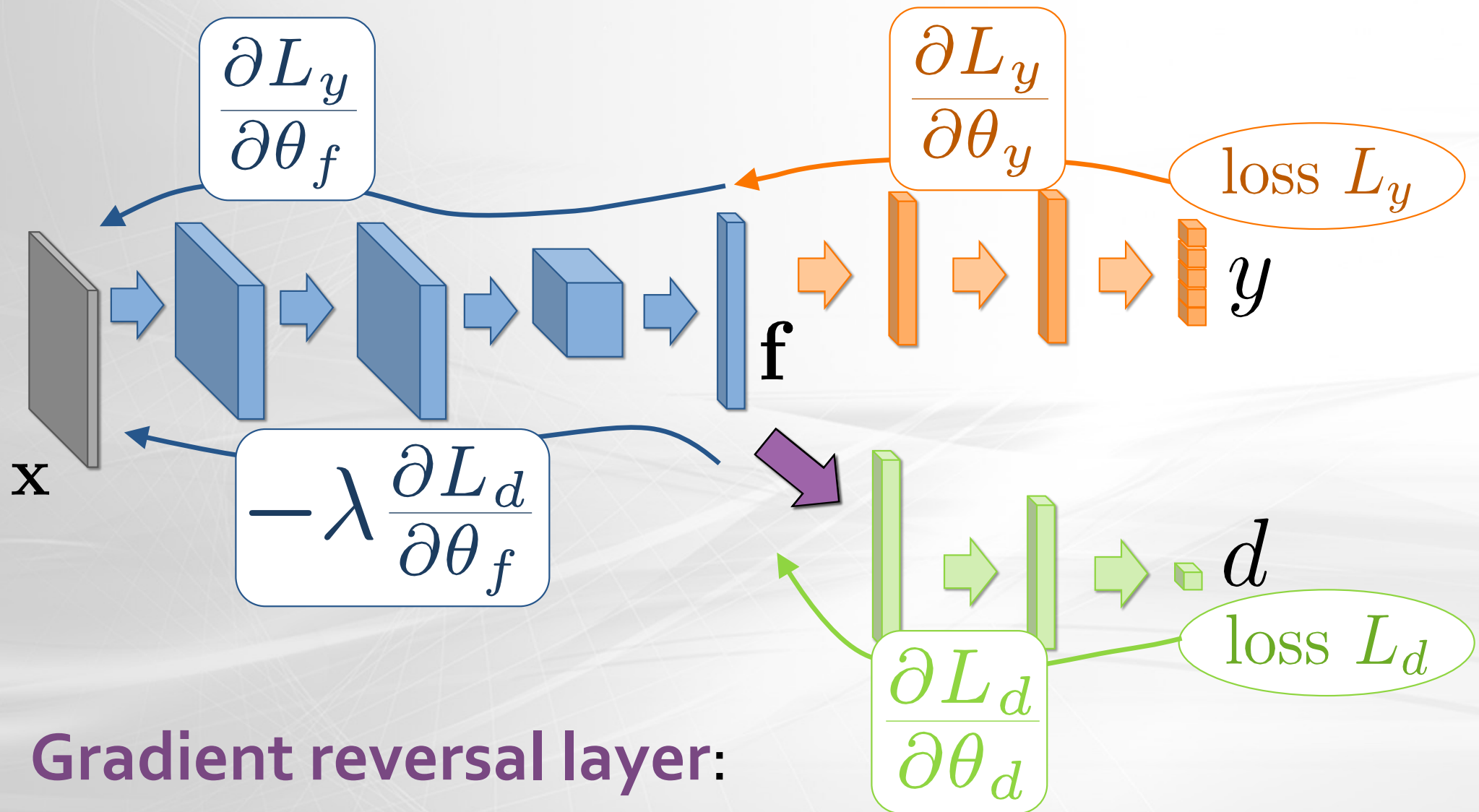
Idea 3: minimizing domain shift



Emerging features:

- Discriminative (good for predicting y)
- Domain-discriminative (good for predicting d)

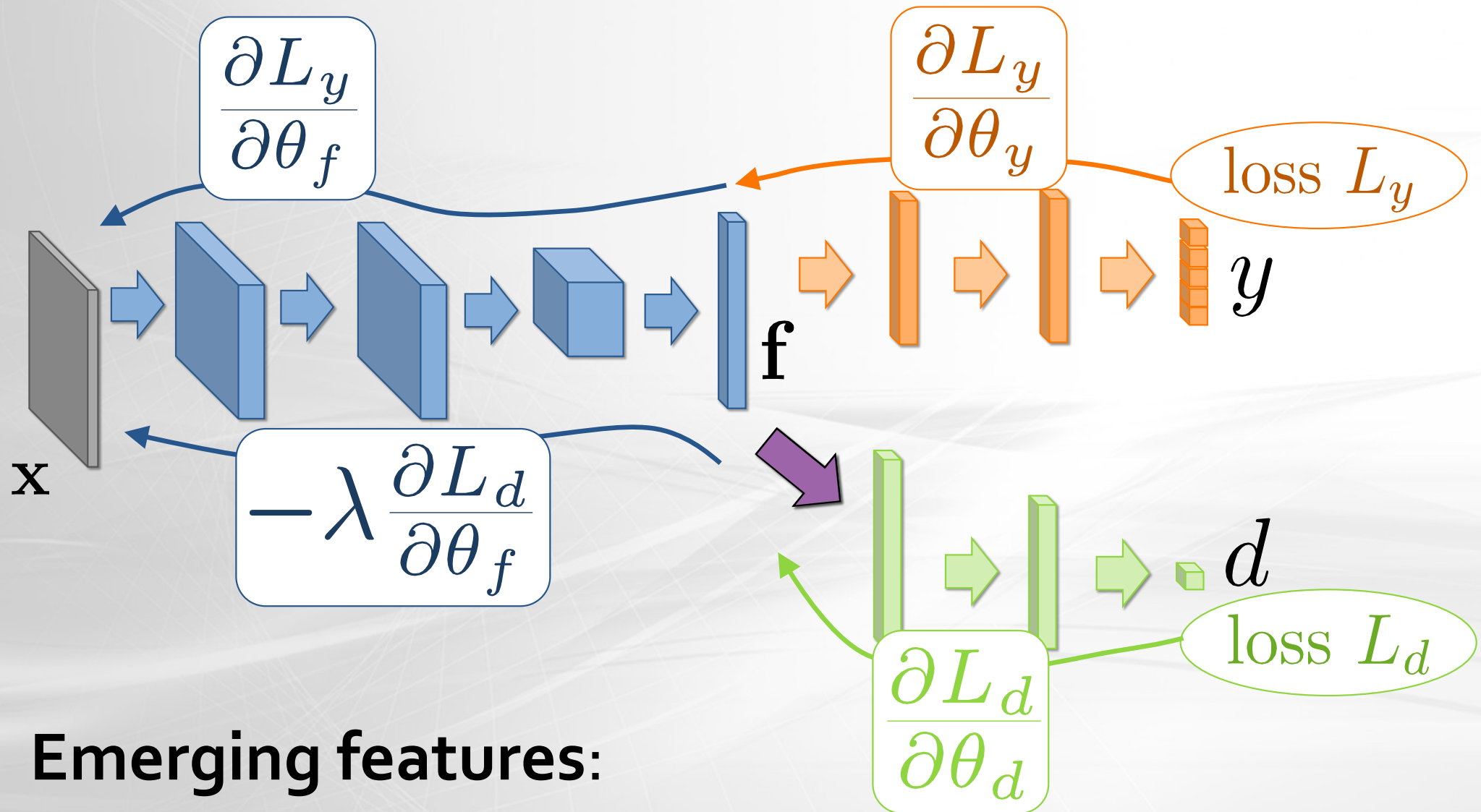
Idea 3: minimizing domain shift



Gradient reversal layer:

- Copies data without change at forwardprop
- Multiplies the gradient by $-\lambda$ at backprop

Idea 3: minimizing domain shift



Emerging features:

- Discriminative (good for predicting y)
- Domain-invariant (not good for predicting d)

Saddle point interpretation

Our objective (small label prediction loss + large domain classification loss wanted)

$$E(\theta_f, \theta_y, \theta_d) = \sum_{\substack{i=1..N \\ d_i=0}} L_y^i(\theta_f, \theta_y) - \lambda \sum_{i=1..N} L_d^i(\theta_f, \theta_d)$$

The backprop converges to a saddle point:

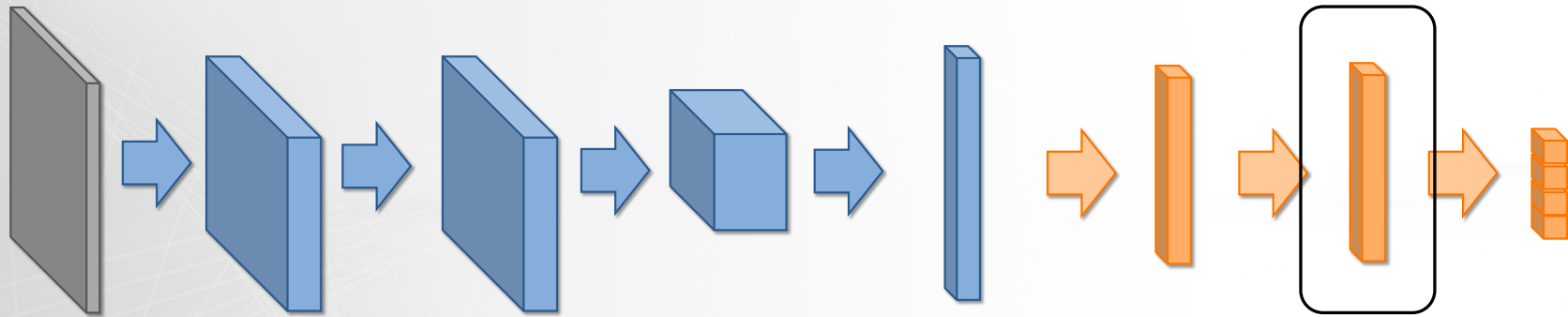
$$(\hat{\theta}_f, \hat{\theta}_y) = \arg \min_{\theta_f, \theta_y} E(\theta_f, \theta_y, \hat{\theta}_d)$$

$$\hat{\theta}_d = \arg \max_{\theta_d} E(\hat{\theta}_f, \hat{\theta}_y, \theta_d).$$

Similar idea for generative networks:

[Goodfellow et al. Generative adversarial nets. In NIPS, 2014]

Initial experiments: baselines

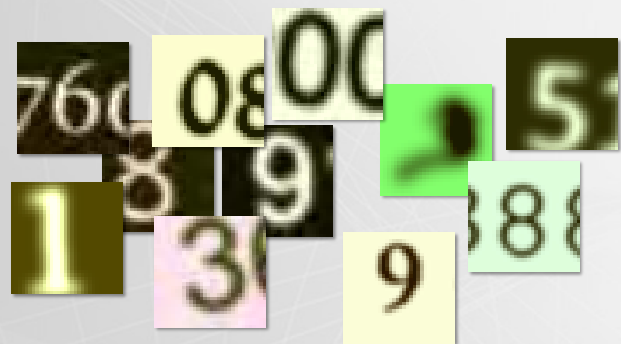


Upper bound: training on target domain **with labels**

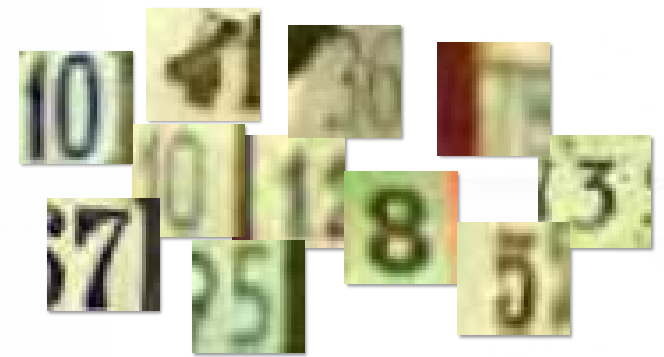
Shallow adaptation baseline: *[Fernando et al., Unsupervised visual domain adaptation using subspace alignment. ICCV, 2013]* applied to the last-but-one layer

Lower bound: training on source domain only

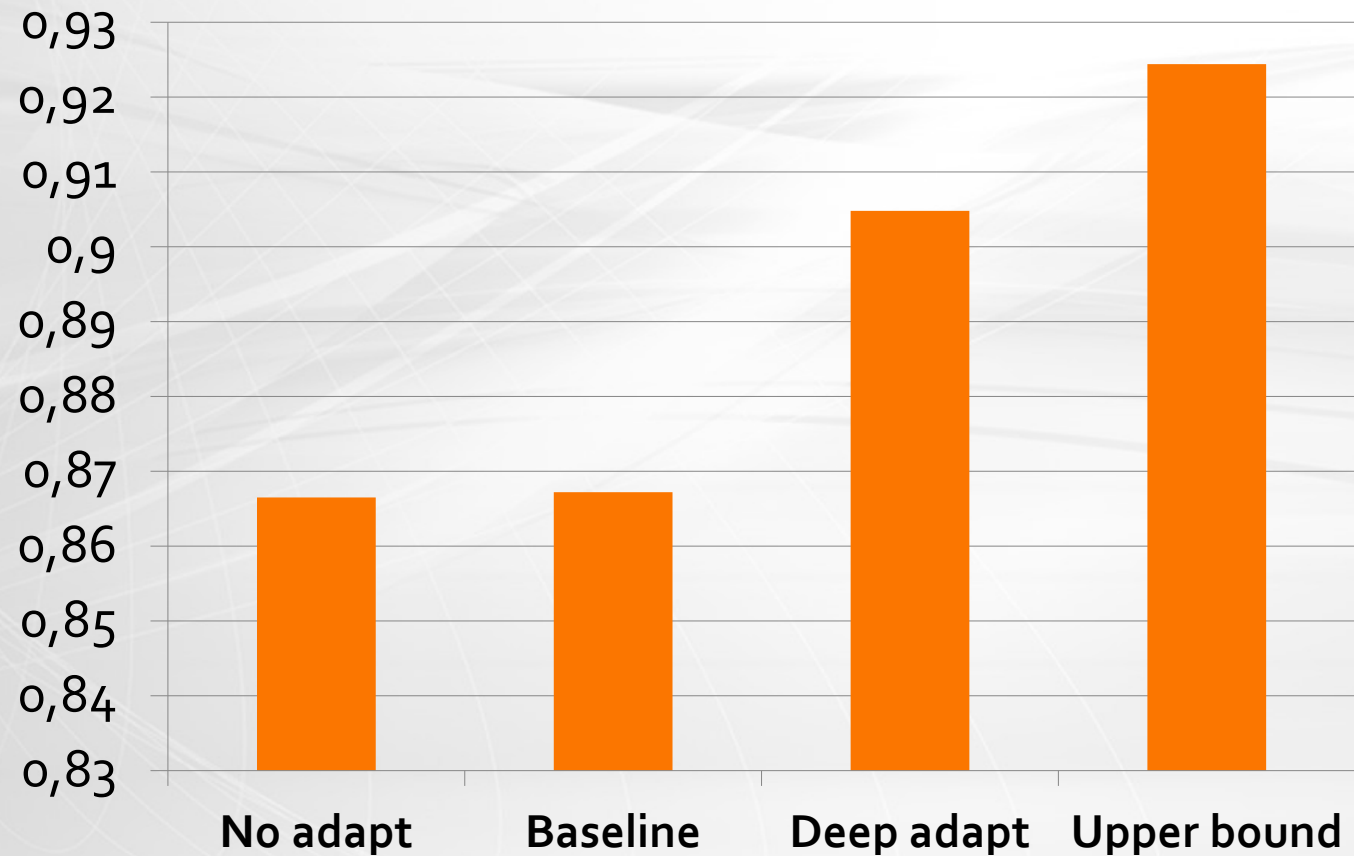
Example: from synthetic to real



"Windows digits"



"House numbers"



Office dataset



[Saenko et al. ECCV2010]



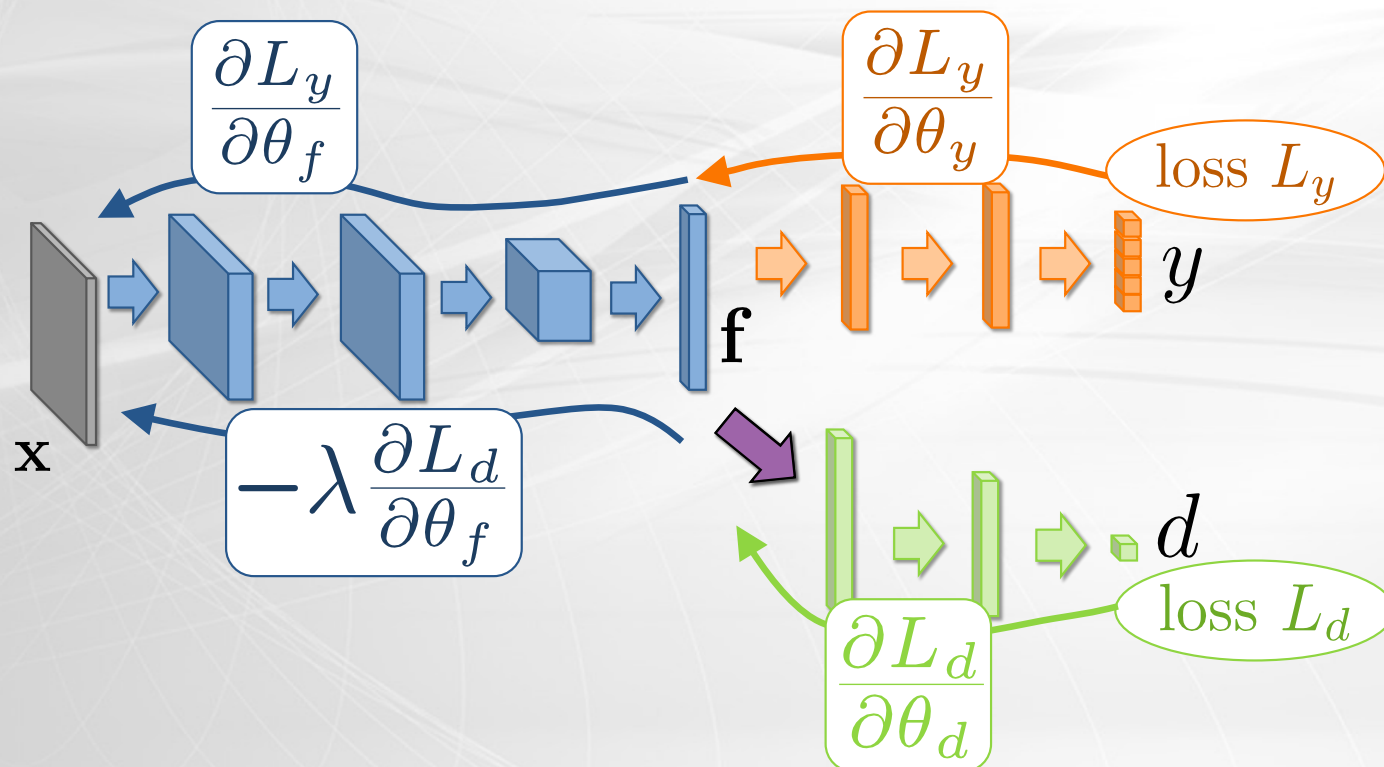
Results on Office dataset

METHOD	SOURCE	AMAZON	DSLIR	WEBCAM
	TARGET	WEBCAM	WEBCAM	DSLIR
GFK(PLS, PCA) (Gong et al., 2012)		.197	.497	.6631
SA* (Fernando et al., 2013)		.450	.648	.699
DLID (Chopra et al., 2013)		.519	.782	.899
DDC (Tzeng et al., 2014)		.618	.950	.985
DAN (Long and Wang, 2015)		.685	.960	.990
SOURCE ONLY		.642	.961	.978
DANN		.730	.964	.992

Most similar approach (matches means of distributions):
[Tzeng et al. Deep domain confusion: Maximizing for domain invariance. CoRR, abs/1412.3474, 2014]

Caveats

- Domains should not be too far apart
- Early on, the gradient from the domain classification loss should not be too strong
- The trick used to obtain the results: gradually increase λ from 0 to 1



Conclusion

- Scalable method for deep unsupervised domain adaptation
- Based on simple idea. Takes few lines of code (+ defining a specific network architecture).
Caffe implementation available.
- State-of-the-art results
- Unsupervised parameter tuning is easy (look at the domain classifier error)
- Main challenge: initialization and stepsize

<http://sites.skoltech.ru/compvision/projects/grl/>