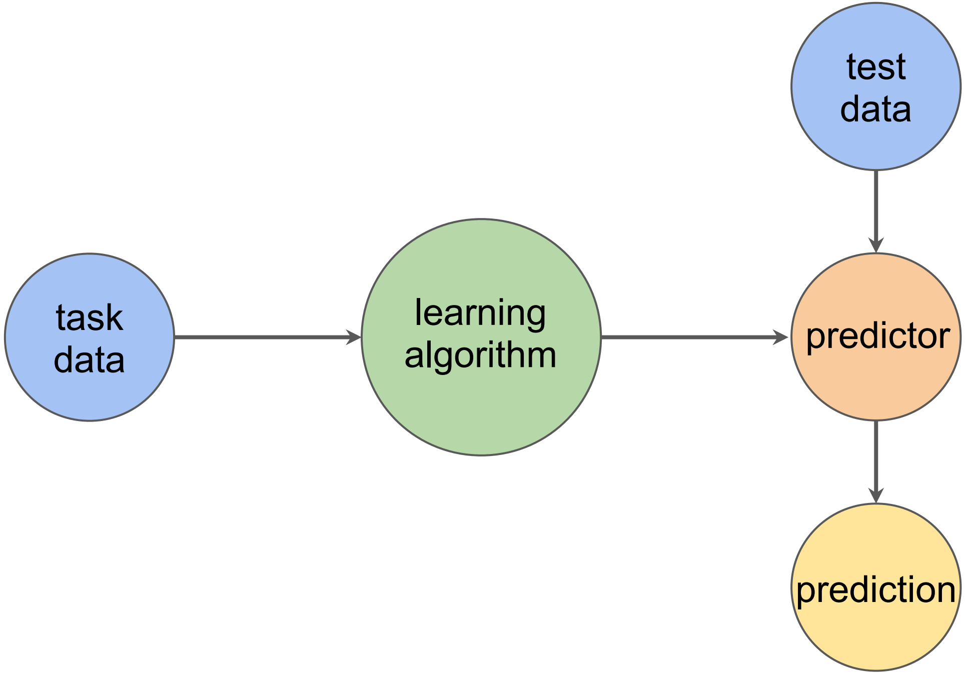
Massachusetts Institute of Technology

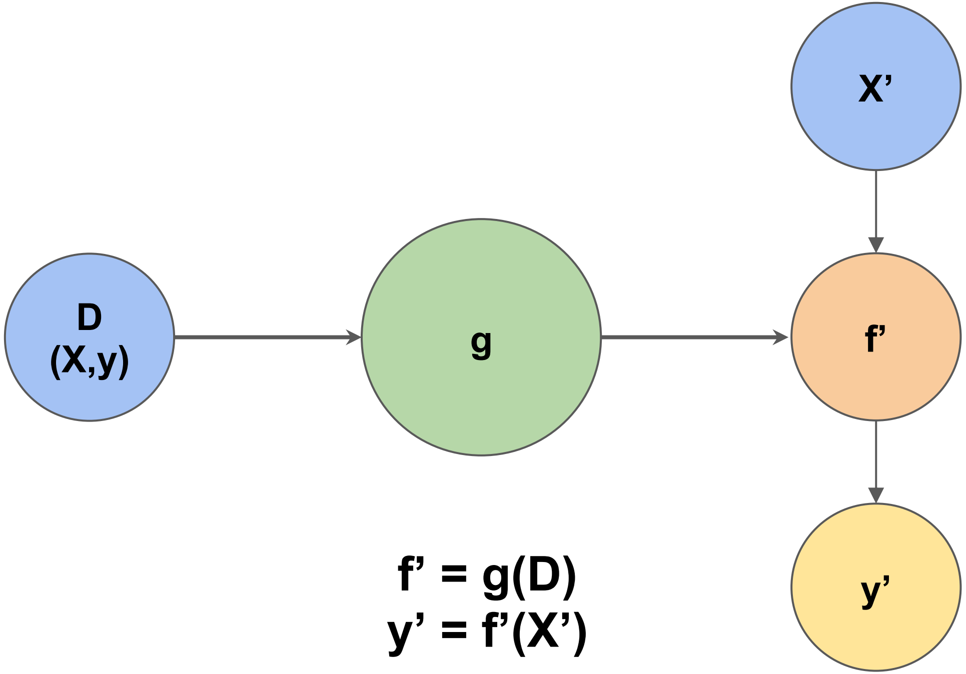
Meta Learning

MIT
Iddo Drori, Fall 2020

Supervised Learning



Supervised Learning



$$f' = g(D)$$
$$y' = f'(X')$$

Image Analogies (style transfer before CNNs)

- Source data Ds = (Xs, Ys)
- Target data Dt = (Xt, ?)
- Source and target data distributions are the same
- Missing Yt
- Xs:Ys :: Xt:?
- Supervised learning

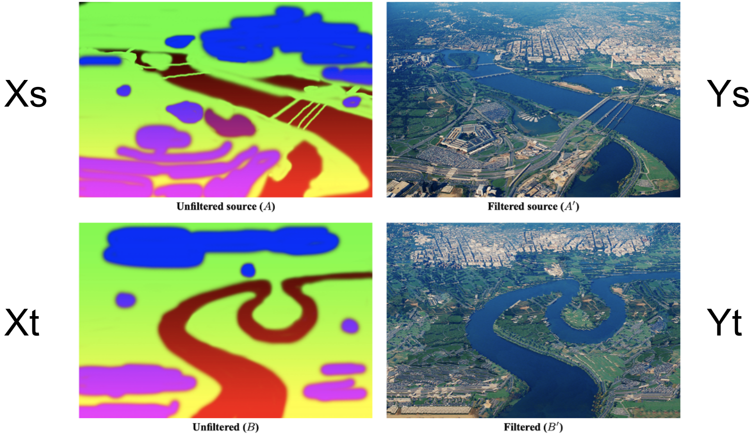
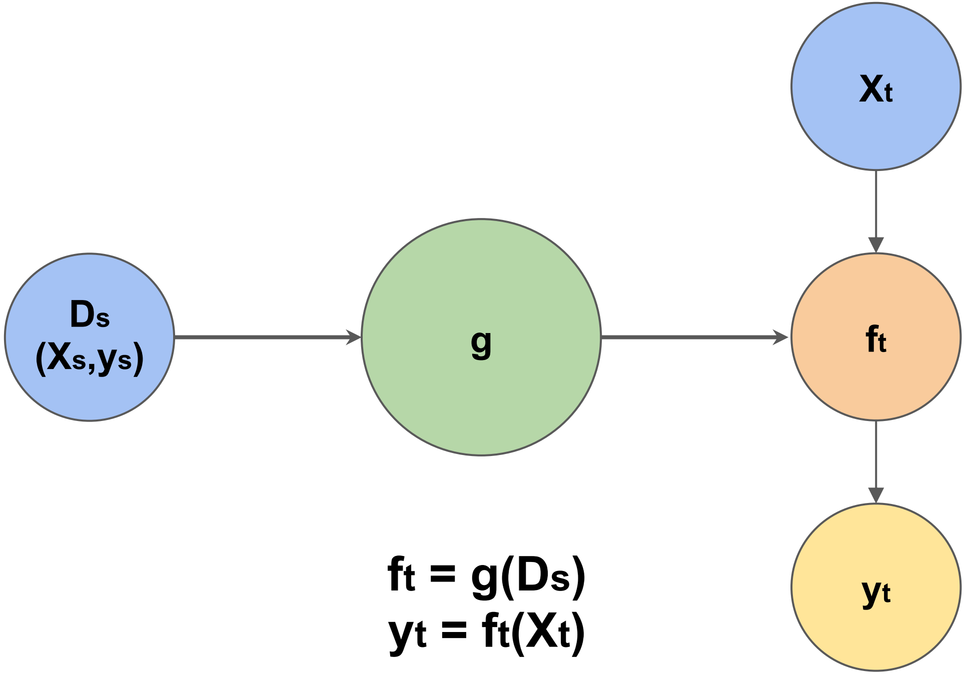


Figure source: Image Analogies, Hertzmann et al, 2001

Massachusetts Institute of Technology

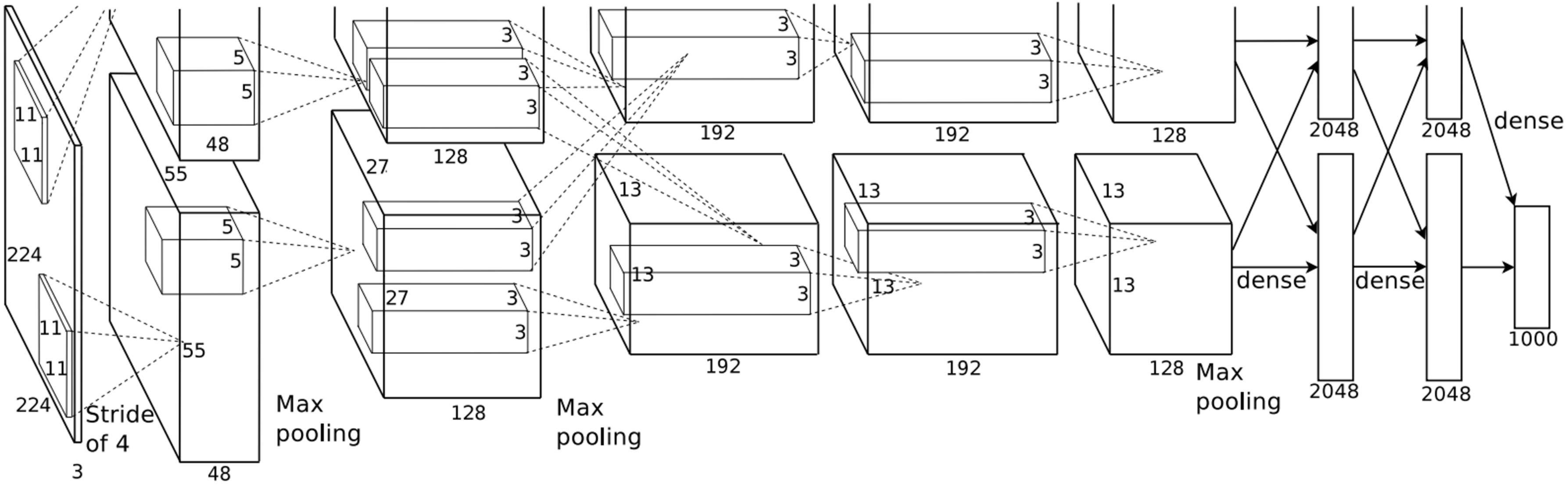
Supervised Learning



$$f_t = g(D_s)$$
$$y_t = f_t(X_t)$$

CNNs Overview

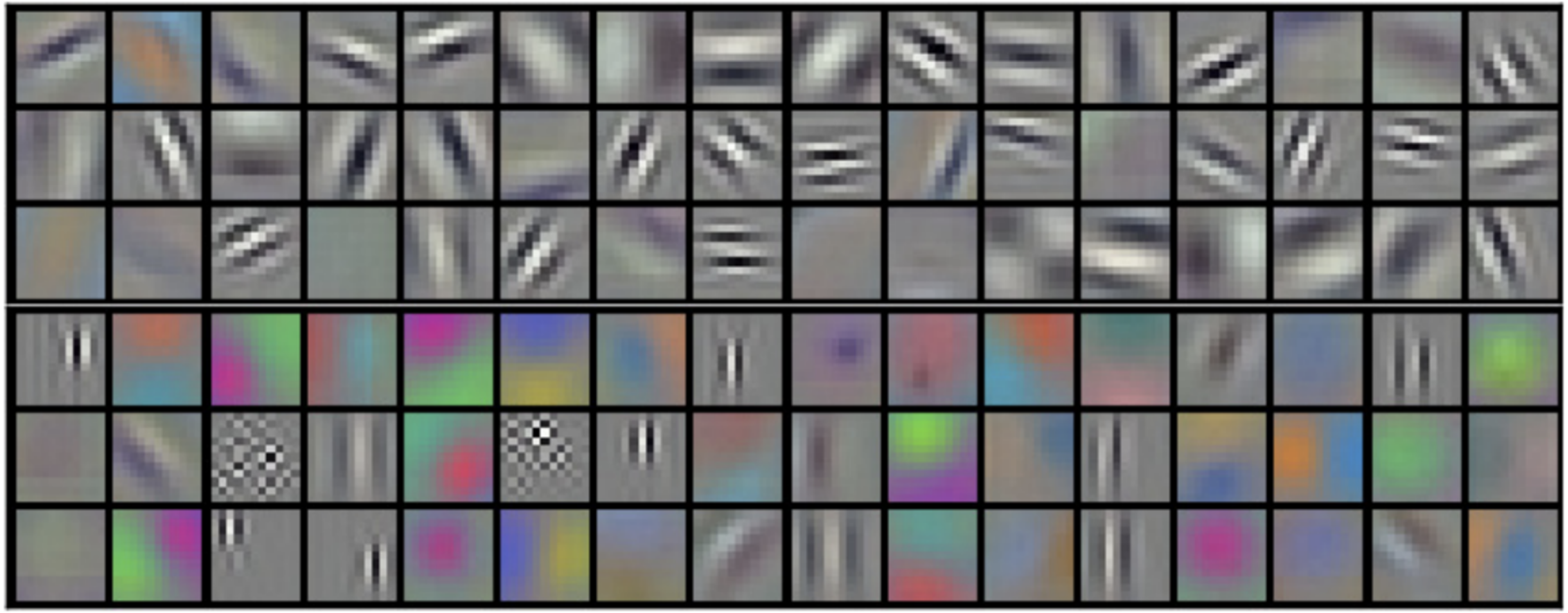
ImageNet



CNN architecture

Figure source: ImageNet Classification with Deep Convolutional Neural Networks, Krizhevsky et al, NIPS 2012

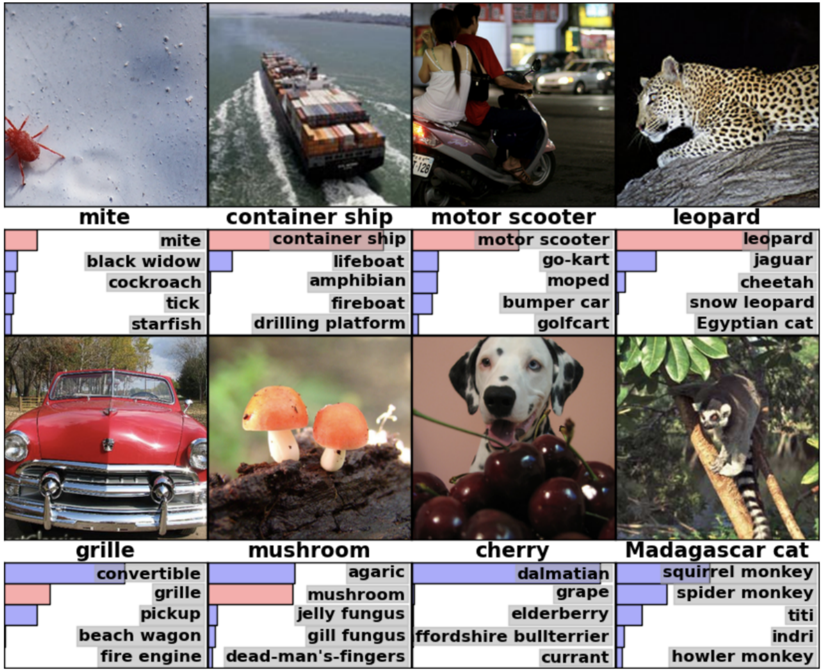
ImageNet Filters



convolutional kernels of first layer

Source: ImageNet Classification with Deep Convolutional Neural Networks, Krizhevsky et al, NIPS 2012

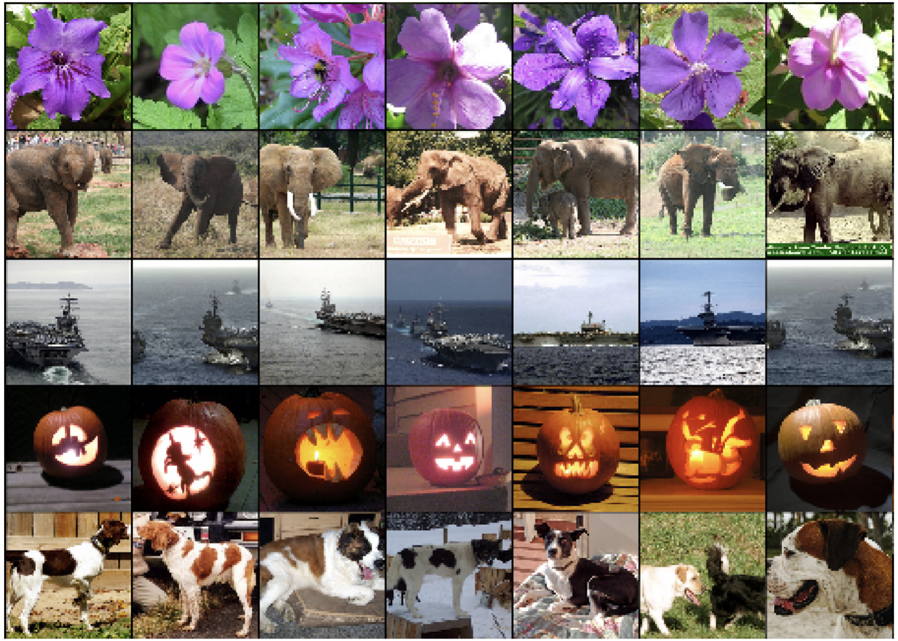
ImageNet Results



most probable classes

Figure source: ImageNet Classification with Deep Convolutional Neural Networks, Krizhevsky et al, NIPS 2012

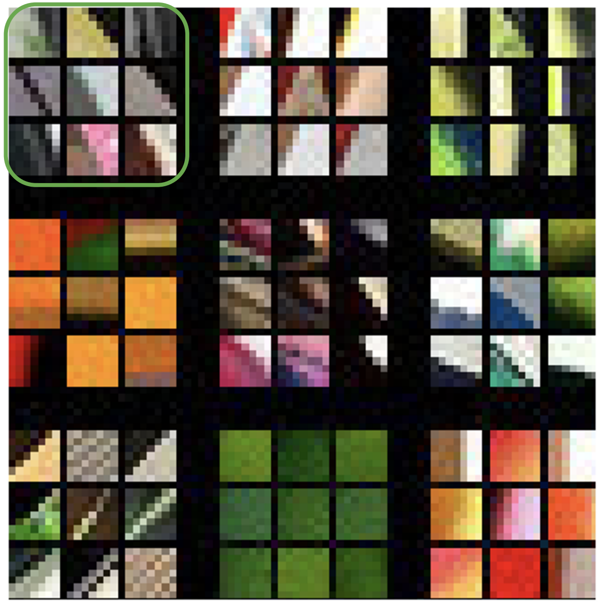
ImageNet Results



test images *training images with last hidden layer feature vectors closest to test feature vector*

Figure source: ImageNet Classification with Deep Convolutional Neural Networks, Krizhevsky et al, NIPS 2012

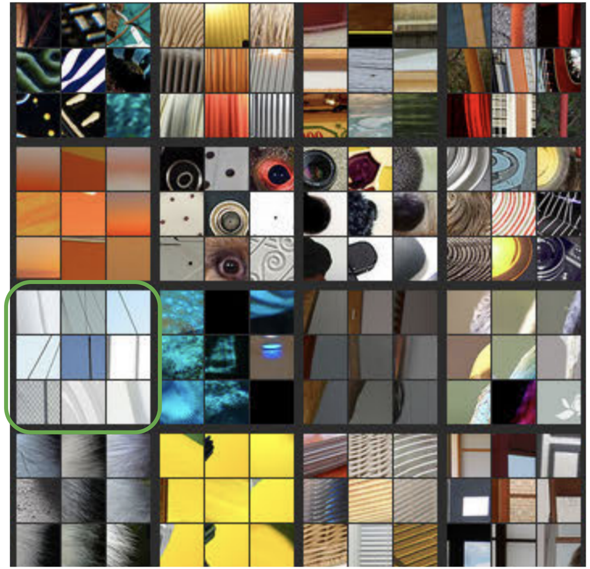
Understanding ImageNet



Which training image patches do specific activation units in layer 1 respond to?

Figure source: Visualizing and understanding convolutional networks, Zeiler and Fergus, ECCV 2014.

Understanding ImageNet



Which training image patches do specific activation units in layer 2 respond to?

Figure source: Visualizing and understanding convolutional networks, Zeiler and Fergus, ECCV 2014.

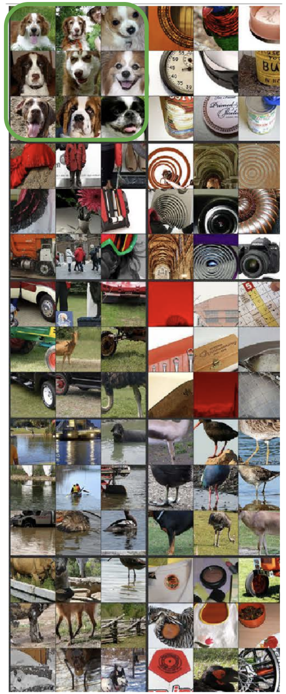
Understanding ImageNet

Which training image patches do specific activation units in layer 3 respond to?

Figure source: Visualizing and understanding convolutional networks, Zeiler and Fergus, ECCV 2014.

Massachusetts Institute of Technology

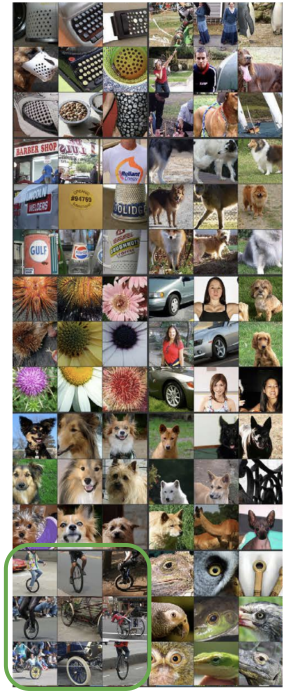
Understanding ImageNet



Which training image patches do specific activation units in layer 4 respond to?

Figure source: Visualizing and understanding convolutional networks, Zeiler and Fergus, ECCV 2014.

Understanding ImageNet



Which training image patches do specific activation units in layer 5 respond to?

Figure source: Visualizing and understanding convolutional networks, Zeiler and Fergus, ECCV 2014.

Input Maximizing Activation

$$\underset{x}{\operatorname{argmax}}\, a_i^l(W, x)$$

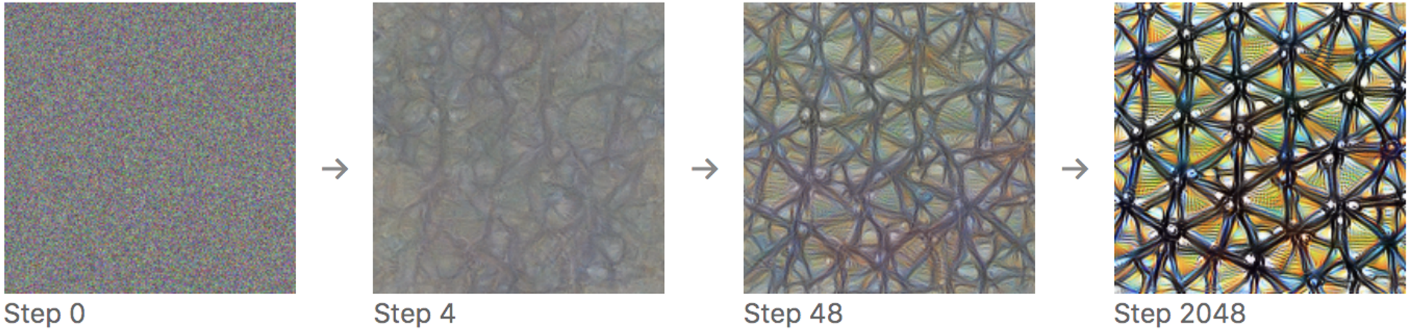
given trained network with weights W

find input x which maximizes activation of unit i at leyer l

starting from x as random noise perform gradient ascent on x

Source: Visualizing Higher-Layer Features of a Deep Network, Erhan et al, 2009.

Input Maximizing Activation



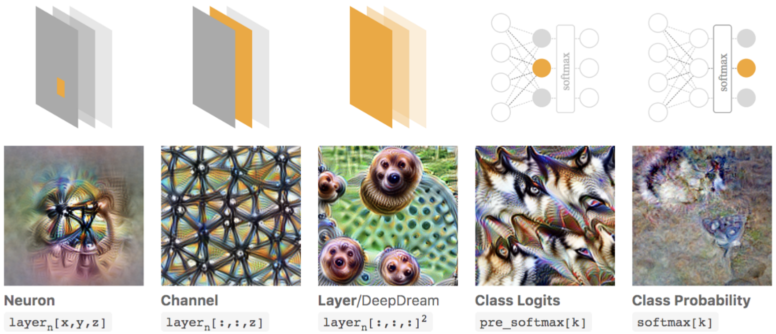
given trained network with weights W

find input x which maximizes activation

starting from x as random noise perform gradient ascent on x

Figure source: Feature Visualization, Olah et al, Distill, 2017.

Input Maximizing Different Objectives



given trained network with weights W

find input x which maximizes different objectives

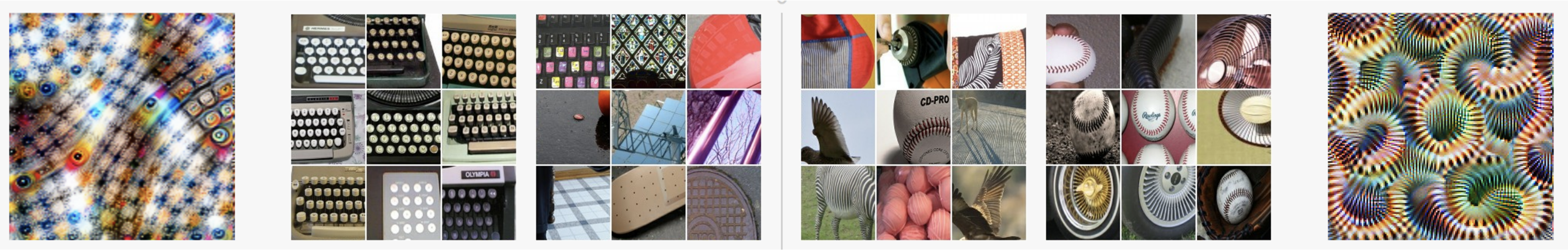
starting from x as random noise perform gradient ascent on x

Figure source: Feature Visualization, Olah et al, Distill, 2017.

Training Patches vs. Optimization

Figure source: Feature Visualization, Olah et al, Distill, 2017.

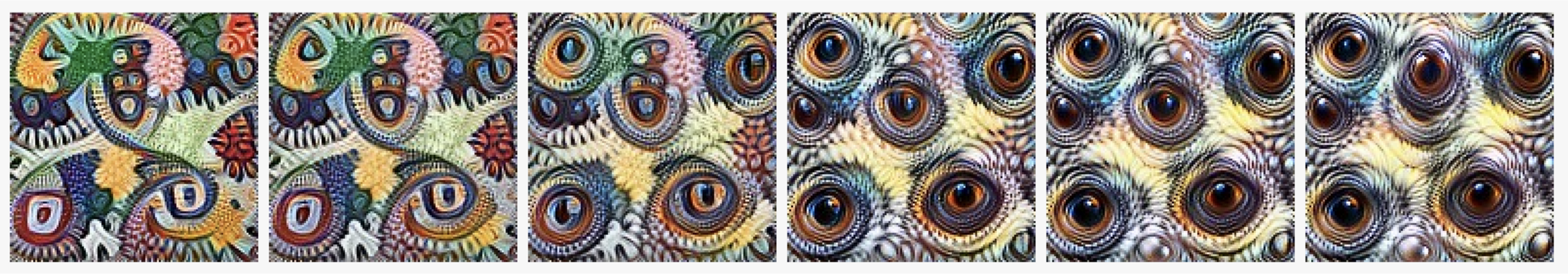
Maximization and Minimization



negative optimized | *maximum negative patches* | *slightly negative patches* | *slightly positive patches* | *maximum positive patches* | *positive optimized*

Figure source: Feature Visualization, Olah et al, Distill, 2017.

Interactions Between Activations



optimizing activation a

joint optimization

linear interpolation between objectives

optimizing activation b

Figure source: Feature Visualization, Olah et al, Distill, 2017.

Visualizing Every Network Activation

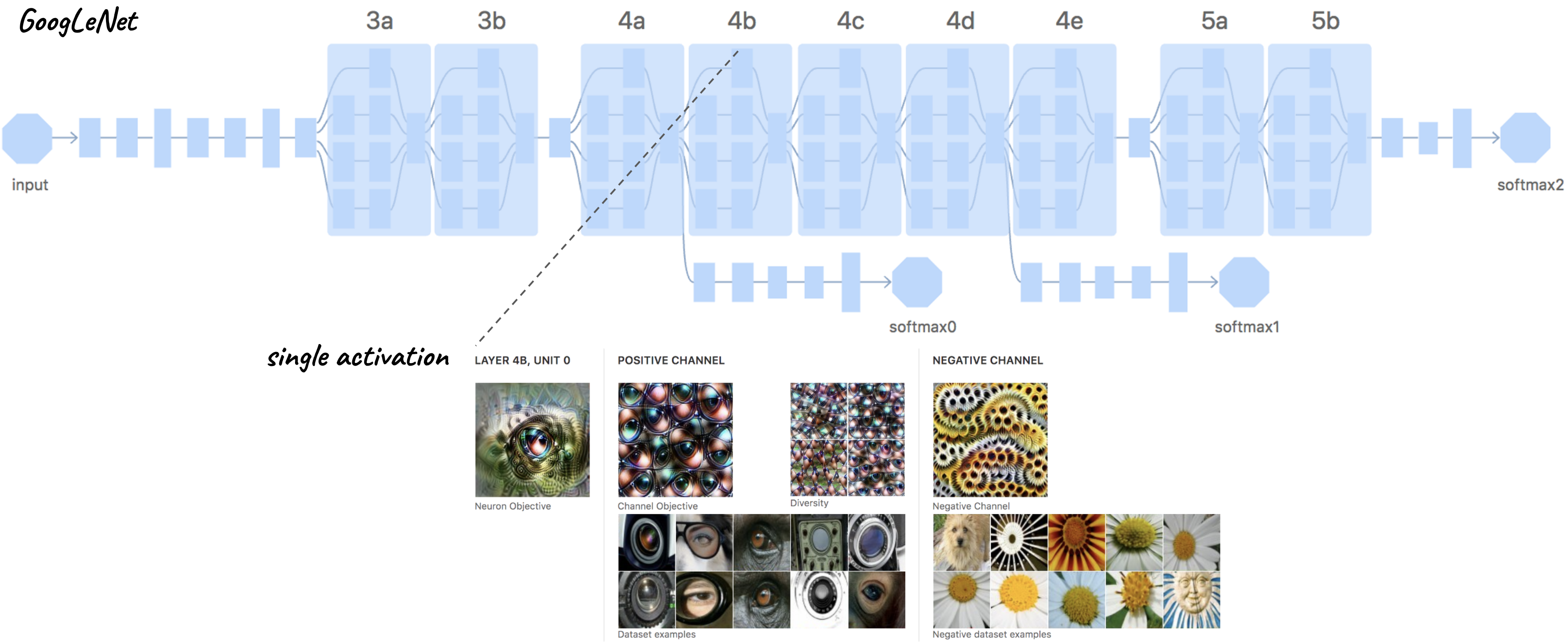
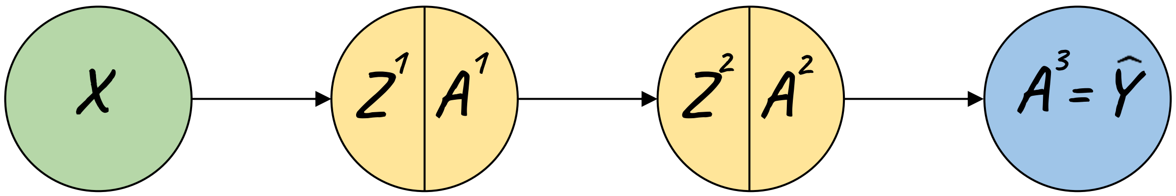


Figure source: Feature Visualization, Olah et al, Distill, 2017
https://distill.pub/2017/feature-visualization/appendix

Transfer Learning

- Task 1: learn to recognize animals given many (10M) examples which are not horses
- Keep layers from task 1, re-train on last layer
- Task 2: learn to recognize horses given a few (100) examples



Siamese Networks

CNN's for Face Recognition

Problem: single example for each person.

Solution: learn similarity rather than identity.

Reduce to verification: are x_i and x_j the same person?

Encode x as $f(x)$ using CNN

Compare $f(x_i)$ with $f(x_j)$ by $d(f(x_i), f(x_j))$

CNN's for Face Recognition

Train on input pairs (x_i, x_j)

Label each pair $y=1$ if x_i and x_j are same person, $y=0$ otherwise

Use CNN encoding of pair $f(x_i), f(x_j)$

Loss function

$$\mathcal{L}(x_i, x_j) = \mathcal{L}(y, \hat{y}) \quad \hat{y} = g\left(d\left(f(x_i), f(x_j)\right)\right)$$

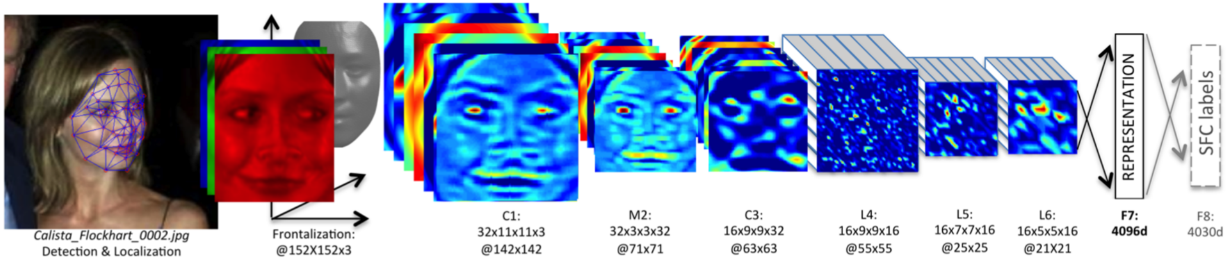


Figure source: Taigman et al, 2014

Style Transfer

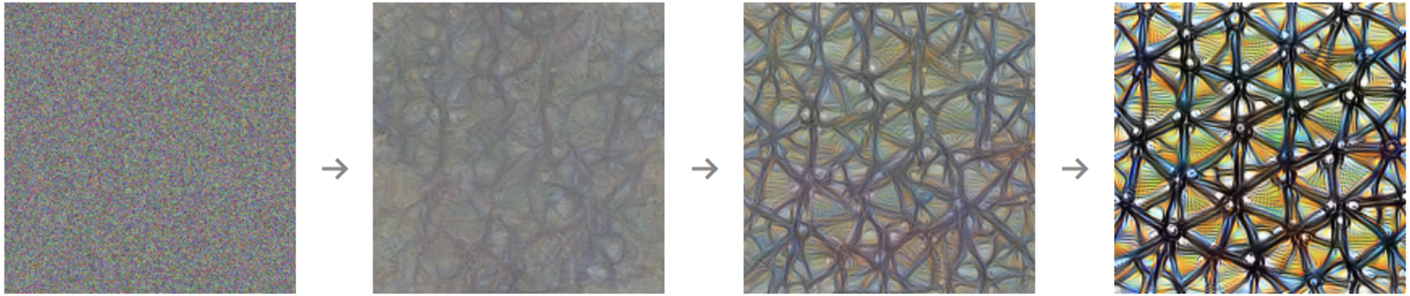
Input Maximizing Activation

$$\underset{x}{\operatorname{argmax}}\, a_i^l(W, x)$$

given trained network with weights W
find input x which maximizes activation of unit i at leyer l
starting from x as random noise perform gradient ascent on x

Source: Visualizing Higher-Layer Features of a Deep Network, Erhan et al, 2009.

Input Maximizing Activation



Step 0 | Step 4 | Step 48 | Step 2048

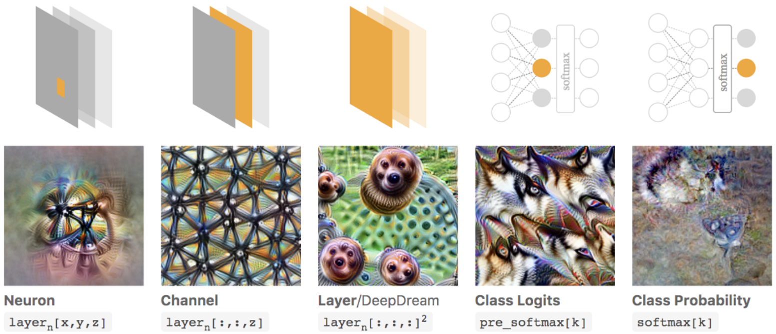
given trained network with weights W

find input x which maximizes activation

starting from x as random noise perform gradient ascent on x

Figure source: Feature Visualization, Olah et al, Distill, 2017.

Input Maximizing Different Objectives



given trained network with weights W

find input x which maximizes different objectives

starting from x as random noise perform gradient ascent on x

Figure source: Feature Visualization, Olah et al, Distill, 2017.

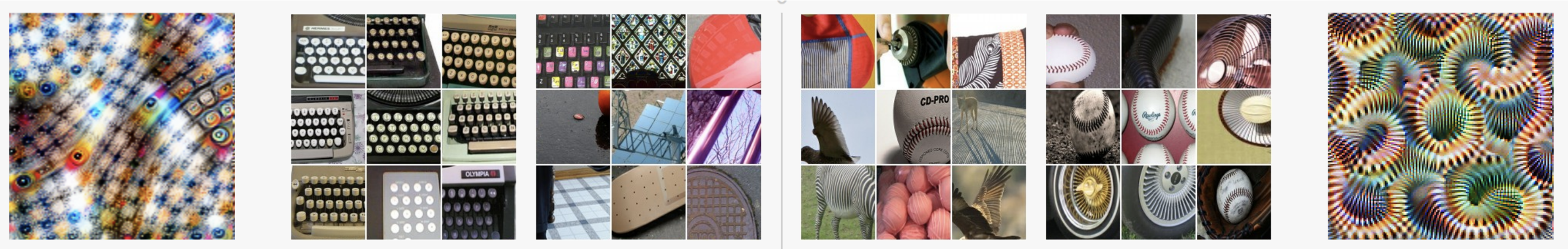
Training Patches vs. Optimization

training patches maximizing activation

optimization of input maximizing activation

Figure source: Feature Visualization, Olah et al, Distill, 2017.

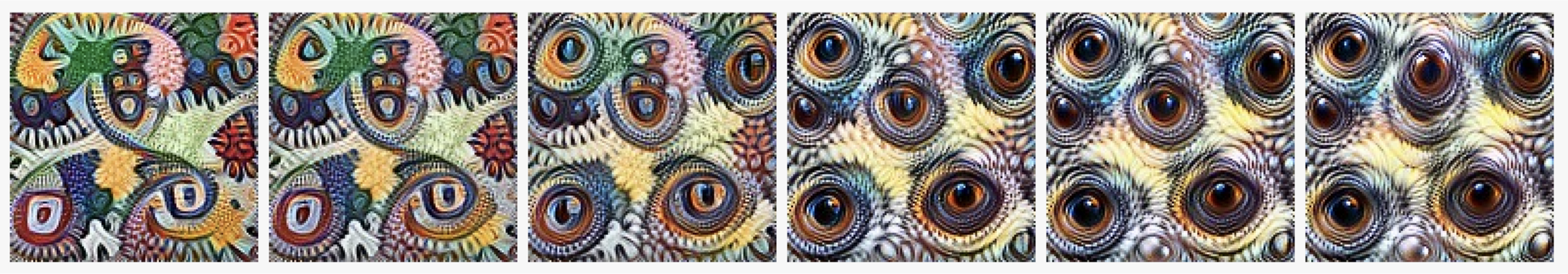
Maximization and Minimization



negative optimized | *maximum negative patches* | *slightly negative patches* | *slightly positive patches* | *maximum positive patches* | *positive optimized*

Figure source: Feature Visualization, Olah et al, Distill, 2017.

Interactions Between Activations



optimizing activation a *joint optimization* *optimizing activation b*

linear interpolation between objectives

Figure source: Feature Visualization, Olah et al, Distill, 2017.

Gram Matrix of Channels

$$G^{l}_{kk'} = \sum_{k=1}^{n_C} \sum_{k'=1}^{n_C} a^{l}_{ijk} a^{l}_{ijk'}$$

Gram matrix

$$-\sum \sum \frac{g_A \cdot g_B}{\|g_A\| \|g_B\|}$$

add term to optimization objective

Source: Feature Visualization, Olah et al, Distill, 2017
https://distill.pub/2017/feature-visualization/appendix

Optimization with Gram Matrix Objective

make results be different from each other: diversity

Figure source: Feature Visualization, Olah et al, Distill, 2017

Style Transfer

Massachusetts Institute of Technology

content

style

style transfer

Figure source: Image style transfer using convolutional neural networks, Gatys et al, CVPR 2016.

Style Transfer

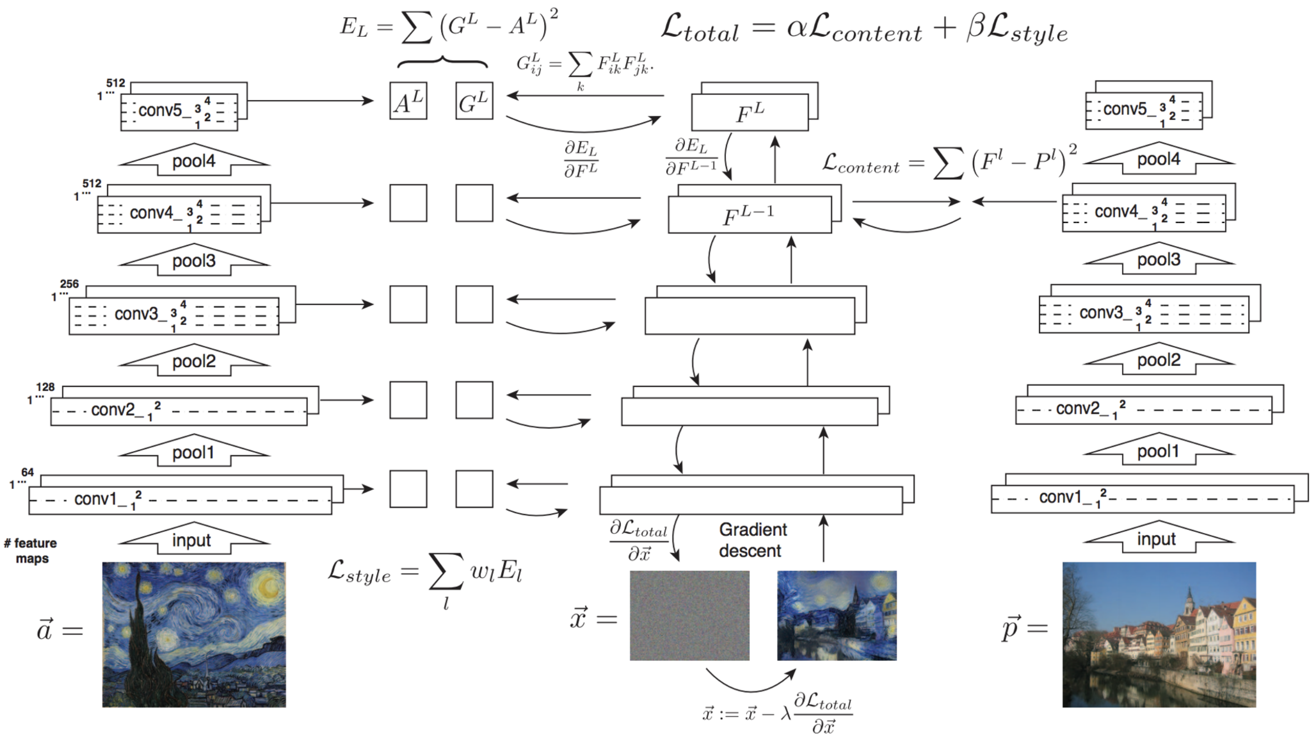


Figure source: Image style transfer using convolutional neural networks, Gatys et al, CVPR 2016.

Style Transfer

$$x = x - \frac{\partial \mathcal{L}(x)}{\partial x}$$

$$\mathcal{L}(x) = \alpha \mathcal{L}_{content}(x, c) + (1 - \alpha)\mathcal{L}_{style}(x, s)$$

Initialize x to random noise or content image or style image
Gradient descent with loss function a linear combination of a style and content terms

Source: Image style transfer using convolutional neural networks, Gatys et al, CVPR 2016.

Massachusetts Institute of Technology

Style Transfer using Gram Matrix

$$\mathcal{L}^{\ell}_{content}(x,c) = \frac{1}{2}\left\|a_c^l - a_x^l\right\|^2 = \frac{1}{2}\sum_i\sum_j\left(a_{c_{ij}}^l - a_{x_{ij}}^l\right)^2$$

$$\mathcal{L}_{style}(x,s) = \frac{1}{(2n_w n_h n_c)^2}\lambda_l\sum_l\sum_k\sum_{k\prime}\left(G_{s_{kk\prime}}^l - G_{x_{kk\prime}}^l\right)^2$$

$$G_{s_{kk\prime}}^l = \sum_{k=1}^{n_c}\sum_{k'=1}^{n_c} a_{s_{ijk}}^l a_{s_{ijk\prime}}^l$$

content loss is element-wise sum of squares between activations
style loss depends on correlation between activations across channels

Source: Image style transfer using convolutional neural networks, Gatys et al, CVPR 2016.

Style Transfer

Figure source: Image style transfer using convolutional neural networks, Gatys et al, CVPR 2016.

GANs Overview

Generative Models

- Real data from real distribution
- Generate samples from model distribution
- Learn model distribution similar to real distribution

Generative Adversarial Networks

Photo-realistic faces synthesized using GANs: images are of high quality and diverse.

Figure source : thispersondoesnotexist.com

Coevolution

Game Theory

- Minimax optimization problem or saddle-point problem:

$$\min_{x} \max_{y} f(x, y)$$

Generative Adversarial Networks

Figure source: thiscatdoesnotexist.com whichfaceisreal.com

GAN Zoo

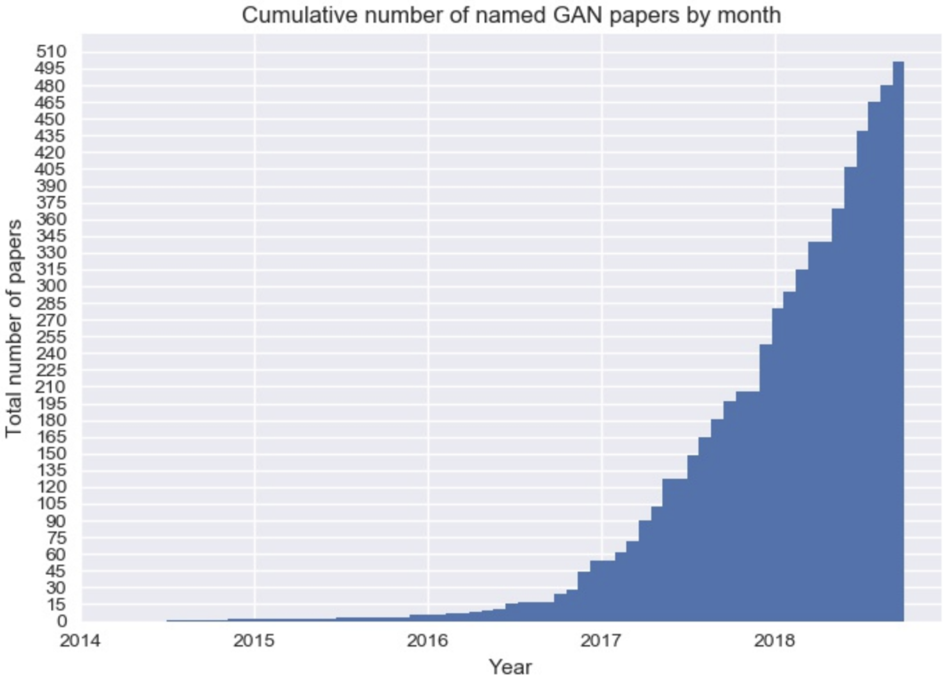
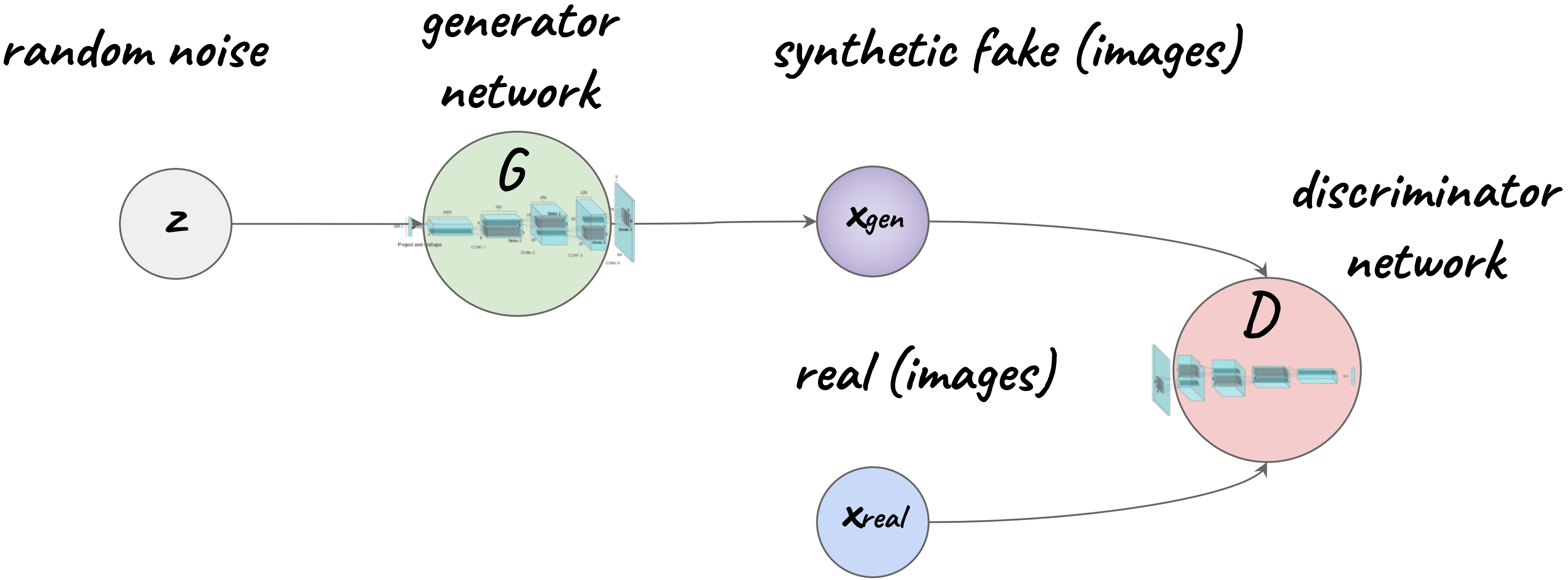


Figure source: https://github.com/hindupuravinash/the-gan-zoo

Generative Adversarial Network (GAN)



BigGAN Results (2019)

Figure source: Large scale GAN training for high fidelity image synthesis, Brock et al, ICLR 2019.

Transfer Learning

Image to Image Translation

- Source data Ds = (Xs, Ys)
- Target data Dt = (Xt, ?)
- Source and target data distributions are the same
- Target data is unlabeled
- Xs:Ys :: Xt:?
- Ys = fs(Xs) is unknown, estimate by ft
- Xs = invfs(Ys) is known, generate data pairs Ds = (Xs, Ys)
- Conditional GAN

Massachusetts Institute of Technology

Image to Image Translation

- Generate $D_s = (X_s, Y_s)$ from Y_s and $X_s = f_s^{-1}(Y_s)$
- Train conditional GAN:
  - Train conditional generator $f_t(X_s)$
  - Train discriminator on fake $(f_t(X_s), X_s)$ and real (Y_s, X_s)
- Apply generator f_t to target data X_t

Conditional GAN

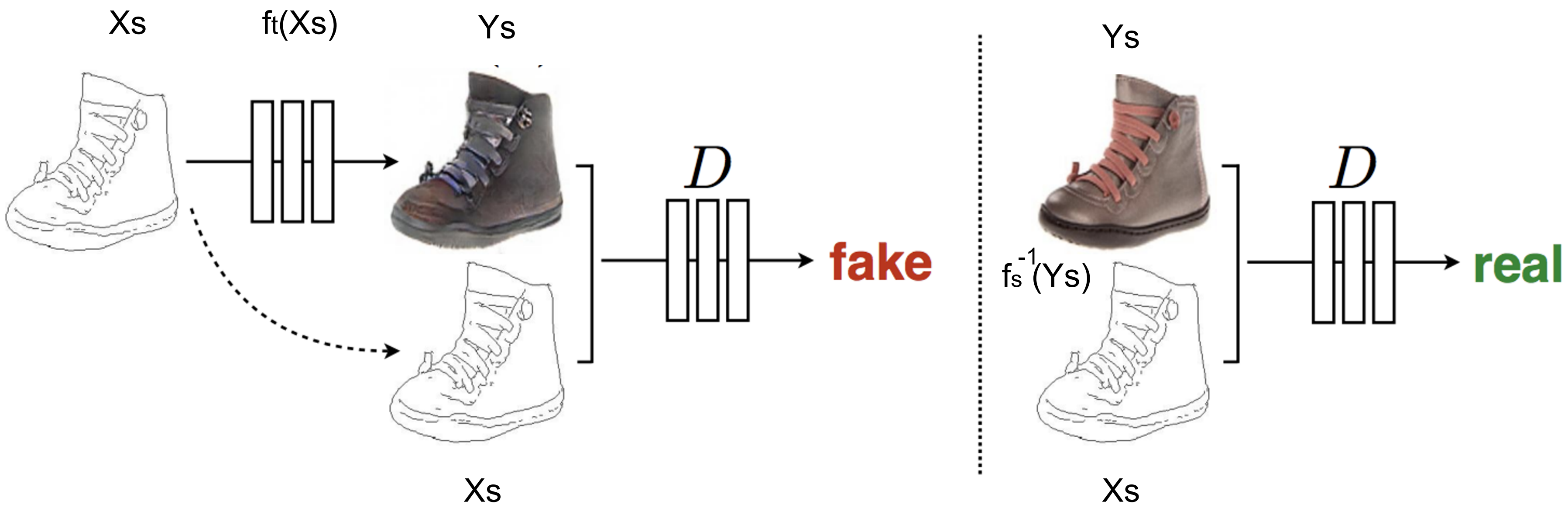


Figure source: Image-to-image translation with conditional adversarial networks, Isola et al, CVPR 2017.

Conditional GAN

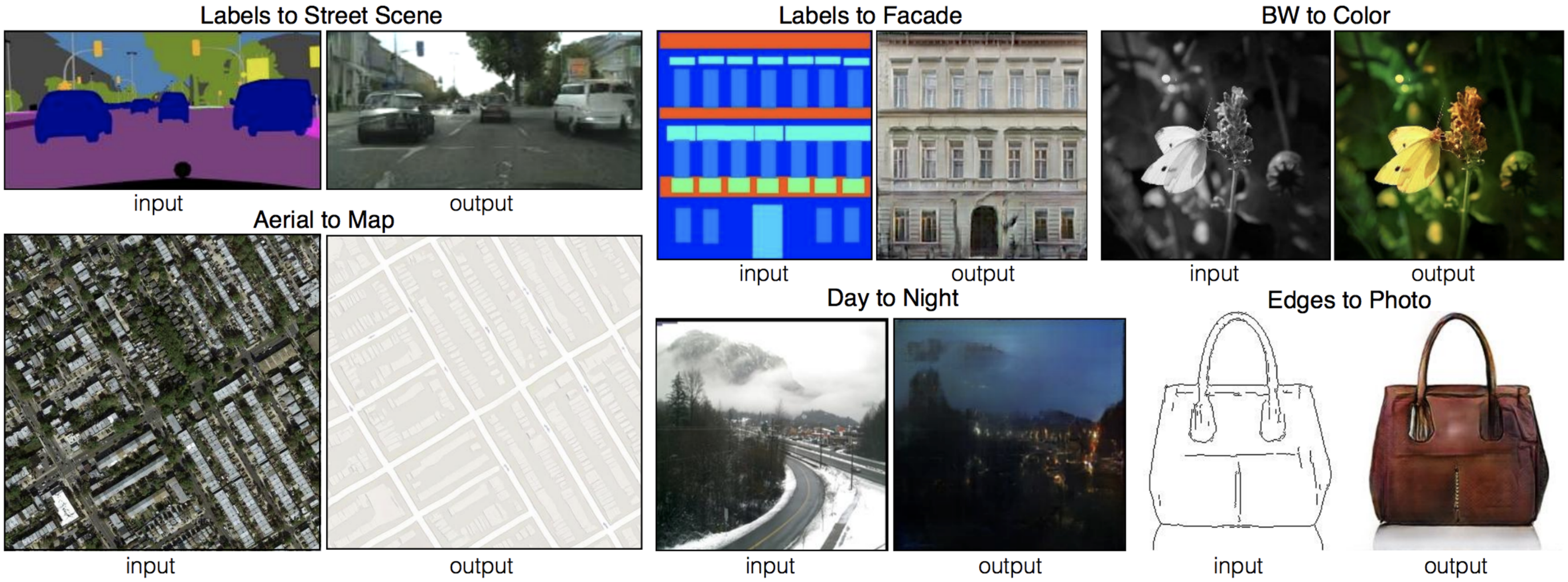


Figure source: Image-to-image translation with conditional adversarial networks, Isola et al, CVPR 2017.

Application

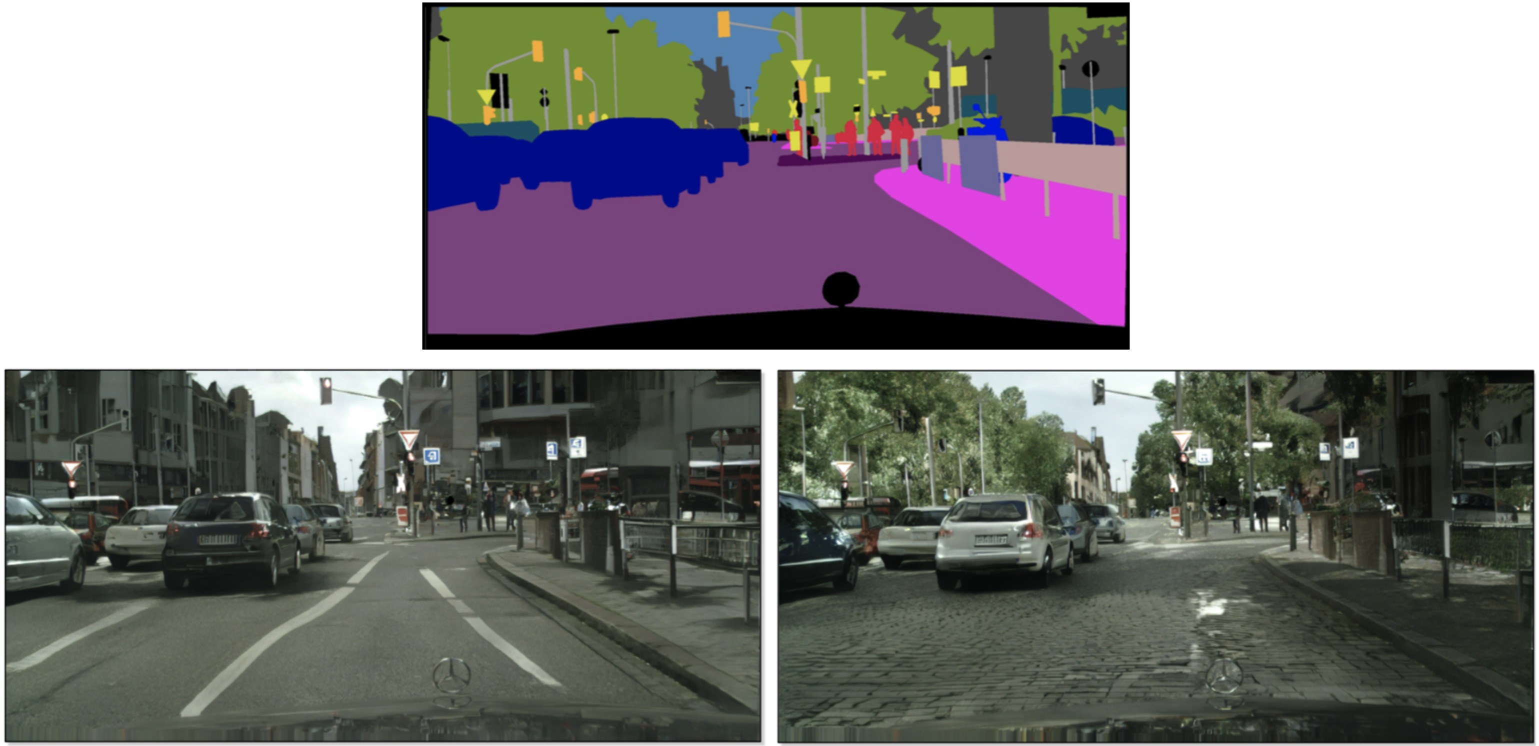


Figure source: High-Resolution image synthesis and semantic manipulation with conditional GANs, Wang et al, 2017.

Unpaired Image to Image Translation

- Cycle GAN
- Train generator g_1 from Xs to Ys
- Train generator g_2 from Ys to Xs
- Apply $g_2(g_1(Xs))$ and check for same value
- Apply $g_1(g_2(Ys))$ and check for same value

Cycle GAN

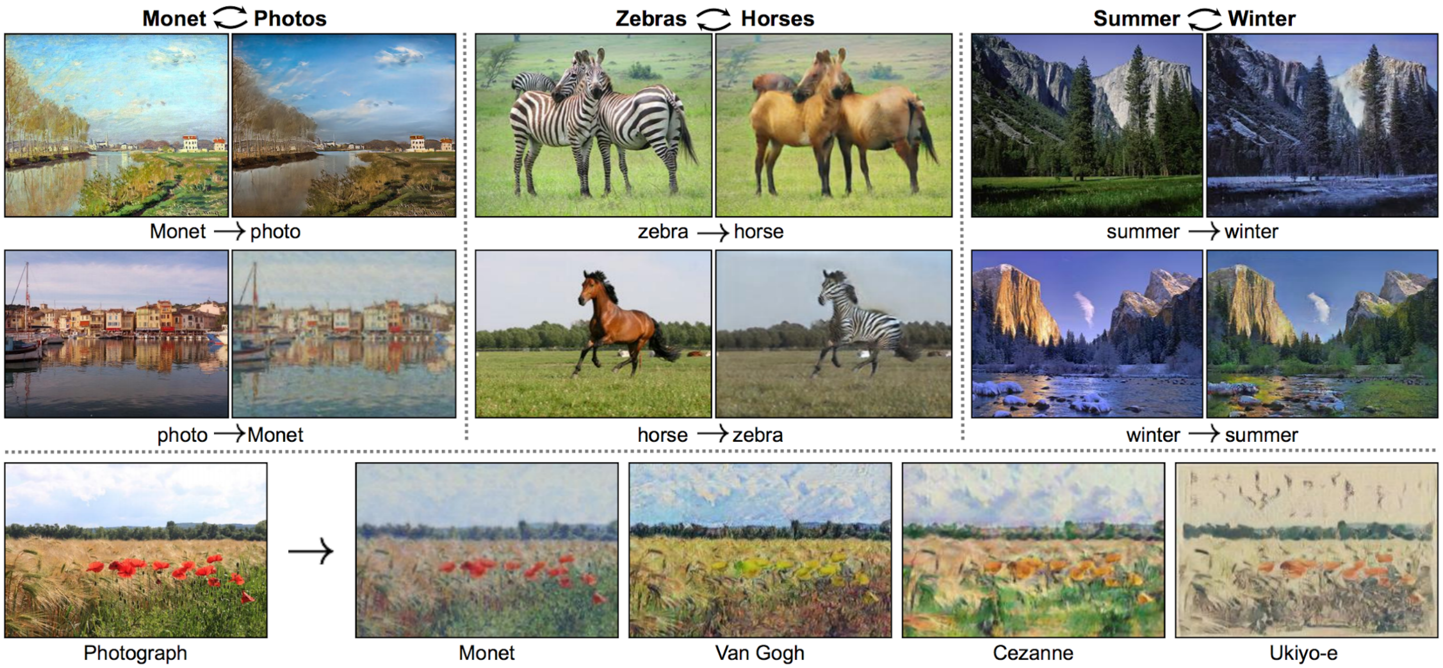


Figure source: Unpaired image-to-image translation using cycle-consistent adversarial networks (Cycle GAN), Zhu et al., ICCV 2017.

Transfer Learning

- Source data Ds = (Xs, Ys)
- Target data Dt = (Xt, Yt)
- Source and target data distributions may be different
- Target data labels may not be available

Transfer Learning

Tasks / Distributions	**Same source and target distributions on X**	**Different source and target distributions on X**
Same tasks on source and target domains	Supervised learning	Transductive transfer learning = **domain adaptation**
Different tasks on source and target domains	Inductive transfer learning	Unsupervised transfer learning

Transfer Learning

Tasks / Distributions	**Data collected from the same user**	**Data collected from different users**
Detect spam	Supervised learning	Transductive transfer learning = **domain adaptation**
Detect spam vs. detect hoax	Inductive transfer learning	Unsupervised transfer learning

Transfer Learning

Tasks / Distributions	**P(Xs) = P(Xt)**	**P(Xs) != P(Xt)**
Ts = Tt	Supervised learning	Transductive transfer learning = **domain adaptation**
Ts != Tt	Inductive transfer learning	Unsupervised transfer learning

Domain Adaptation

- Source and target tasks are the same Ts = Tt
- Source dataset with many labeled examples
- Target dataset with few or no labeled examples

Training data

- Supervised: available labeled data
- Semi-supervised: uses both labeled and unlabeled data
- Unsupervised: only unlabeled data

Domain Adaptation

- Supervised: labeled source and labeled target data
- Unsupervised: labeled source and unlabeled target data

Invariance

- Most learning tasks are invariant to sets of transformations
- Classification is invariant to translation, rotation, reflection,..
- $y = f(t(X)) = f(X)$
- Class does not change when transforming the input by t

Invariance

- Data augmentation: train on larger dataset

- Work with unlabeled data:
  Pretext: generate classes by transformations
  Supervised training

Equivariance

- Function commutes with transformation: $f(t(x)) = t(f(x))$
- For example, edge detection is equivariant to translation
- Translation of input image translates the output in exactly the same way

Transfer Learning Example

- Learn policy using reinforcement learning to balance small pendulum
- Transfer to large pendulum
- Option 1: Learn policy using reinforcement learning to balance large pendulum
- Option 2: Transfer learning
- Q: What is the common information or shared structure between the tasks?
- A: in this example, the ODE that models the pendulum

Transfer Learning

- Use same representation for tasks
- What changes between tasks?
- Set of transformations t that transform one task to another
- Related tasks can be transformed from one to another using a specific set of transformations
- Equivalence class t~
- Best approximators m_{t1} and m_{t2} related in the same way as t_1 and t_2
- Equivariance

Domain Adaptation

Massachusetts Institute of Technology

Adversarial Unsupervised Domain Adaptation

- Train GAN generator from source to target
- Train classifier on mapped source and source labels
- Apply classifier to target

Adversarial Unsupervised Domain Adaptation

- Ds = (Xs, Ys) for example simulated data
- Dt = (Xt, ?) for example real data
- Train GAN generator from source Xs to target Xt
  - Xt = g(Xs)
- Train GAN discriminator d(g(Xs), Xt)
- Train classifier on (g(Xs), Ys)
- Apply classifier on Xt

SimGAN

- Train GAN generator from synthetic to real images
- Train classifier on mapped synthetic and synthetic labels
- Apply classifier to real images

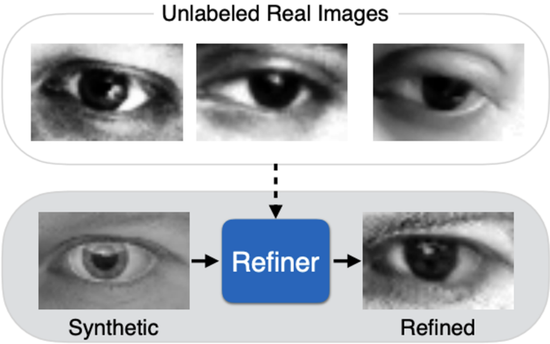


Figure source: Learning from simulated and unsupervised images through adversarial training, Shrivastava et al, CVPR 2017

Massachusetts Institute of Technology

SeUDA

- Ds = (Xs, Ys) for example simulated data
- Dt = (Xt, ?) for example real data
- Train GAN generator from target Xt to source Xs
  - Xs = g(Xt)
- Train classifier on (Xs, Ys)
- Apply generator to g(Xt) and classify source domain

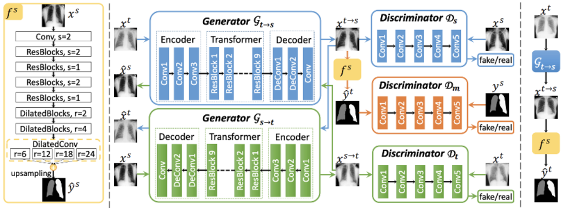


Figure source: Semantic-aware generative adversarial nets for unsupervised domain adaptation in chest X-ray segmentation, Cheng et al, 2018

ADDA

- $D_s = (X_s, Y_s)$, $Y_s = f_s(X_s)$
- $LA(D_s) = f_2(f_1(X_s))$
- Train f_1 CNN and f_2 classifier on D_s
- Train f'_1 CNN on X_t using discriminator $d(f_1(X_s), f'_1(X_t))$
- Apply $f_2(f'_1(X_t))$

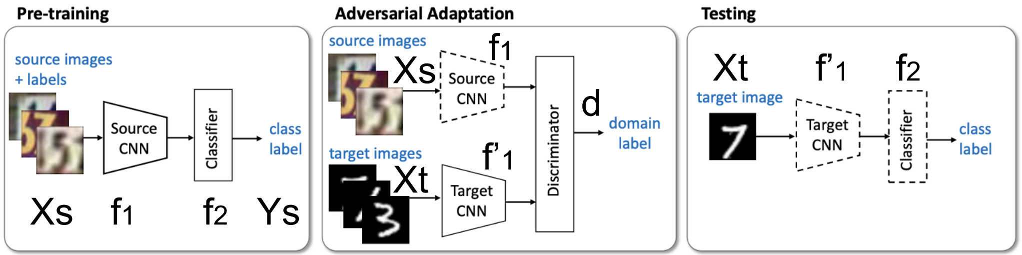
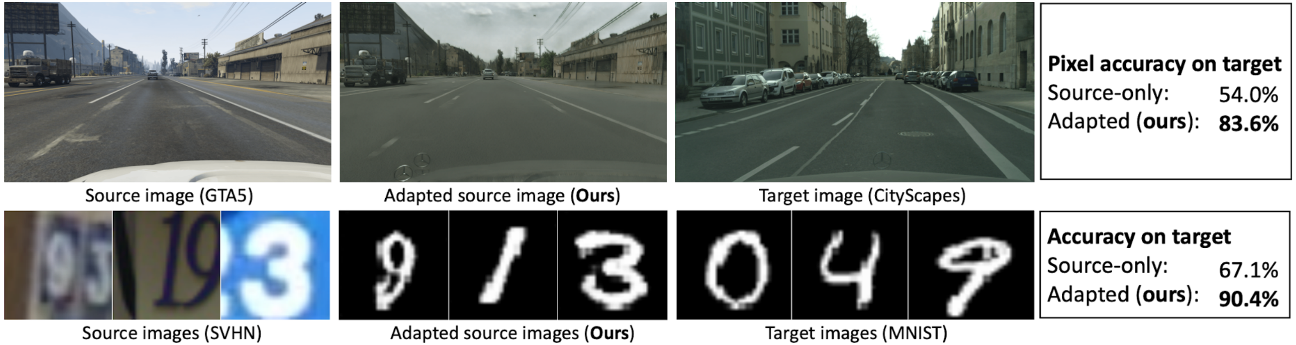
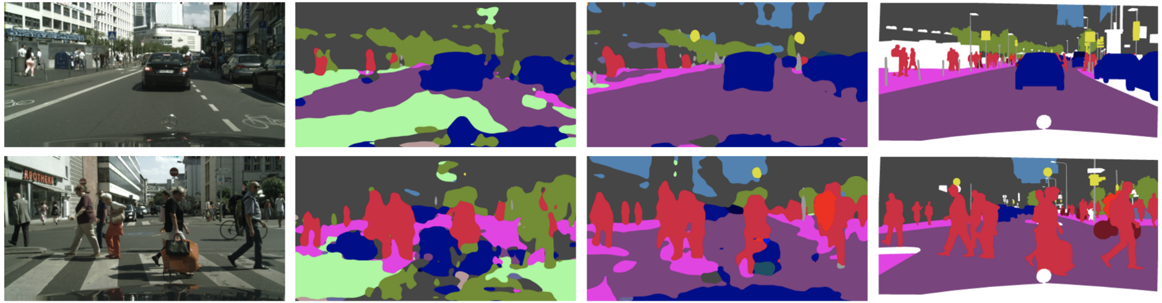


Figure source: Adversarial discriminative domain adaptation, Tzeng et al, CVPR 2017

Cycada

- CycleGAN





(a) Test Image (b) Source Prediction (c) CyCADA Prediction (d) Ground Truth

Figure source: CyCADA: Cycle-Consistent Adversarial Domain Adaptation, Hoffman et al, 2018

Meta Learning

MIT

Iddo Drori, Fall 2020

Massachusetts Institute of Technology