Image Synthesis and Domain Adaptation in Computer Vision

Christoph Lampert

IST AUSTRIA
Institute of Science and Technology

SASHIMI workshop at MICCAI,
October 21, 2016

IST Austria (Institute of Science and Technology Austria)

http://www.ist.ac.at

- institute for **basic research**
- opened in 2009
- located in outskirts of Vienna

Research at IST Austria

- curiosity-driven
- foster interdisciplinarity
- currently 42 research groups:
  - Computer Science, Mathematics, Physics, Biology, Neuroscience

We're hiring! (on all levels)

- faculty (deadline Nov 3!), postdocs, PhD students (deadline Jan 8), interns

Overview

Computer Vision

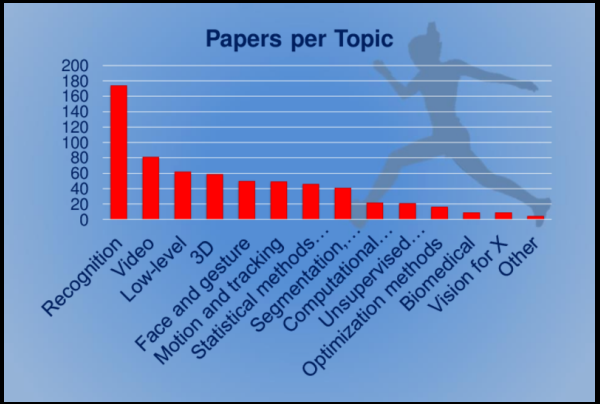
Synthetic Images for Classifier Training

Domain Adaptation

Generating Images with Neural Networks

Computer Vision

What is Computer Vision in 2016?



- a lot of **high-level vision**
  - object recognition
  - action classification
  - human pose estimation
- less **low-level vision**
  - image denoising
  - edge detection
- less **geometric vision**
  - 3D scene reconstruction
- almost no **biomedical imaging**

Images: CVPR 2016 Opening slides

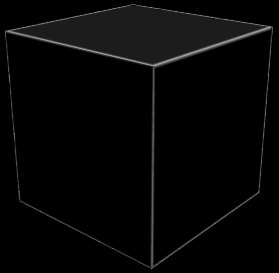
High-Level Computer Vision ≡ Applied Machine Learning

Training data

input data (images) with desired outputs, typically provided by a human expert

$\longrightarrow$

Learning algorithm



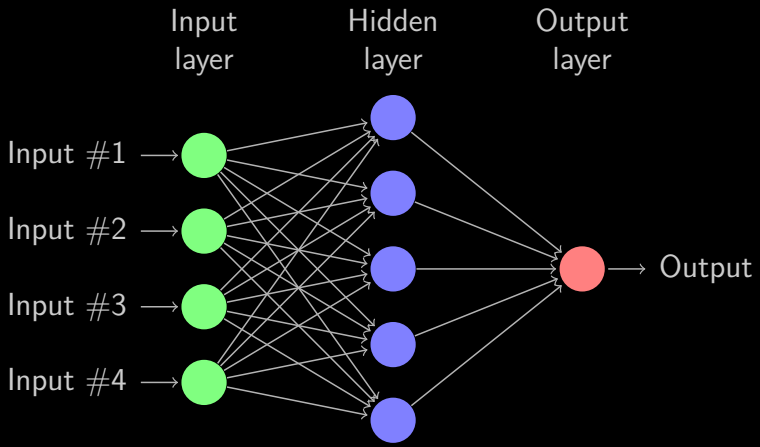
$\longrightarrow$

Trained system

software routine that can make predictions on new data

Image: http://pixabay.com

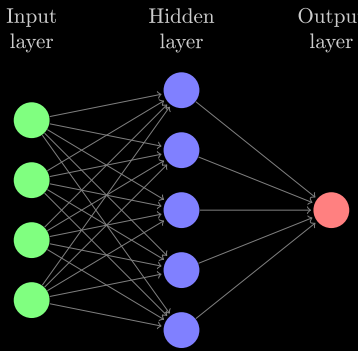
Computer Vision's (currently) Favorite Black Box



Neural Networks...

Image: http://www.texample.net/

Computer Vision's (currently) Favorite Black Box



State-of-the-art object classification network (VGG):

input resolution	224 x 224 x 3
number of outputs	1000
number of layers	19 ($\rightarrow$ "deep learning")
number of parameters	130.000.000
memory req.	100 MB per image
computational req.	19.6 GFLOPs per image

Neural Networks... but BIG and hungry for data

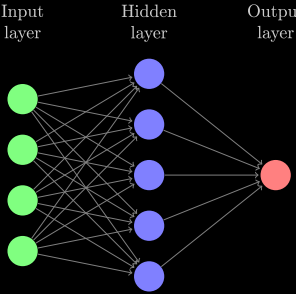
Image: http://www.texample.net/

Example: Object Recognition in Natural Images

ImageNet dataset

$\geq$ 14 million images
$\geq$ 20.000 categories
collected from the Internet
labels verified on MTurk

Deep Neural Network



Object Classifier

?

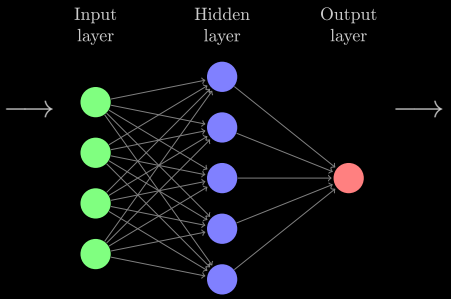
Images: ImageNet dataset, http://www.image-net.org

Example: Object Recognition in Natural Images

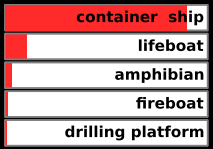
ImageNet dataset

$\geq$ 14 million images
$\geq$ 20.000 categories
collected from the Internet
labels verified on MTurk

Deep Neural Network

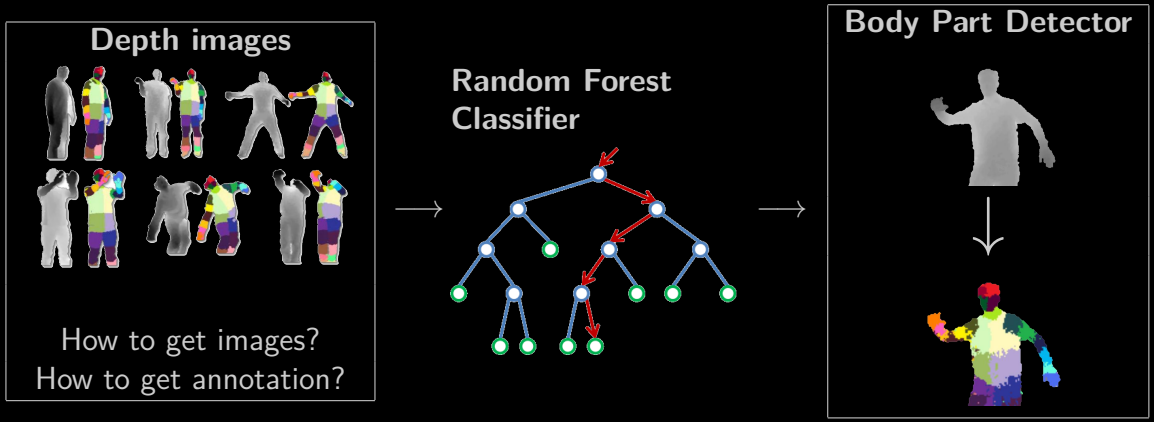


Object Classifier



Images: ImageNet dataset, http://www.image-net.org

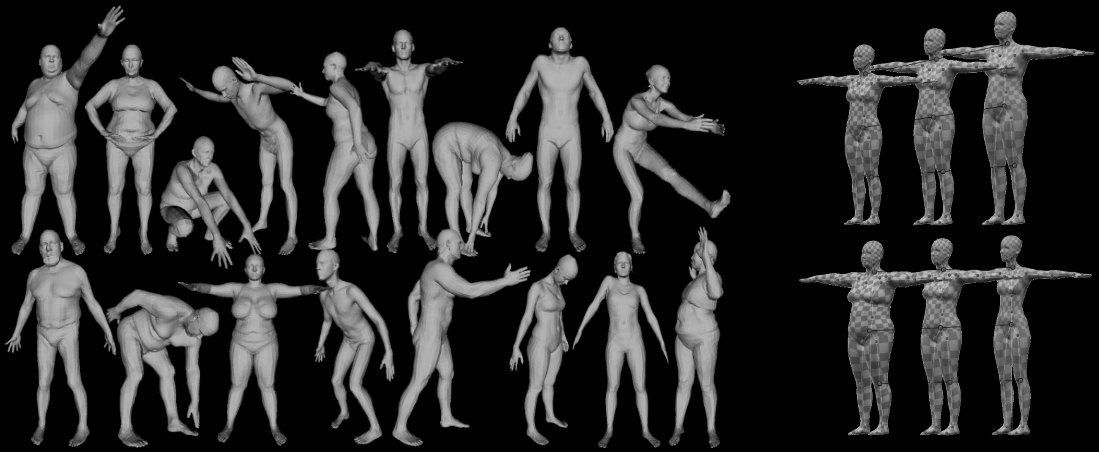
Example: Body Part Recognition for Microsoft Kinect



Images: [Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011]

Synthetic Images for Classifier Training

Synthetic Images for Classifier Training



- Computer Graphics has great models for generating body shapes and poses
- no need for manual annotation: by design we know which body part is where

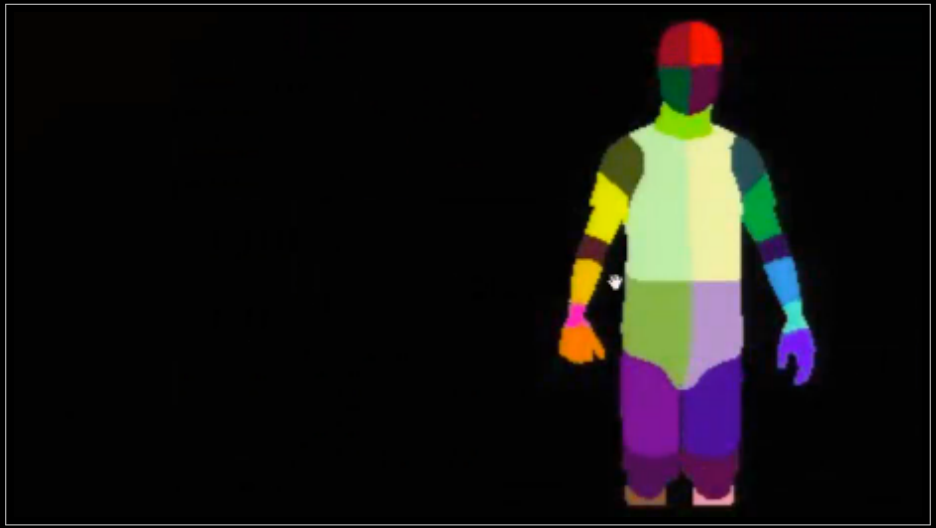
Images: [Loper *et al.*, "SMPL: A Skinned Multi-Person Linear Model", SIGGRAPH Asia, 2015]

Synthetic Images for Classifier Training

- We can create unlimited amounts of training data.

Images: [Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011]

Example: Synthetic Images for Human Pose Estimation

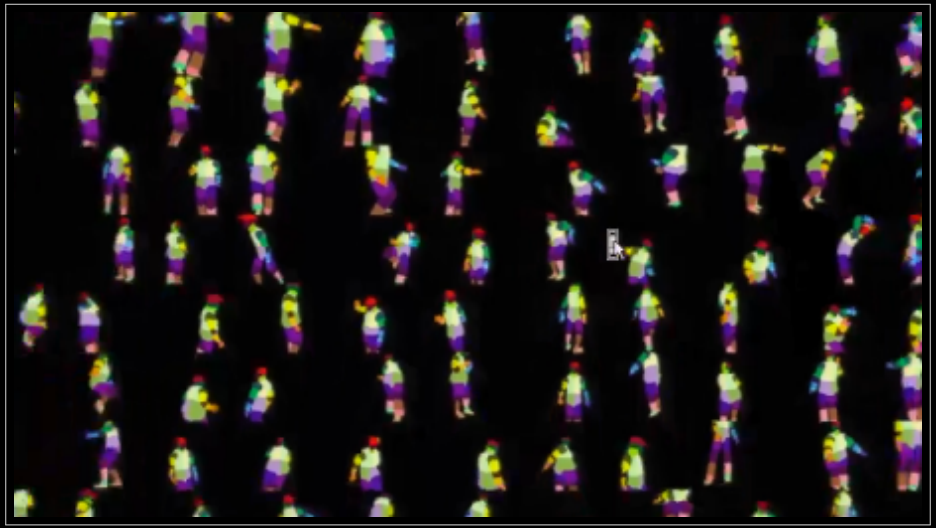


[Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011], http://www.youtube.com/watch?v=sejlrzcmhfU

Example: Synthetic Images for Human Pose Estimation

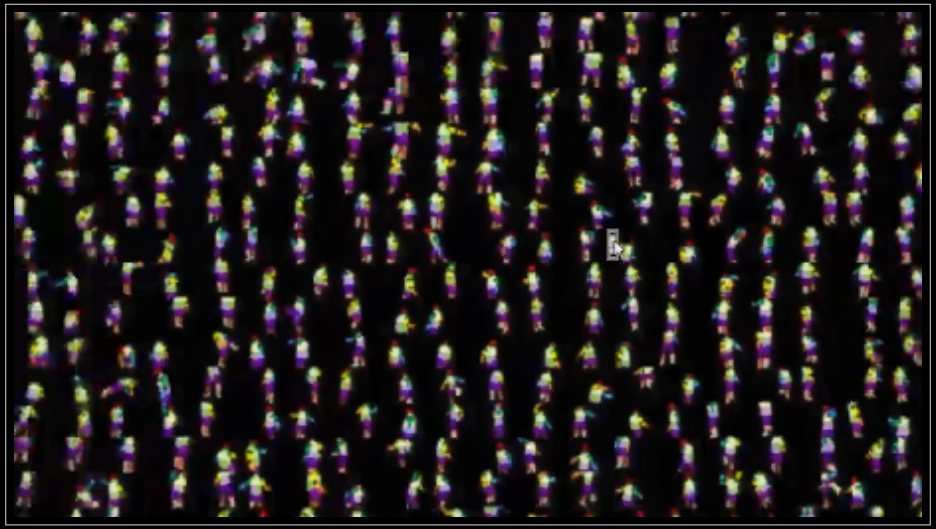
[Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011], http://www.youtube.com/watch?v=sejlrzcmhfU

Example: Synthetic Images for Human Pose Estimation



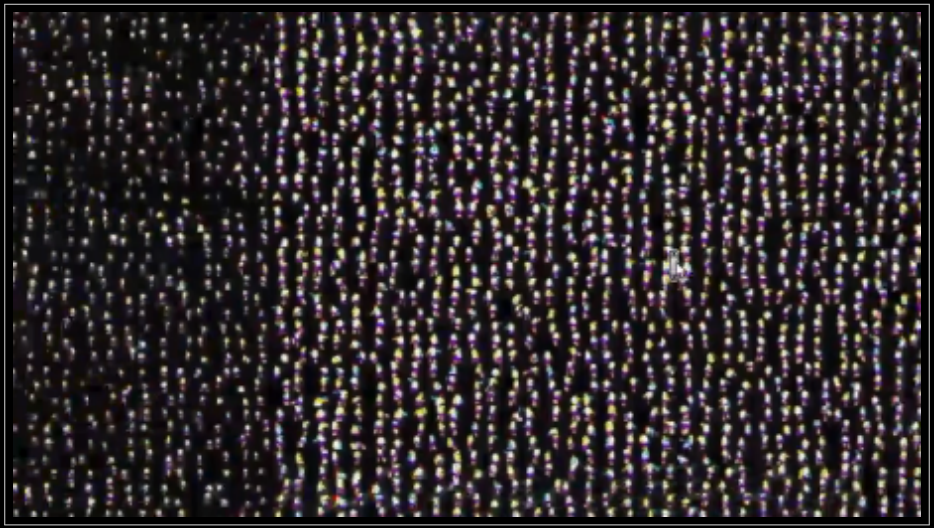
[Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011], http://www.youtube.com/watch?v=sejlrzcmhfU

Example: Synthetic Images for Human Pose Estimation



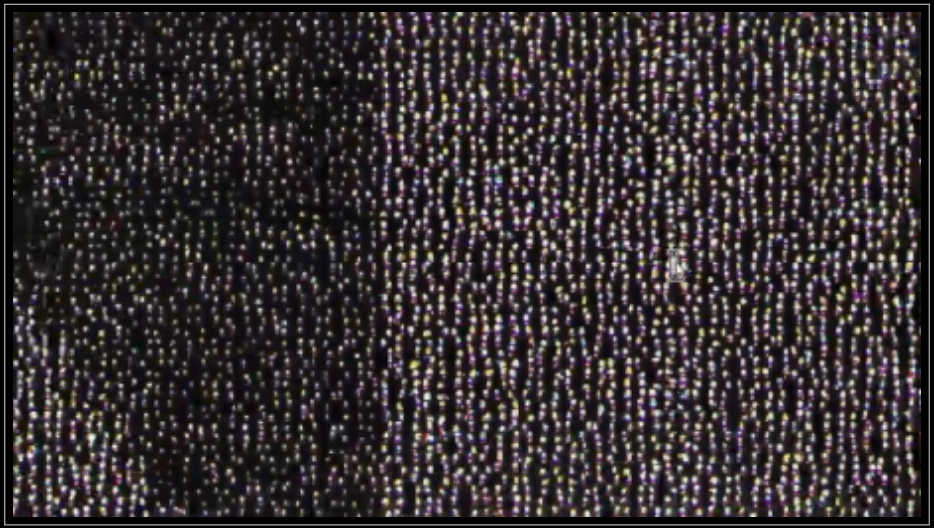
[Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011], http://www.youtube.com/watch?v=sejlrzcmhfU

Example: Synthetic Images for Human Pose Estimation



[Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011], http://www.youtube.com/watch?v=sejlrzcmhfU

Example: Synthetic Images for Human Pose Estimation



[Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011], http://www.youtube.com/watch?v=sejlrzcmhfU

Example: Synthetic Images for Human Pose Estimation

[Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011], http://www.youtube.com/watch?v=sejlrzcmhfU

Example: Synthetic Images for Human Pose Estimation

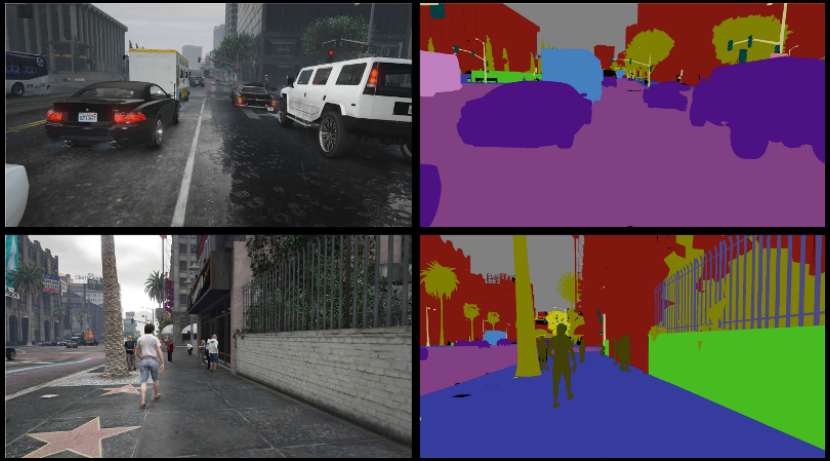
[Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images", CVPR 2011], http://www.youtube.com/watch?v=sejlrzcmhfU

Example: Synthetic Images for Human Pose Estimation

Commercial best-seller ✔

Image: https://www.engadget.com/

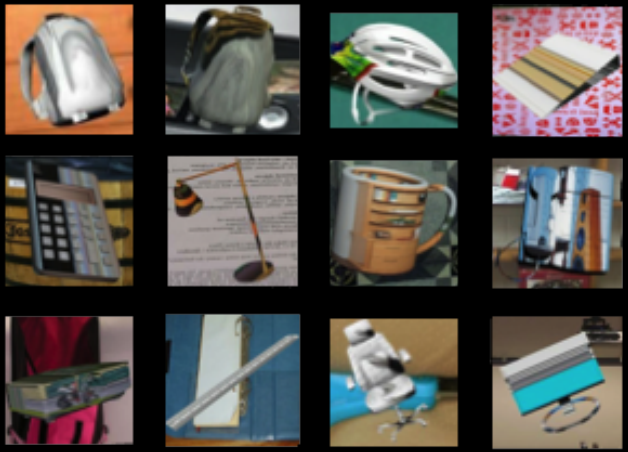
Synthetic Images for Semantic Segmentation of Street Scenes



Good results on challenging benchmarks ✔

Image: [Richter *et al.*, "Playing for Data: Ground Truth from Computer Games", ECCV 2016]

Synthetic Images for Object Recognition



Good results on artificial images, bad results on real images ✗

Images: [Sun, Saenko. "From Virtual to Reality: Fast Adaptation of Virtual Object Detectors to Real Domains", BMVC 2014]

Image Denoising with Synthetic Noise

Good results on images with artificial noise, bad results on images with real noise ✗

Image: "Barbara" by Allen Gersho

Summary: Synthetic Images for Classifier Training

Many computer visions methods require large amounts of annotated training data

Image synthesis/rendering can produce unlimited amounts of such training data.

But: training on synthetic instead of real images does not always yield good results.

Domain Adaptation

How does machine learning work?

- $x \in \mathcal{X}$: inputs, e.g. images
- $y \in \mathcal{X}$: outputs, e.g. class labels
- we looking for a **good prediction function**

$$f : \mathcal{X} \to \mathcal{Y}$$

How does machine learning work?

- $x \in \mathcal{X}$: inputs, e.g. images
- $y \in \mathcal{X}$: outputs, e.g. class labels
- we looking for a **good prediction function**

$$f : \mathcal{X} \to \mathcal{Y}$$

What's "good"?

- f should make few mistakes on real data, i.e.

$$\mathrm{er}_{\mathsf{real}}(f) := \frac{1}{n} \sum_{i=1}^{n} [\![y_i \neq f(x_i)]\!] \qquad \text{should be small}$$

where $S_{\mathsf{real}} = \{x_1, \ldots, x_n\}$ are real examples we want to classify, and y_i are the correct outputs for x_i (that are unknown to us)

How does machine learning work?

How to find a good f?

- given (synthetic) training set of input-output pairs $S_{\text{synth}} = \{ (x_1, y_1), \ldots, (x_n, y_n) \}$
- find f by minimizing

$$\min_f \quad \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) + \Omega(f)$$

- $\ell(y, f(x))$ is a smoothed version of $[\![y \neq f(x)]\!]$, e.g. log-loss or hinge loss
- $\Omega(f)$ is *regularizer* that prevents overfitting

This is going to work, if

- the training samples, S_{synth}, and the test samples, S_{real}, are "similar enough"

The problem of synthetic training data

If training data is very different from data at prediction time, predictions might be bad.

This is what happened in earlier examples:

- pose estimation: synthetic depth images very similar to real depth images
- semantic segmentation: game images very similar to real street scenes
- object classification: synthetic images looked "artificial", e.g. too clean
- denoising: synthetically generated image noise is not representative of real noise

Research questions:

- Can we quantify this effect?
- Can we prevent the problems?

Domain Adaptation

Very useful inequality:

Lemma [Ben-David *et al*. 2010]

$$\mathsf{er}_{\mathsf{real}}(f) \leq \mathsf{er}_{\mathsf{synth}}(f) + \mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}}) + \lambda$$

- $\mathsf{er}_{\mathsf{real}}(f)$ error on the test set (what we really care about)
- $\mathsf{er}_{\mathsf{synth}}(f)$ error on the training set (that we use for training)
- $\mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}})$ "discrepancy" between train and test data
- $\lambda = \mathbf{min}_h(\mathsf{er}_{\mathsf{synth}}(h), \mathsf{er}_{\mathsf{real}}(h))$

Lemma [Ben-David *et al*. 2010]

Discrepancy measures how well two sets of examples can be distinguished.

$$\mathrm{disc}(S_1, S_2) = 2(1 - 2e)$$

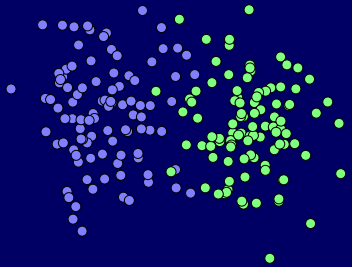
where e is the training error of the best classifier for the binary classification problem with S_1 as class 1 and S_2 as class 2.

Lemma [Ben-David *et al*. 2010]

Discrepancy measures how well two sets of examples can be distinguished.

$$\mathrm{disc}(S_1, S_2) = 2(1 - 2e)$$

where e is the training error of the best classifier for the binary classification problem with S_1 as class 1 and S_2 as class 2.

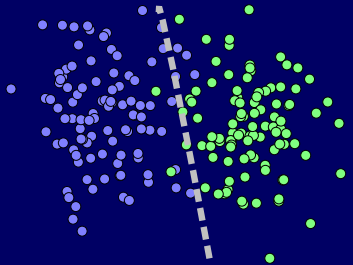


Lemma [Ben-David *et al*. 2010]

Discrepancy measures how well two sets of examples can be distinguished.

$$\mathrm{disc}(S_1, S_2) = 2(1 - 2e)$$

where e is the training error of the best classifier for the binary classification problem with S_1 as class 1 and S_2 as class 2.



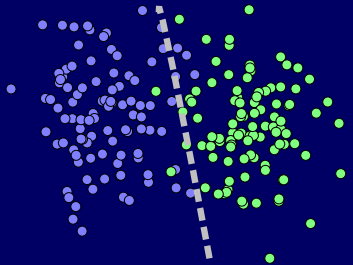
easy to separate
large discrepancy

Lemma [Ben-David *et al*. 2010]

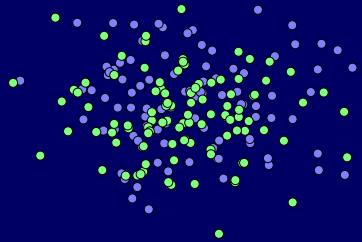
Discrepancy measures how well two sets of examples can be distinguished.

$$\mathrm{disc}(S_1, S_2) = 2(1 - 2e)$$

where e is the training error of the best classifier for the binary classification problem with S_1 as class 1 and S_2 as class 2.



easy to separate
large discrepancy

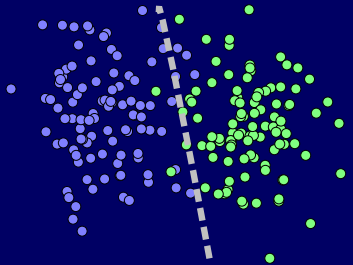


Lemma [Ben-David *et al*. 2010]

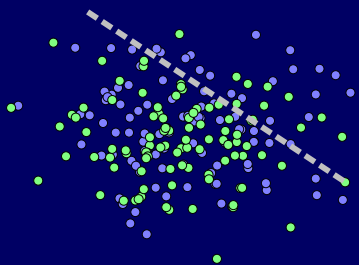
Discrepancy measures how well two sets of examples can be distinguished.

$$\mathrm{disc}(S_1, S_2) = 2(1 - 2e)$$

where e is the training error of the best classifier for the binary classification problem with S_1 as class 1 and S_2 as class 2.



easy to separate
large discrepancy



hard to separate
small discrepancy

Domain Adaptation

Lemma [Ben-David *et al*. 2010]

$$\mathsf{er}_{\mathsf{real}}(f) \leq \mathsf{er}_{\mathsf{synth}}(f) + \mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}}) + \lambda \qquad (*)$$

- $\mathsf{er}_{\mathsf{real}}(f)$ error on the test set (what we really care about)
- $\mathsf{er}_{\mathsf{synth}}(f)$ error on the training set (that we use for training)
- $\mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}})$ "discrepancy" between train and test data
- $\lambda = \mathbf{min}_h(\mathsf{er}_{\mathsf{synth}}(h), \mathsf{er}_{\mathsf{real}}(h))$

If we find f such that the right hand side of $(*)$ is small, then f will work on test data.

Domain Adaptation

Lemma [Ben-David *et al*. 2010]

$$\mathsf{er}_{\mathsf{real}}(f) \leq \mathsf{er}_{\mathsf{synth}}(f) + \mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}}) + \lambda \qquad (*)$$

- $\mathsf{er}_{\mathsf{real}}(f)$ error on the test set (what we really care about)
- $\mathsf{er}_{\mathsf{synth}}(f)$ error on the training set (that we use for training)
- $\mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}})$ "discrepancy" between train and test data
- $\lambda = \mathbf{min}_h(\mathsf{er}_{\mathsf{synth}}(h), \mathsf{er}_{\mathsf{real}}(h))$

If we find f such that the right hand side of $(*)$ is small, then f will work on test data.

- ignore λ, as it is independent of the classifier f

Domain Adaptation

Lemma [Ben-David *et al*. 2010]

$$\mathsf{er}_{\mathsf{real}}(f) \leq \mathsf{er}_{\mathsf{synth}}(f) + \mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}}) + \lambda \qquad (*)$$

- $\mathsf{er}_{\mathsf{real}}(f)$ error on the test set (what we really care about)
- $\mathsf{er}_{\mathsf{synth}}(f)$ error on the training set (that we use for training)
- $\mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}})$ "discrepancy" between train and test data
- $\lambda = \mathbf{min}_h(\mathsf{er}_{\mathsf{synth}}(h), \mathsf{er}_{\mathsf{real}}(h))$

If we find f such that the right hand side of $(*)$ is small, then f will work on test data.

- ignore λ, as it is independent of the classifier f
- minimize $\mathsf{er}_{\mathsf{synth}}(f)$, i.e. learn a good classifier on synthetic training set

Domain Adaptation

Lemma [Ben-David *et al*. 2010]

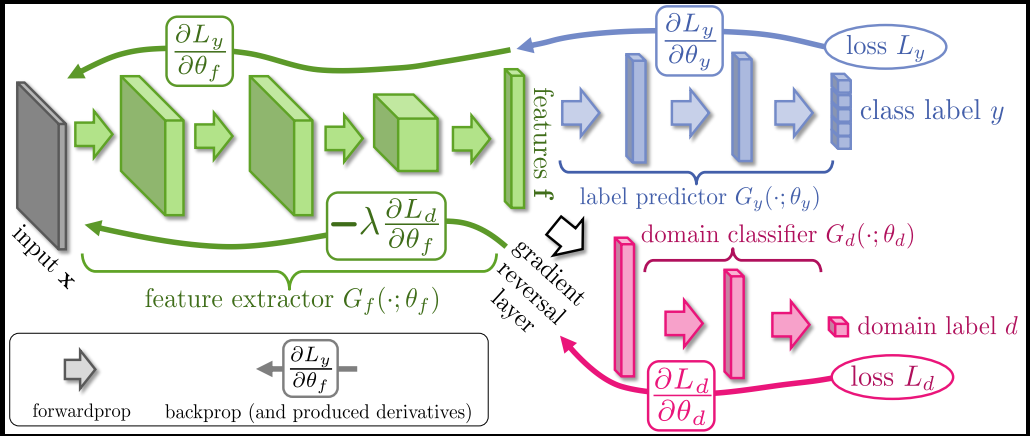
$$\mathsf{er}_{\mathsf{real}}(f) \leq \mathsf{er}_{\mathsf{synth}}(f) + \mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}}) + \lambda \qquad (*)$$

- $\mathsf{er}_{\mathsf{real}}(f)$ error on the test set (what we really care about)
- $\mathsf{er}_{\mathsf{synth}}(f)$ error on the training set (that we use for training)
- $\mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}})$ "discrepancy" between train and test data
- $\lambda = \mathbf{min}_h(\mathsf{er}_{\mathsf{synth}}(h), \mathsf{er}_{\mathsf{real}}(h))$

If we find f such that the right hand side of $(*)$ is small, then f will work on test data.

- ignore λ, as it is independent of the classifier f
- minimize $\mathsf{er}_{\mathsf{synth}}(f)$, i.e. learn a good classifier on synthetic training set
- make sure that $\mathrm{disc}(S_{\mathsf{real}}, S_{\mathsf{synth}})$ is small (a property of the data representation)

We can train for both properties at the same time: [Ganan, Lempitsky 2015]



- blue classifier, f, predicts labels, trained by minimizing training error L_y
- red classifier, predicts if samples are `synthetic` or `real` by minimizing loss L_d
- green representation, trained to **minimize** L_y and **maximize** L_d

Image: [Ganin, Lempitsky. "Unsupervised Domain Adaptation by Backpropagation", ICML 2015]

Summary: Domain Adaptation

Domain Adaptation is a branch of machine learning that studies what happens when the characteristics of the training data is not identical to the characteristics of the data at prediction time.

Typical example:

- training time: synthetic images
- prediction time: real images

Theory and algorithms exist to

- quantify, and
- overcome

the differences.

Generating Natural Images

So far, we can generate images based on well-designed models:

- physical simulations
- geometric models
- computer games

A lot of work to produce and not available for all tasks.

So far, we can generate images based on well-designed models:

- physical simulations
- geometric models
- computer games

A lot of work to produce and not available for all tasks.

Can we teach a computer to generate natural looking images?
If yes, which ones?

So far, we can generate images based on well-designed models:

- physical simulations
- geometric models
- computer games

A lot of work to produce and not available for all tasks.

Can we teach a computer to generate natural looking images?
If yes, which ones?

Idea: learn a function that samples random images from the manifold of all images

Excursion: sampling from a difficult distribution

- How to sample from a difficult probability distribution?
- Sample from an easy distribution and apply a transformation to the sample.

Excursion: sampling from a difficult distribution

- How to sample from a difficult probability distribution?
- Sample from an easy distribution and apply a transformation to the sample.

Examples:

- Box-Muller transform: uniform distribution to Gaussian distribution

$$u_1, u_2 \sim \text{uniform}([0,1]) \quad \rightarrow \quad v = \sqrt{-2 \log u_1} \cos(2\pi u_2) \sim \mathcal{N}(0,1)$$

- standard Gaussian to arbitrary Gaussian

$$u \sim \mathcal{N}(0,1) \qquad \rightarrow \qquad v = \sigma u + \mu \ \sim \mathcal{N}(\mu, \sigma^2)$$

- general 1D distribution P with cumulative distribution function Φ:

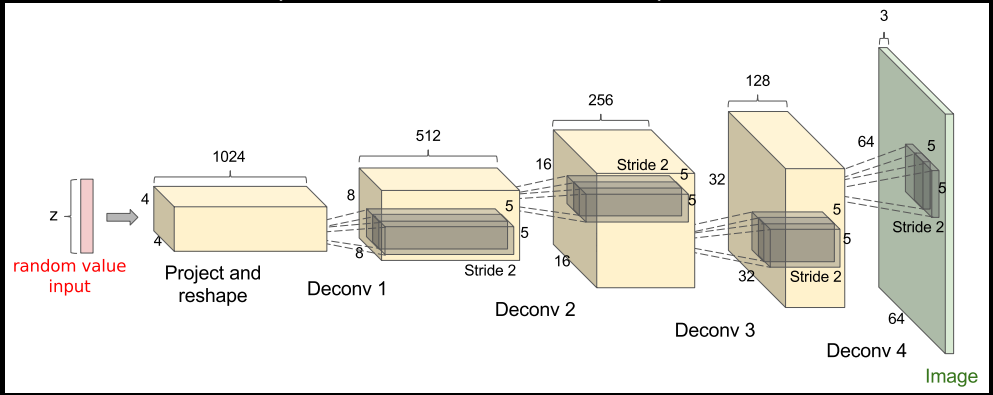
$$u \sim \text{uniform}([0,1]) \qquad \rightarrow \qquad v = \Phi^{-1}(u) \sim P$$

Image Generation with a Neural Network

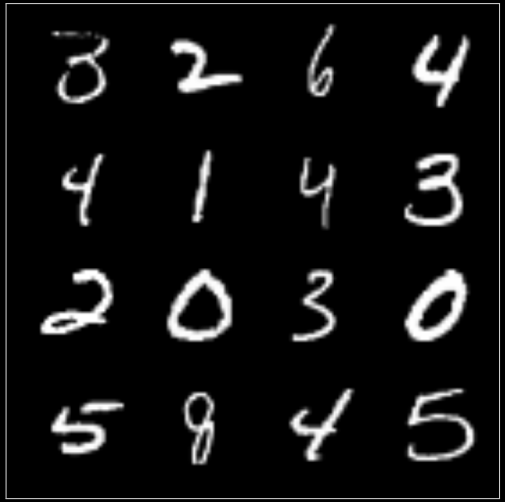
Idea: *learn* a neural network that samples randomly from the manifold of all images

$$f : \mathcal{Z} \to \mathcal{I} \qquad \text{for } \mathcal{Z} \subseteq \mathbb{R}^d, \quad \mathcal{I} = \{ \text{ set of images } \}$$

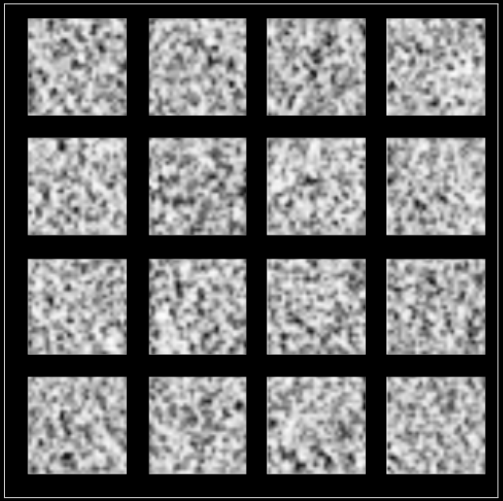
Feed-forward architecture (convolutional neural network):



Examples: generating digits



examples digits



network output

Images: https://oshearesearch.com/index.php/2016/07/01/mnist-generative-adversarial-model-in-keras/

Image Generation with a Neural Network

Problem: we have to *train* the network, f, to do what we want

- find network parameters that minimize a loss function:

$$\min_f \mathcal{L}(f)$$

where

$$\mathcal{L}(f) = \begin{cases} \text{small} & \text{if } f \text{ produces naturally looking images,} \\ \text{large} & \text{otherwise.} \end{cases}$$

Image Generation with a Neural Network

Problem: we have to *train* the network, f, to do what we want

- find network parameters that minimize a loss function:

$$\min_f \mathcal{L}(f)$$

where

$$\mathcal{L}(f) = \begin{cases} \text{small} & \text{if } f \text{ produces naturally looking images,} \\ \text{large} & \text{otherwise.} \end{cases}$$

Problem: we don't really know what makes an image look "real"

Image Generation with a Neural Network

Problem: we have to *train* the network, f, to do what we want

- find network parameters that minimize a loss function:

$$\min_f \mathcal{L}(f)$$

where

$$\mathcal{L}(f) = \begin{cases} \text{small} & \text{if } f \text{ produces naturally looking images,} \\ \text{large} & \text{otherwise.} \end{cases}$$

Problem: we don't really know what makes an image look "real"

Solution: we **learn** the loss function as well

Generative Adversial Networks [Goodfellow *et al.*, 2014]

Learn two networks at the same time:

- generator network: $f : \mathcal{Z} \to \mathcal{I}$
- discriminator network: $g : \mathcal{I} \to [0, 1]$ "how confident is g that x is real?"
  - $g(x) = 0$: definitely fake, not real
  - $g(x) = 1$: definitely real, not fake
  - $g(x) \approx \frac{1}{2}$: can't tell

- for real images, $I_1, \ldots, I_n$ and random inputs $z_1, \ldots, z_n$
- train g to have high values on real images and low values on outputs of f

$$\max_g V(f, g) \quad \text{for } V(f, g) = \frac{1}{n} \sum_{i=1}^{n} \log(g(I_i)) + \frac{1}{n} \sum_{j=1}^{n} \log(1 - g(f(z_j)))$$

- train f to make it impossible for g to make good predictions

$$\min_f \max_g V(f, g)$$

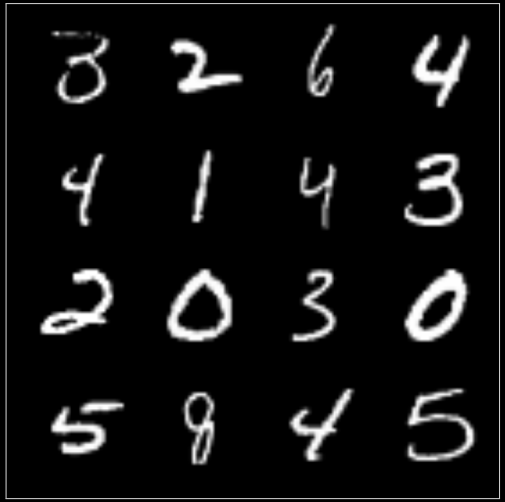
Generative Adversial Networks [Goodfellow *et al.*, 2014]

Think of:

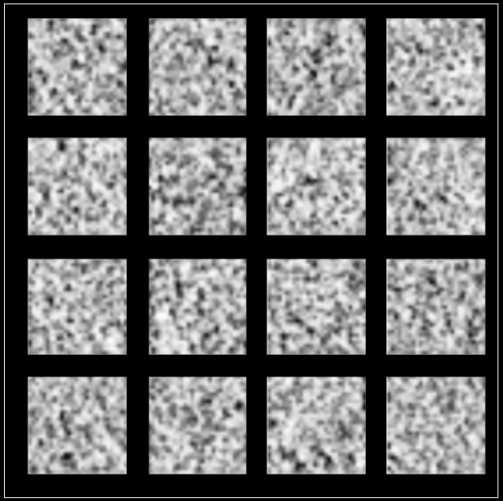
- generator network: art forger, produces images that look like old paintings

- discriminator network: art expert, tried to distinguish fakes from real paintings
- both learn to get better over time

Examples: generating digits

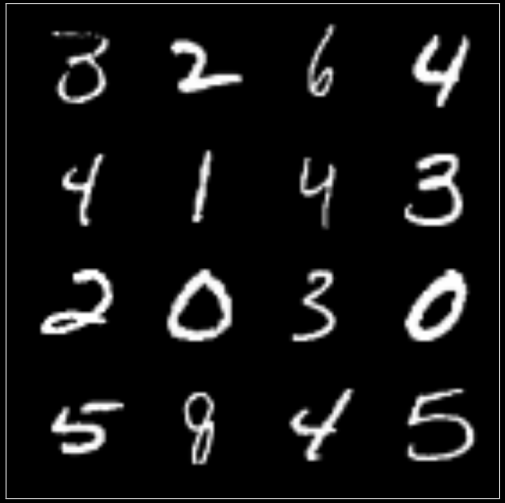


examples digits (real)



network output before training

Examples: generating digits



examples digits (real)

network output after training

Image Generation: Faces

[Radford, Metz, Chintala. *"Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks"*, ICLR 2016]

Image Generation: Bedrooms

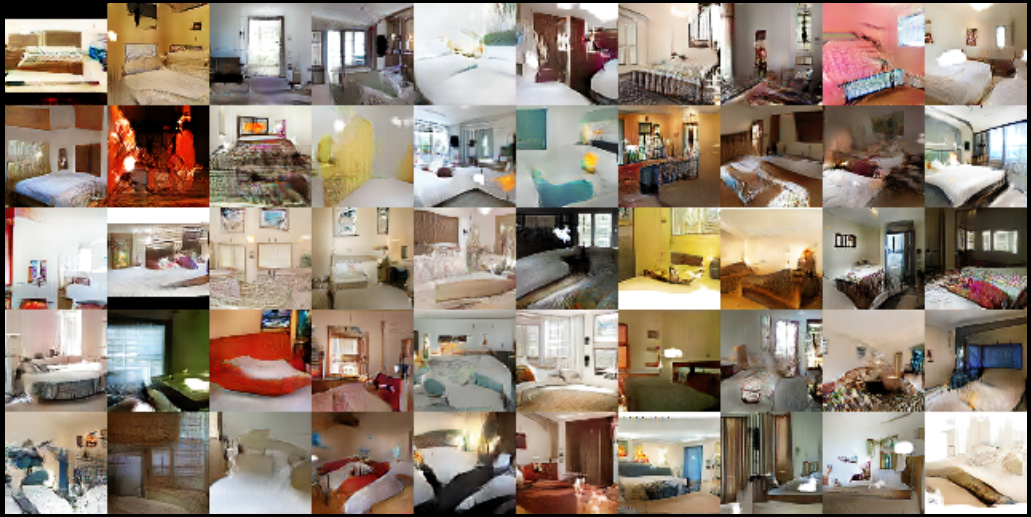


Image Generation: Album Covers

[Radford, Metz, Chintala. *"Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks"*, ICLR 2016]

Image Generation: General Natural Objects

[Salimans, Goodfellow, Zaremba, Cheung, Radford, Chen. *"Improved Techniques for Training GANs"*, arXiv:1606.03498 [cs.LG] 2016]

Extension: Conditional Image Generation

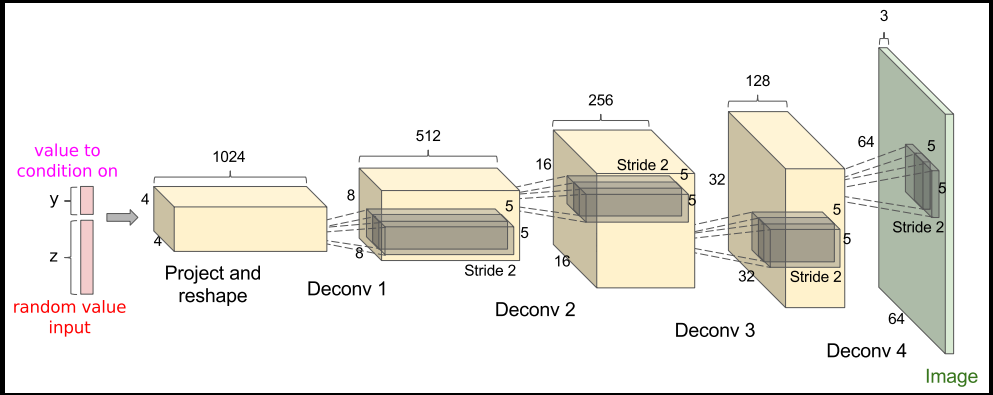
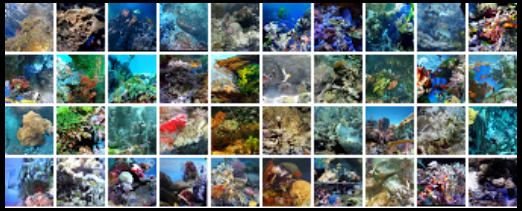


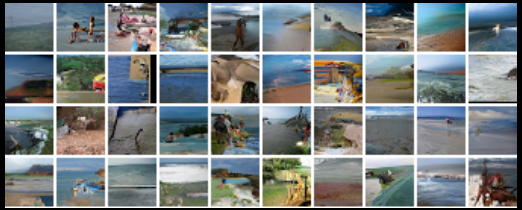
Image: adapted from https://openai.com/blog/generative-models/

Conditional Image Generation: One Network for Many Object Classes

African Elephant



Coral Reef

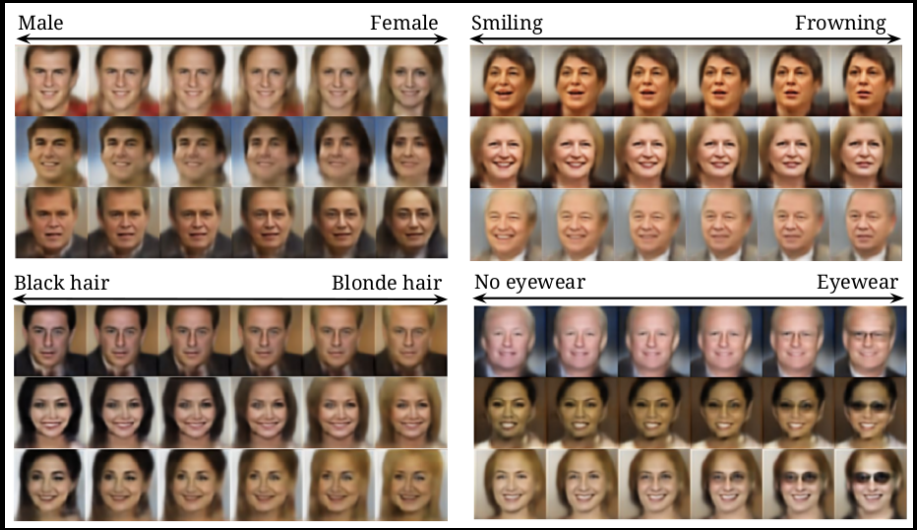


Sandbar

Sorrel horse

Images: [Van den Oord, Kalchbrenner, Vinyals, Espeholt, Graves, Kavukcuoglu. *"Conditional Image Generation with PixelCNN Decoders"*, arXiv:1606.05328]

Conditional Image Generation: From Properties to Images



Images: [Yan, Yang, Sohn, Lee. *"Attribute2Image: Conditional Image Generation from Visual Attributes"*, ECCV 2016]

Conditional Image Generation: From Descriptions to Images

"all black bird with a distinct thick, rounded bill"

"this small bird has a yellow breast, brown crown, and black superciliary"

[Reed, Akata, Yan, Logeswaran, Schiele, Lee. *"Generative Adversarial Text to Image Synthesis"*, ICML 2016]

Image Generation by Optimization

Image Generation by Optimization

Well known: construct new image by solving an optimization problem, e.g.

- observation y, e.g. noisy image
- regularizer that expresses additional information, e.g. smoothness

$$\min_{\text{all images } x} \text{dissimilarity}(x, y) + \lambda \, \text{regularizer}(x)$$

Image Generation by Optimization

Well known: construct new image by solving an optimization problem, e.g.

- observation y, e.g. noisy image
- regularizer that expresses additional information, e.g. smoothness

$$\min_{\text{all images } x} \ \text{dissimilarity}(x, y) + \lambda \, \text{regularizer}(x)$$

For example, *total variation denoising*

$$\text{dissimilarity}(x, y) = \|x - y\|^2 \qquad \text{regularizer}(x) = \|x\|_{TV}$$

Much more flexible, if we **learn the dissimilarity and regularizer.**

Artistic Style Transfer [Gatys *et al.*, 2016]

Neckarfront in Tübingen, Germany

Images: [Gatys, Ecker, Bethge. *"Image Style Transfer Using Convolutional Neural Networks"*, CVPR 2016]

Artistic Style Transfer [Gatys *et al.*, 2016]

Neckarfront in Tübingen, Germany

The Shipwreck of the Minotaur
J.M.W. Turner, 1805

Images: [Gatys, Ecker, Bethge. *"Image Style Transfer Using Convolutional Neural Networks"*, CVPR 2016]

Artistic Style Transfer [Gatys *et al.*, 2016]

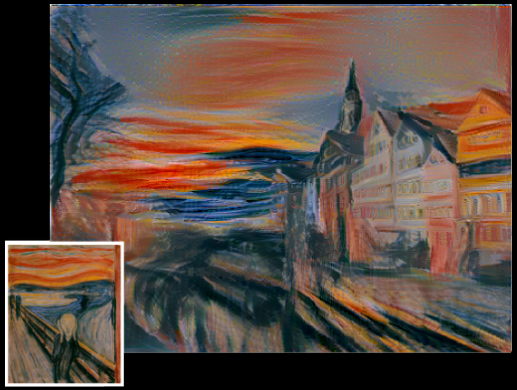
Neckarfront in Tübingen, Germany

The Starry Night
Vincent van Gogh, 1889

Images: [Gatys, Ecker, Bethge. *"Image Style Transfer Using Convolutional Neural Networks"*, CVPR 2016]

Artistic Style Transfer [Gatys *et al.*, 2016]

Neckarfront in Tübingen, Germany

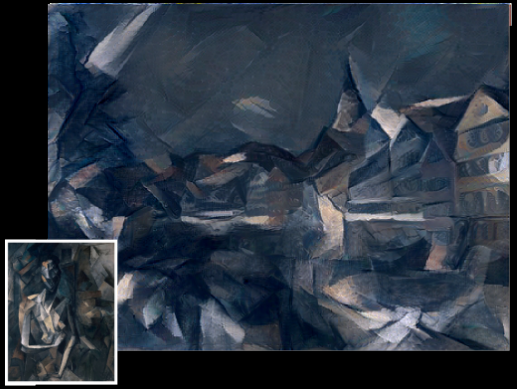


Der Schrei der Natur
Edvard Munch, 1893

Images: [Gatys, Ecker, Bethge. *"Image Style Transfer Using Convolutional Neural Networks"*, CVPR 2016]

Artistic Style Transfer [Gatys *et al.*, 2016]

Neckarfront in Tübingen, Germany

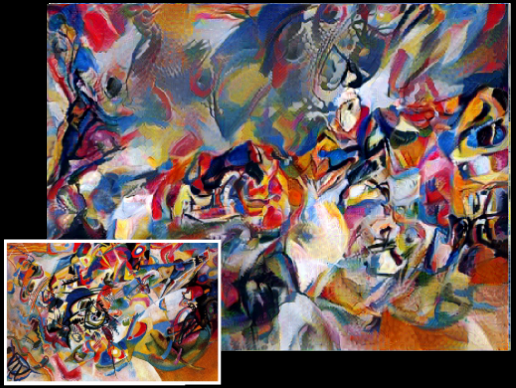


Femme nue assise
Pablo Picasso, 1910

Images: [Gatys, Ecker, Bethge. *"Image Style Transfer Using Convolutional Neural Networks"*, CVPR 2016]

Artistic Style Transfer [Gatys *et al.*, 2016]

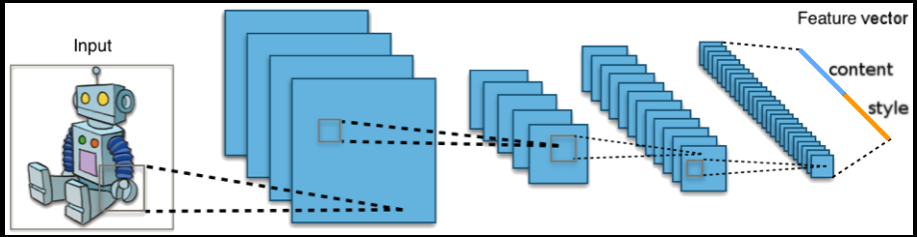
Neckarfront in Tübingen, Germany



Composition VII
Wassily Kandinsky, 1913

Images: [Gatys, Ecker, Bethge. *"Image Style Transfer Using Convolutional Neural Networks"*, CVPR 2016]

Image Generation by Optimization



Network learns separate feature representations: φ_{content} for content, φ_{style} for style.

Given two images:

- y: image from which we want to preserve the content
- z: image from which we want to preserve the style

Construct new image by trying to match content of one, and style of the other:

$$\min_{\text{all images } x} \|\varphi_{\text{content}}(x) - \varphi_{\text{content}}(y)\|^2 + \lambda\|\varphi_{\text{style}}(x) - \varphi_{\text{style}}(z)\|^2$$

Summary: Generating Natural Images (with Neural Networks)

Very recent trend: use neural networks to generate images.
Not yet clear what it's going to be useful for...

Generative adversarial networks: train two networks

- generator: produce output images from random inputs
- discriminator: tries to distinguish the generated images from real ones

Sampled images are good, if the generator can fool the discriminator.

Image generation by optimization, e.g. Style Transfer

- more flexible, almost arbitrary criteria
- less efficient, because each output needs solving an optimization problem

Thanks to...

All people who's research or material I presented...

- the organizers of CVPR 2016
- Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, Michael J. Black
- Jamie Shotton, Andrew Fitzgibbon, Andrew Blake, Alex Kipman, Mark Finocchio, Richard Moore, Toby Sharp
- Leon Gatys, Alexander Ecker, Matthias Bethge
- Baochen Sun, Kate Saenko
- Yaroslav Ganin, Victor Lempitsky
- Soumith Chintala, Yann LeCun
- Alec Radford, Luke Metz, Soumith Chintala
- Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio
- Andrej Karpathy, Pieter Abbeel, Greg Brockman, Peter Chen, Vicki Cheung, Rocky Duan, Ian Goodfellow, Durk Kingma, Jonathan Ho, Rein Houthooft, Tim Salimans, John Schulman, Ilya Sutskever, Wojciech Zaremba
- Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, Koray Kavukcuoglu
- Xinchen Yan, Jimei Yang, Kihyuk Sohn, Honglak Lee
- Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, Honglak Lee
- Stephan Richter, Vibhav Vineet, Stefan Roth, Vladlen Koltun
- Aphex34
- Allen Gersho
- Tim O'Shea
- www.engadget.com
- www.pixabay.com
- www.texample.net
- www.image-net.org

Funding Sources:

IST AUSTRIA
Institute of Science and Technology

erc

Summary

Synthetic Images (for Classifier Training)

- based on geometric models, e.g. human body, and physically accurate rendering
- successful in specific application (e.g. Microsoft Kinect) but not everywhere

Domain Adaptation

- study when it is successful to train on one kind of data and test on another kind
- important insight: if domains differ, make them *indistiguishable*

Recent Trend (since 2014): Image Generation with Neural Networks

- generative adversarial networks
- image generations by optimization
- what are the applications?