Visual AI

CPSC 532R/533R – 2019/2020 Term 2

Lecture 11. Attention models

Helge Rhodin

Assignment 3

- Rendering
- Learning shape spaces
- Interpolating in shape spaces

- Due today

Assignment 3: Neural Rendering and Shape Processing

CPSC 532R/533R Visual AI
by Helge Rhodin and Yuchi Zhang

This assignment is on neural rendering and shape processing—computer graphics. We provide you with a dataset of 2D icons and corresponding vector graphics as shown in Figure 1. It stems from a line of work on translating low-resolution icons to visually appealing vector forms and was kindly provided by Sheffer et al. [1] for the purpose of this assignment.

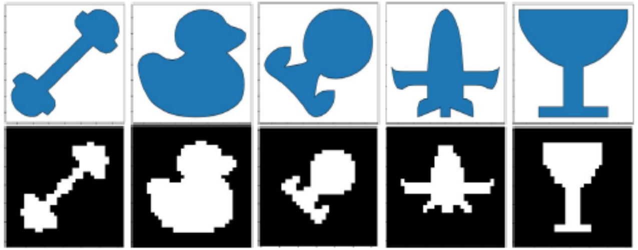


Figure 1: Icon vector graphics and their bitmap representation.

The overall goal of this assignment is to find transformation between icons. We provide the `ImagerIcon` dataset as an HDF5 file. As usual, the `Assignment3_TaskI.ipynb` notebook provides dataloading, training and validation splits, as well as display and training functionality. Compatibility of the developed neural networks with color images is ensured by storing the contained 32 x 32 icon bitmaps as $3 \times W \times H$ tensors. Vector graphics are represented as polygons with $N = 96$ vertices and are stored as $2 \times N$ tensors, with neighboring points stored sequentially. The polygon representation with a fixed number of vertices was attained by subsampling the originally curved vector graphics.

Recap: GAN training

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

---

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

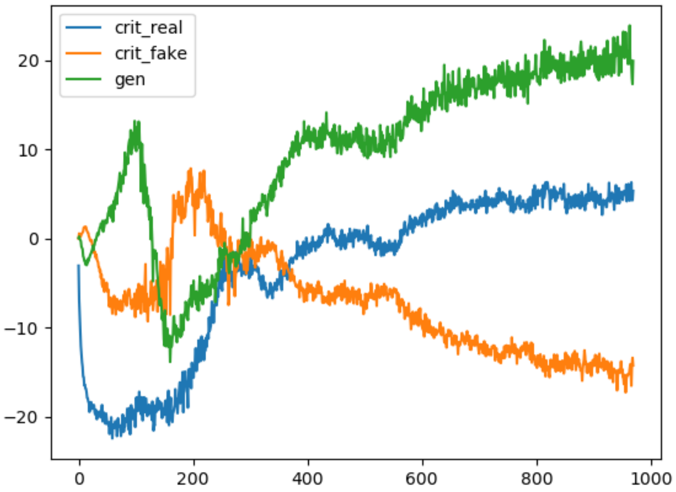
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Green: outer loop on generator (gradient descent)

Orange: inner loop on discriminator (gradient ascent)



Chaotic GAN loss behavior
(e.g., generator loss going up not down)

[Goodfellow et al., Generative Adversarial Networks. 2014]

Recap: Wasserstein GAN

Diverse measures exist to compare probability distributions (here generated and real image distribution)

- The *Total Variation* (TV) distance

$$\delta(\mathbb{P}_r, \mathbb{P}_g) = \sup_{A \in \Sigma} |\mathbb{P}_r(A) - \mathbb{P}_g(A)| \ .$$

- The *Kullback-Leibler* (KL) divergence

$$KL(\mathbb{P}_r \| \mathbb{P}_g) = \int \log \left(\frac{P_r(x)}{P_g(x)} \right) P_r(x) d\mu(x) \ ,$$

- The *Jensen-Shannon* (JS) divergence

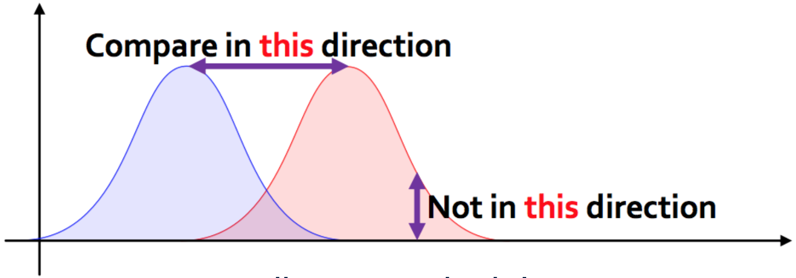
$$JS(\mathbb{P}_r, \mathbb{P}_g) = KL(\mathbb{P}_r \| \mathbb{P}_m) + KL(\mathbb{P}_g \| \mathbb{P}_m) \ ,$$

where $\mathbb{P}_m = (\mathbb{P}_r + \mathbb{P}_g)/2$

JS is what the classical GAN optimizes

- The *Earth-Mover* (EM) distance or Wasserstein-1

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y) \sim \gamma} \big[\, \|x - y\| \, \big] \ ,$$



EM distance principle

[Arjovsky et al., Wasserstein GAN. 2017]

Recap: Comparison: VAE and GAN

	VAE	GAN
Objective	$\min_{\theta,\phi} -\mathbf{E}_{\mathbf{h}\sim q_\phi(\mathbf{h}\|\mathbf{x})}\big(\log p_\theta(\mathbf{x}\|\mathbf{h})\big) + D_{\mathrm{KL}}(q_\phi(\mathbf{h}\|\mathbf{x})\\|p(\mathbf{h}))$	$\min_G \max_D [E_{x\sim p_r}[\log D(x)] + E_{z\sim p_z}[\log(1 - D(G(z)))]]$

Sampling a ‘natural’ image

VAE:

- Draw a random sample from a Gaussian

$$\mathbf{h} \sim \mathcal{N}(0,1)$$

- Apply the decoder on h

GAN:

- Draw a random sample from a Gaussian

$$z \sim \mathcal{N}(0,1)$$

- Apply the generator on z

Computing the probability of a given image x

VAE:

- Apply the encoder on x

$$\mathbf{h} = e_\theta(\mathbf{x})$$

- Evaluate the prior on h

$$\mathcal{N}(\mathbf{h}|0,1)$$

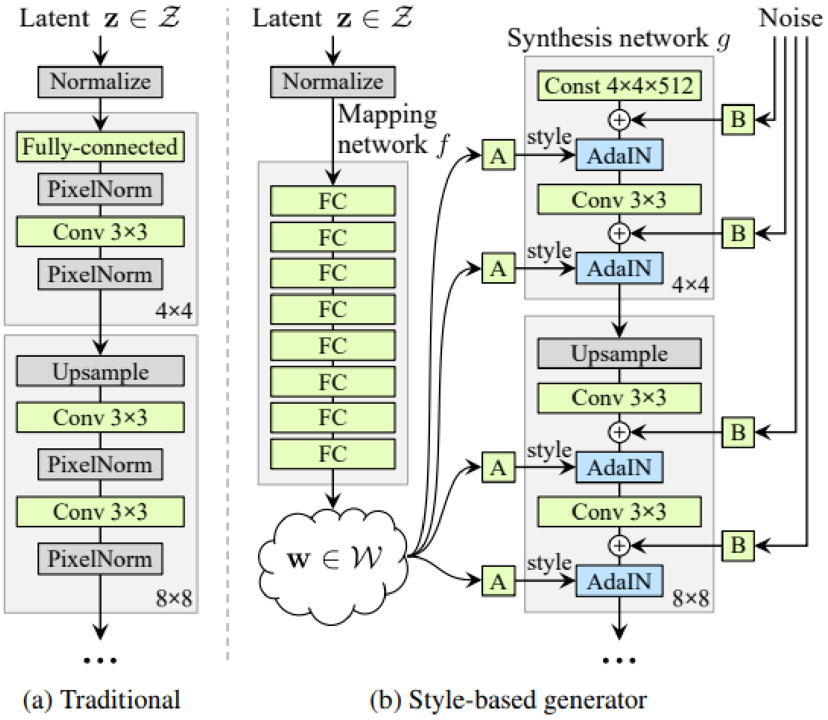
- thanks to explicit density model

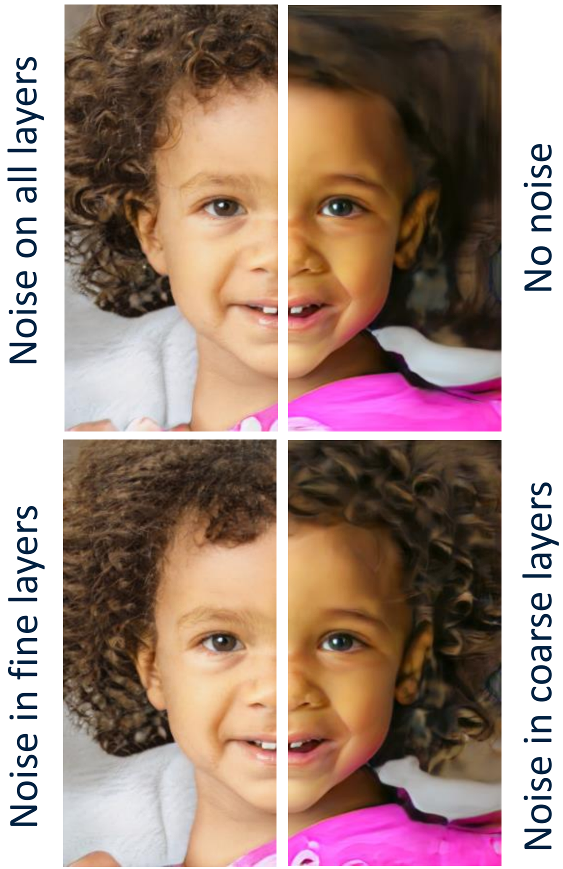
GAN:

- Not applicable!
  - it models an implicit density

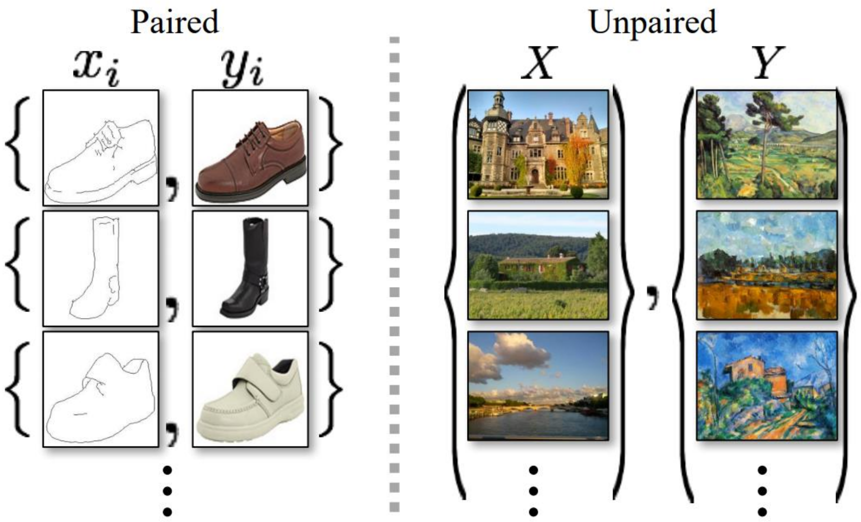
Recap: Style GAN internals

- Compute style description given noise (form of non-Gaussian noise)
- Apply style and add noise at all layers (of ProgGAN generator)



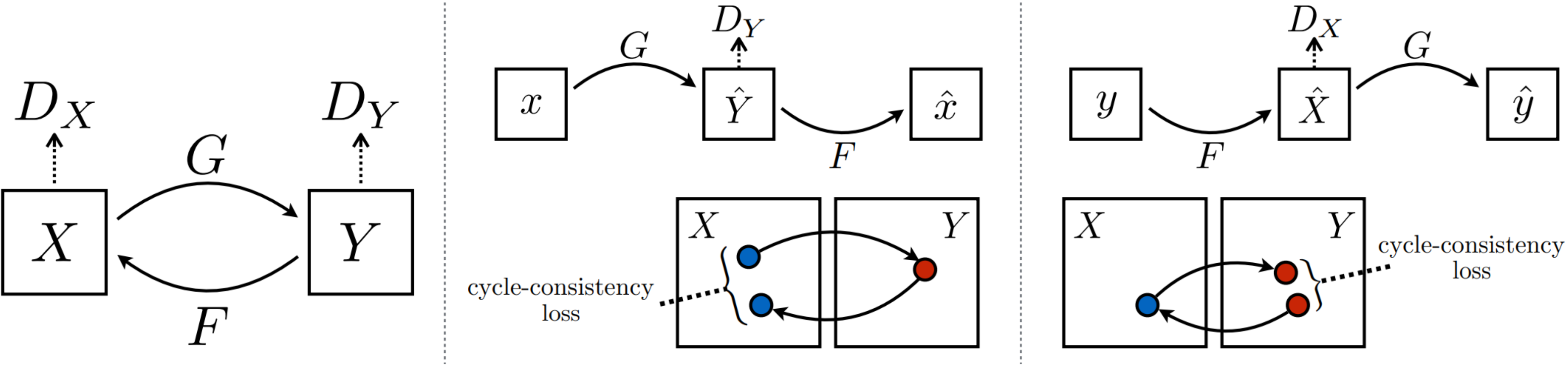


Recap: Paired vs. unpaired image translation



Recap: Cycle GAN principle

Construct an identity function by chaining two translation networks



- Jointly learn to
  - map from X to Y and back to X
  - map from Y to X and back to Y

Canonical solutions?

Attention mechanisms, preliminaries

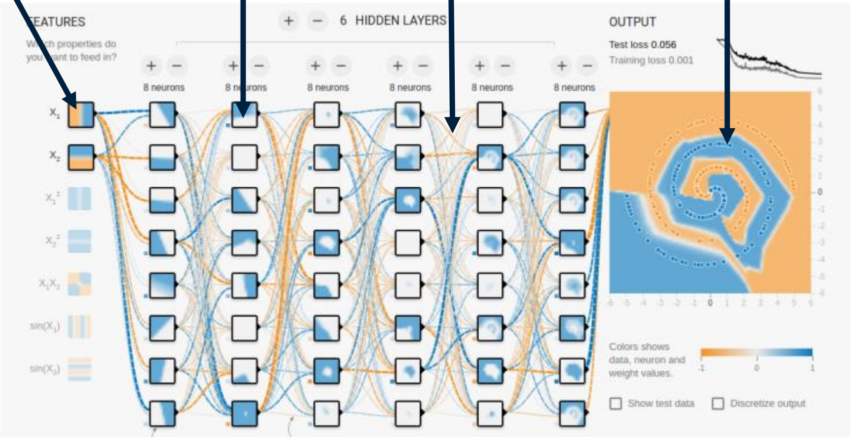
Image reconstruction from NN activations

Neural network training

- given architecture, objective and dataset
- optimize the weights to explain the data

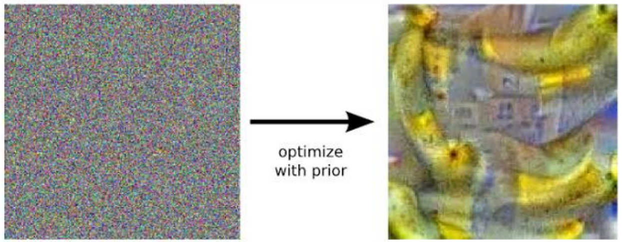
Image reconstruction

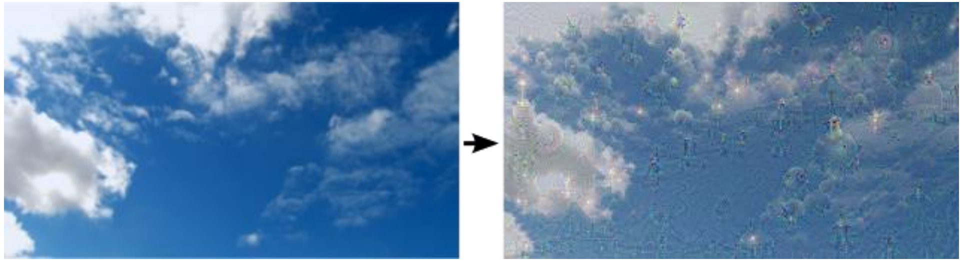
- given target features/activations at a layer (e.g., elephant class = true)
- optimize the input to yield the target feature
  - starting from noise or
  - starting from an example image
    - constrain solution to be close to the example image and target feature

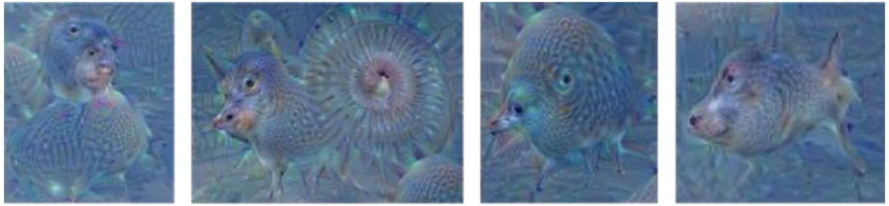


Multilayer perceptron (fully connected network)
live at playground.tensorflow.org

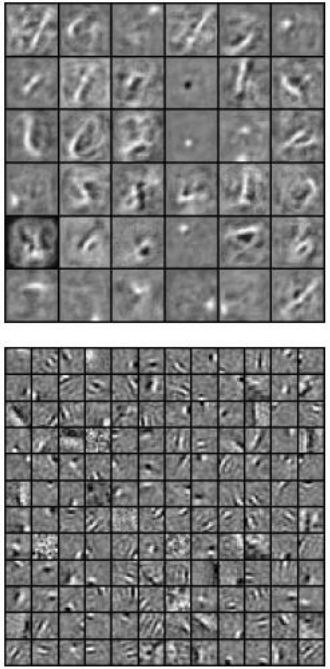
DeepDream (Google)





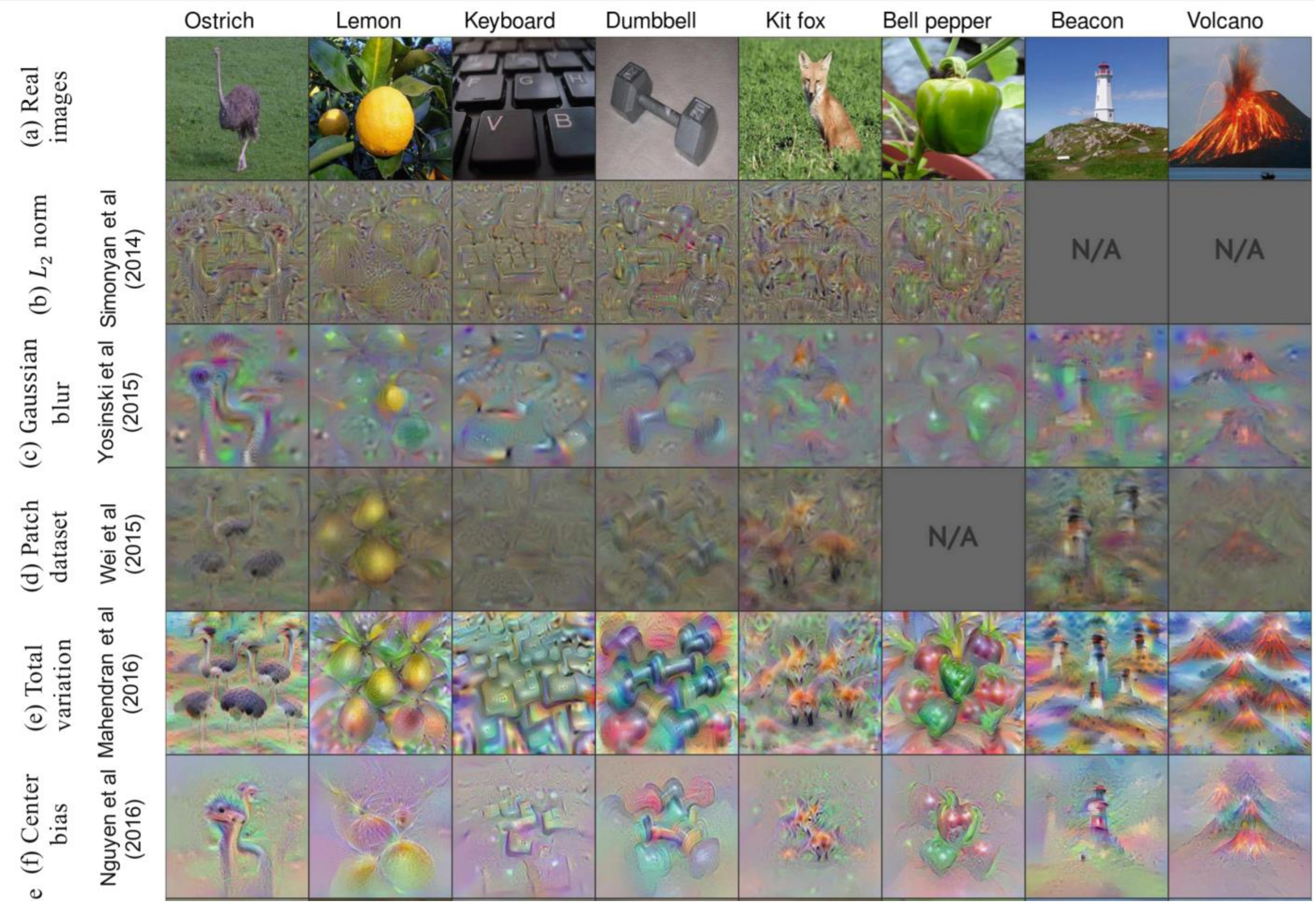


"Admiral Dog!" "The Pig-Snail" "The Camel-Bird" "The Dog-Fish"



[Erhan et al., Visualizing Higher-Layer Features of a Deep Network. 2009]

More dreams

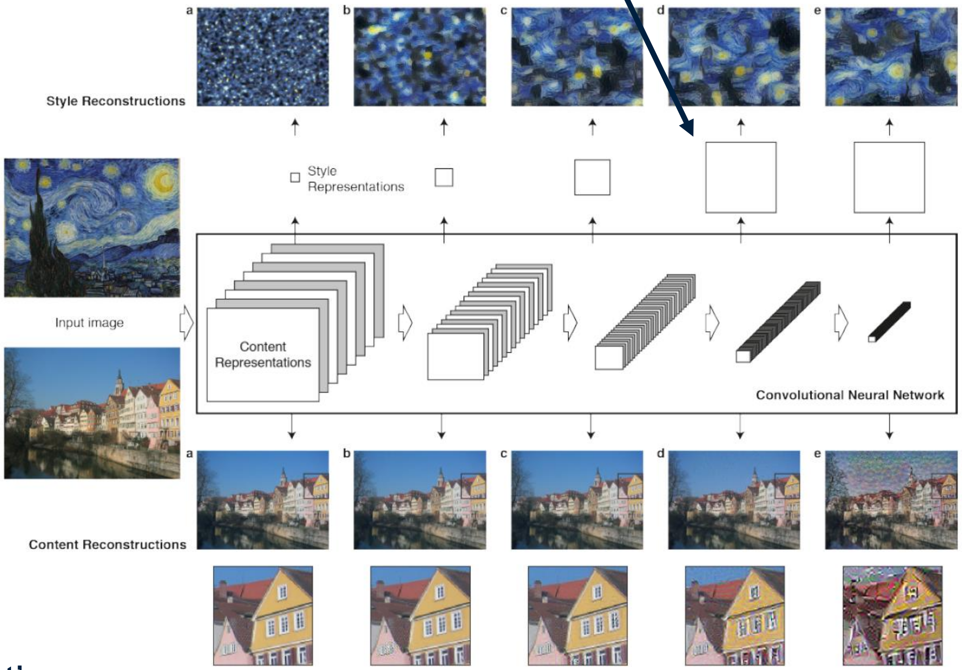

Ostrich
Lemon
Keyboard
Dumbbell
Kit fox
Bell pepper
Beacon
Volcano
(a) Real images
(b) L_2 norm
Simonyan et al (2014)
(c) Gaussian blur
Yosinski et al (2015)
(d) Patch dataset
Wei et al (2015)
(e) Total variation
Mahendran et al (2016)
(f) Center bias
Nguyen et al (2016)
N/A
N/A
N/A

Recap: Style transfer

Style influence grows with depth

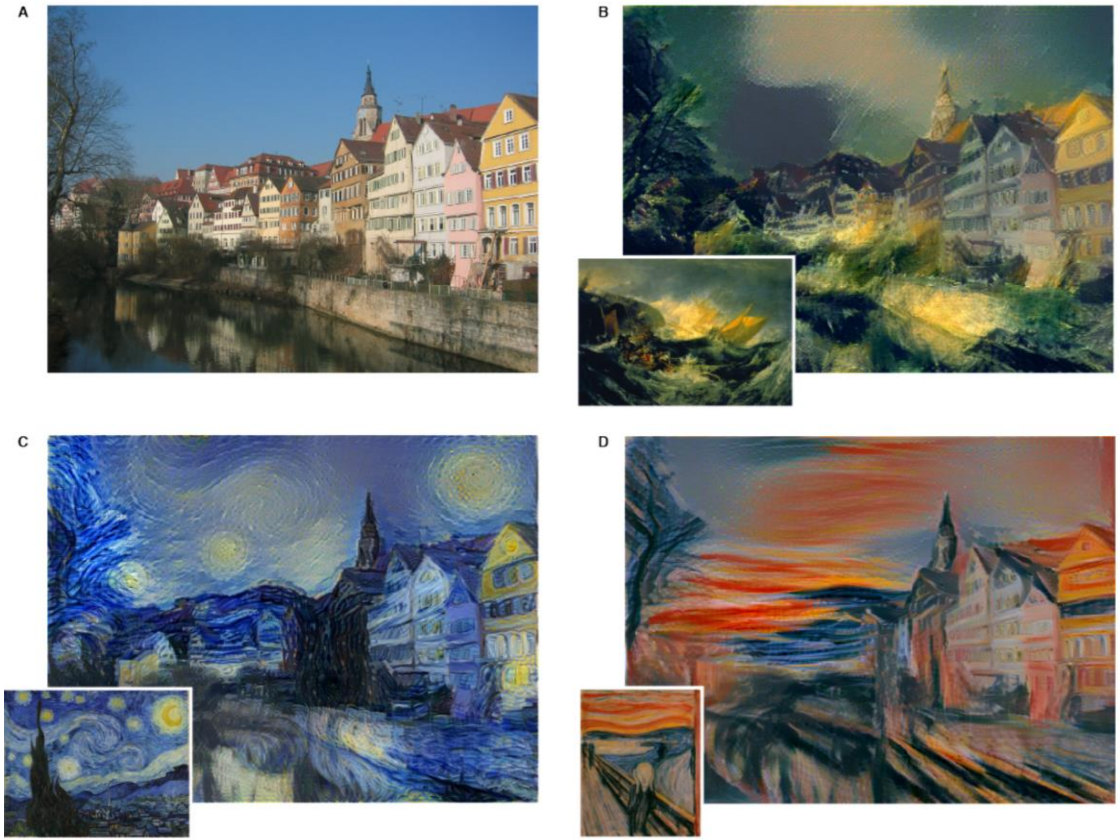
Idea: ‘turn NN training on its head’

- apply gradient descent
  - with respect to the ‘input’ image (instead of NN weights)
  - keep the neural network weights fixed
- find neural network features that
  1. capture style (averaged spatially)
     - correlation between features of a layer
  2. capture content
     - l2 difference between features of a layer
- set the objective function as the distance of ‘input’
  - to style target (painting), in terms of style features
  - to content target (photo), in terms of content features



[Gatys et al., A Neural Algorithm of Artistic Style 2015]

Style transfer results

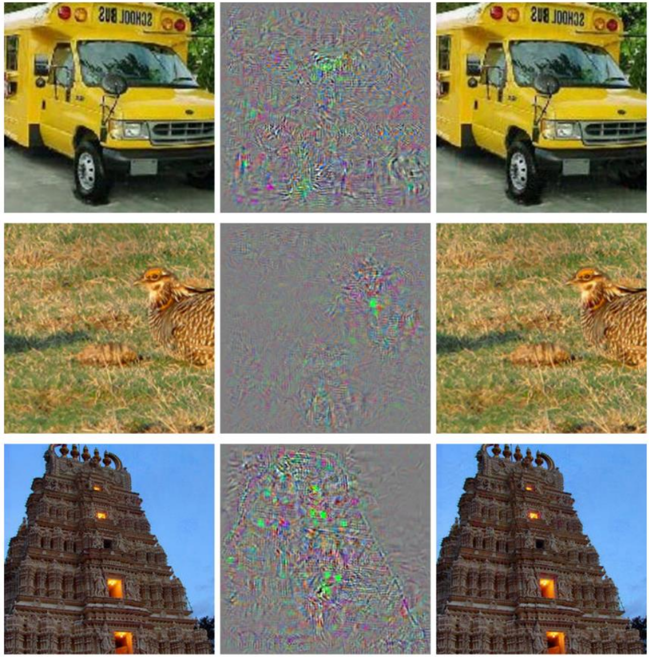


Adversarial examples

How to fool a classifier

- Goal:
  - an image that is lose to the original
  - yields the wrong output,
    - on the right ‘ostrich, Struthio camelus’
- Solution:
  - gradient descent on the colors of the input image

- New branch of research:
  - how to protect from adversarial examples?

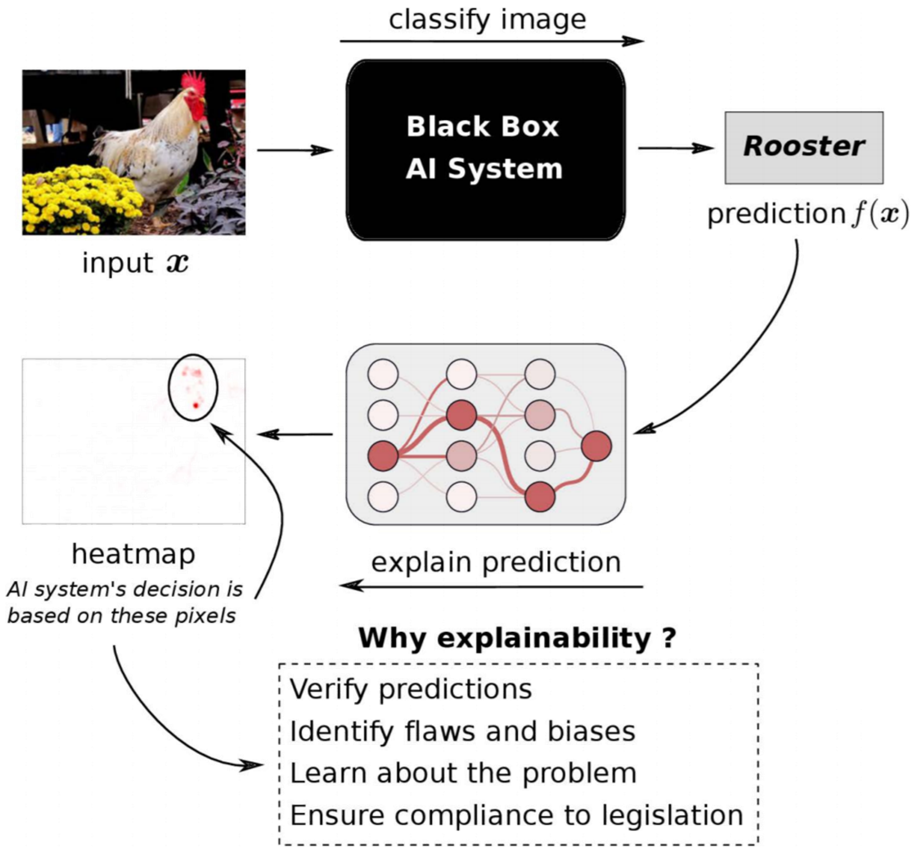


original | noise (10x amplified) | adversarial example

[Szegedy et al. Intriguing properties of neural networks, 2013]

Explaining predictions

- Some form of tracing back the NN computations
- Measuring the contribution of each input pixel to the final outcome
- A heatmap that measures importance



Interpretability of neural networks

Analyze the gradient of the objective with respect to the input pixels [Baehrens et al. 2010]

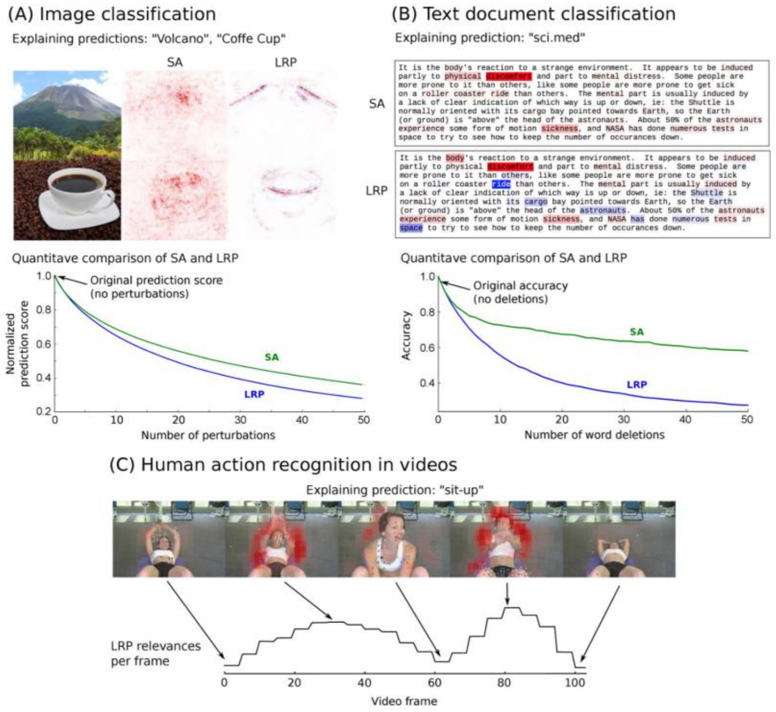
- a local linear approximation of the model’s behavior
- quite sensitive to noise

Applies to various different domains

- images, text, motion (videos)

Extensions:

- integrated gradients [Sundararajan et al. 2016]
- SmoothGrad [Smilkov et al. 2017]
- layer-wise relevance propagation (LRP) [Bach et al. 2015]

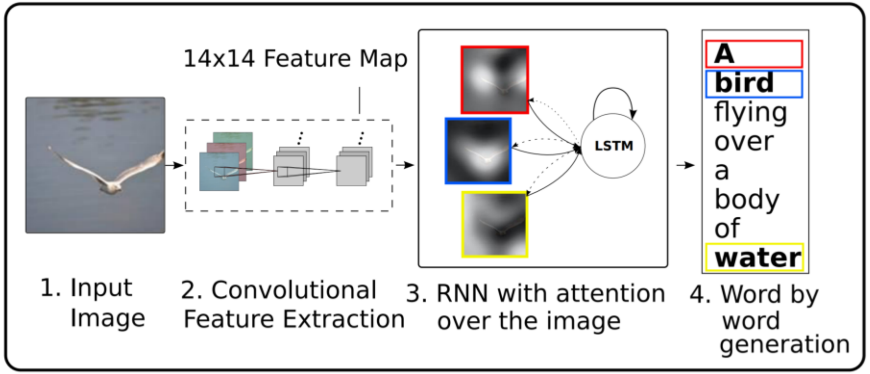


Attention mechanisms

Attention maps

- Spatially adaptive pooling of features
  - weighted average of feature map
  - weighted by attention window
  - a recursive network can look at multiple image parts
    - inspired by human attention

- Provides an interpretable representation
  - spatially localized



[Xu et al, Show, Attend and Tell: Neural Image Caption Generation with Visual Attention]

Constrained attention maps

Idea: use a pre-defined window function

- Gaussian window
  - smooth
  - infinite support
  - exponential falloff
  - simple to compute

- Other possible functions?
  - bump functions
    - smooth
    - **finite, compact support**
    - exponential falloff
    - simple to compute
  - box function?

[Show, Attend and Tell: Neural Image Caption Generation with Visual Attention]

Hard attention windows

Cropping a subset of pixels

- g = I[y:y+h, x:x+w]
- efficient (the subsequent network only looks at a smaller part)
- non-differentiable

RoI pooling

- compute a crop of fixed resolution
- part the crop window into a fixed number of bins
  - e.g., 7x7 bins
- distribute pixels to bins
  - round for those on the boundary between bins (nearest bin)
- average or max pool within each bin

http://akosiorek.github.io/ml/2017/10/14/visual-attention.html

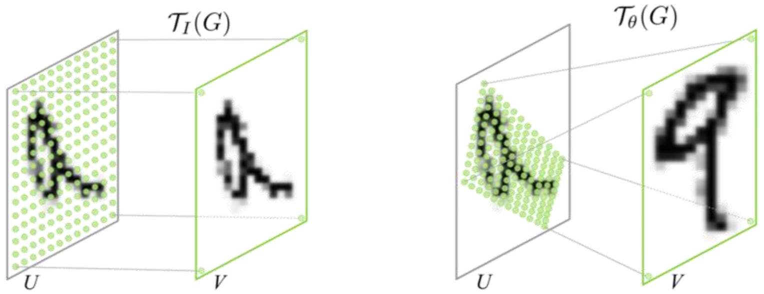
Spatial transformer networks (STN)

Definition: A neural network layer that

- subsamples the original image
  - e.g., cropping with sub-pixel accuracy
- parameterized by the grid of target pixels
- using bilinear interpolation for each grid point

The grid is usually defined by a parametric function

- is itself an other network layer
- rigid transforms (translation, rotation scaling)
  - most common
- thin-plate spline
  - a non-linear deformation
- as the integral of velocities
- …



Sampling according to arbitrary grid

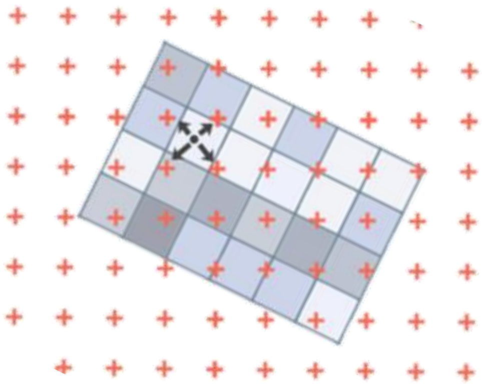
[Jaderberg et al., Spatial Transformer Networks, 2015]

STN summary

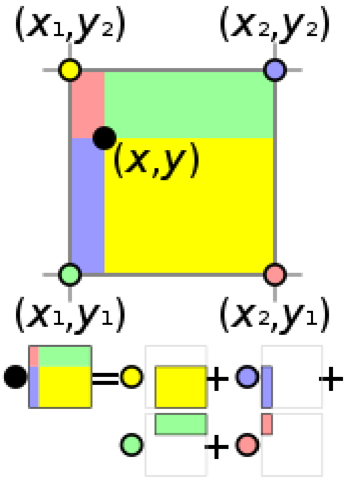
The STN consists of

1. grid generation
   - parametric
     - differentiable
2. grid sampling
   - bilinear interpolation
     - differentiable

- still efficient
  (compared to non-differentiable cropping and soft windows)
- moderate smoothness guarantees
  (piecewise linear)



grid (dagrayrk) on
image grid (red)

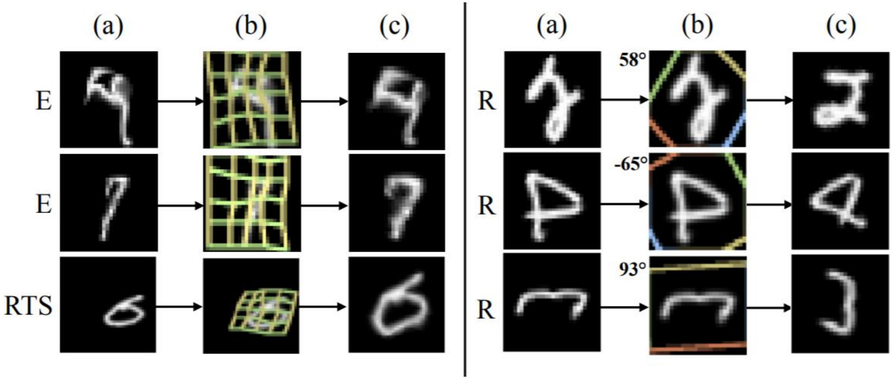


Bilinear
interpolation

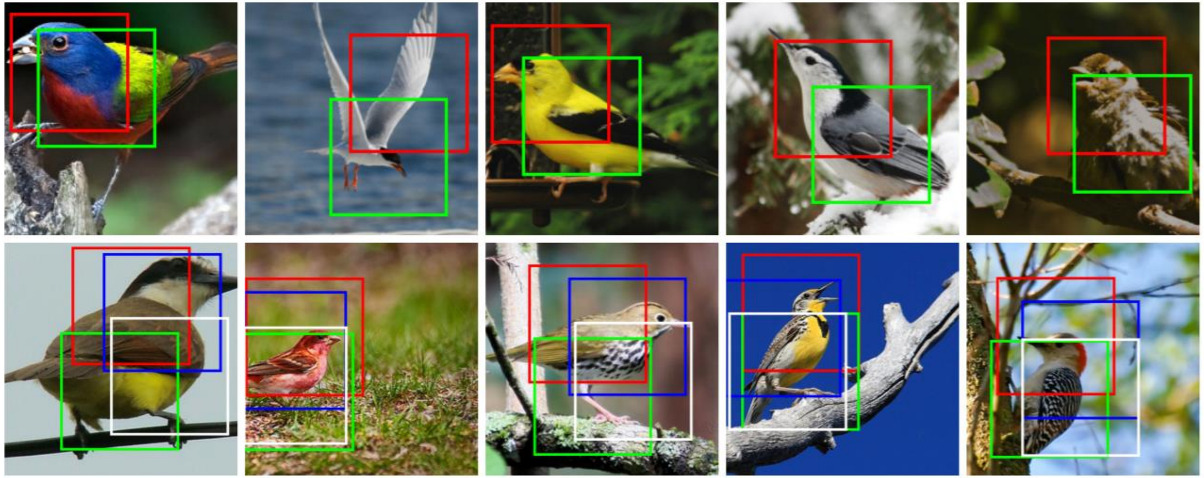
STN results

Advantages

- zoom into image (normalize scale)
- can rotate (normalize orientation)
- undo other deformations
- -> higher accuracy



Model		
Cimpoi '15 [5]		66.7
Zhang '14 [40]		74.9
Branson '14 [3]		75.7
Lin '15 [23]		80.9
Simon '15 [30]		81.0
CNN (ours)	224px	82.3
2×ST-CNN	224px	83.1
2×ST-CNN	448px	83.9
4×ST-CNN	448px	**84.1**



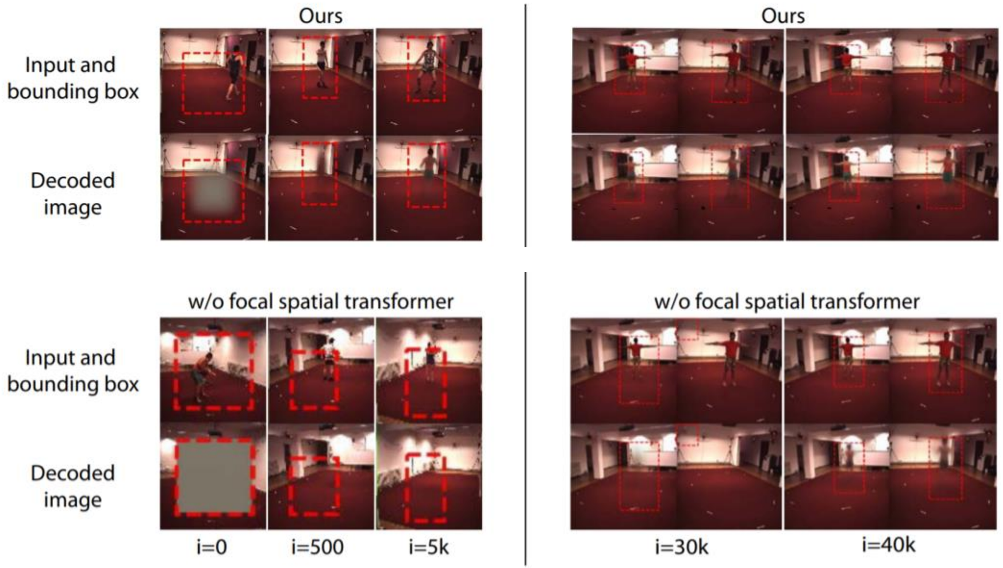
Focal spatial transformer = smooth window + hard window

A combination of smooth and hard windowing

- scale normalization by hard window
- smoothness by soft window
- only a small computational overhead
  - multiplication

Not quite sure why and when it helps

- under investigation (by Willis)
- in this work, the spatial transformer is used twice at encoding and decoding time
  - note, STNs can go from low to high and high to low resolution



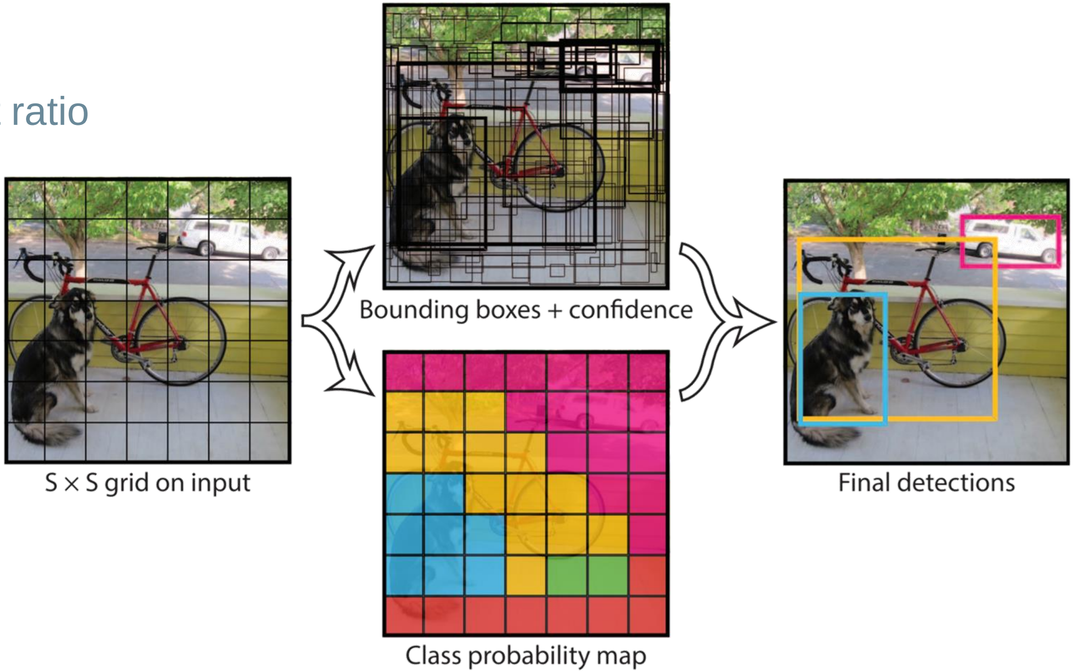
[Rhodin et al. Neural Scene Decomposition, CVPR'19]

Applications

You only look once (YOLO): Real-Time Object Detection

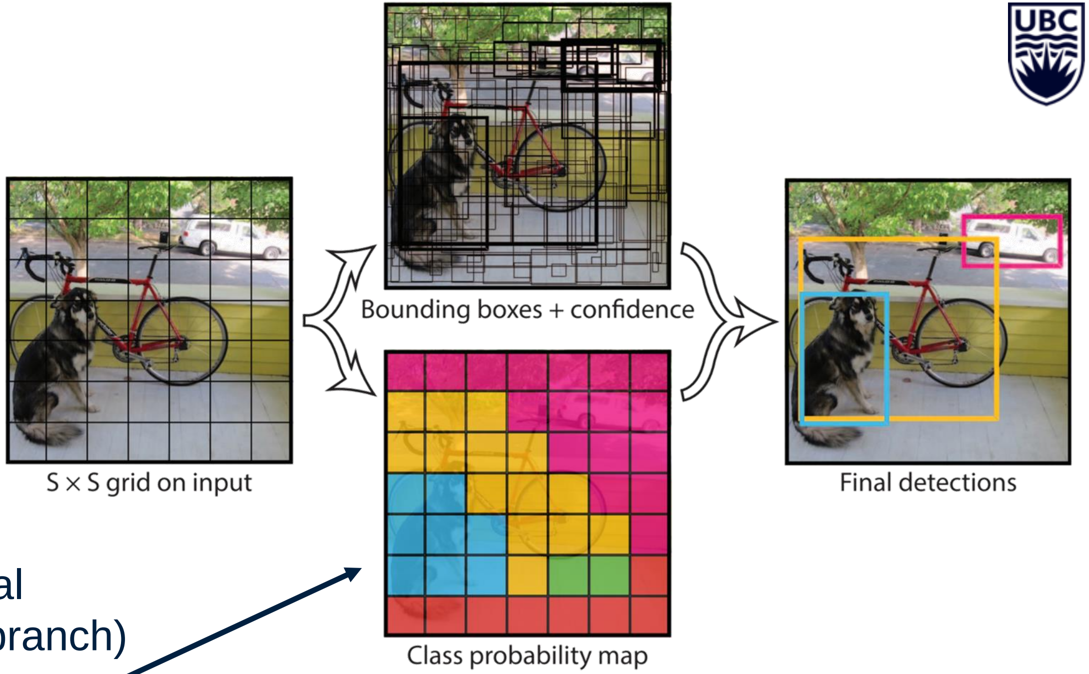
Goal: A joint model for object detection and classification

- a clever network architecture
  - pure feed-forward, fast by design
  - high accuracy
  - varying bounding box size and aspect ratio
- an efficient implementation
  - ‘dark-net’ framework
  - inspired by GoogLeNet
  - custom C/CUDA implementation
    - all layers hand coded
    - all derivatives hand coded!

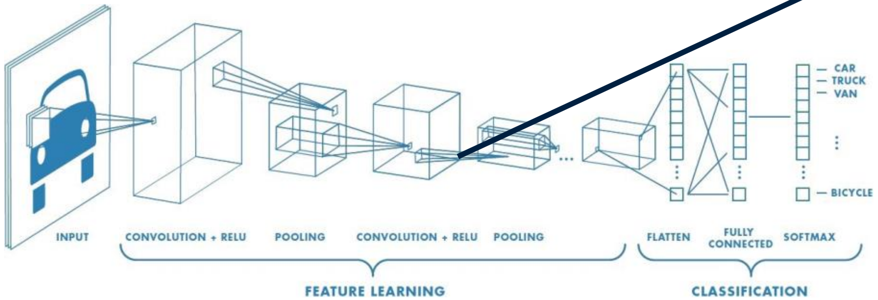


[Redmon et al. YOLO: Real-Time Object Detection 2016]

Yolo Details



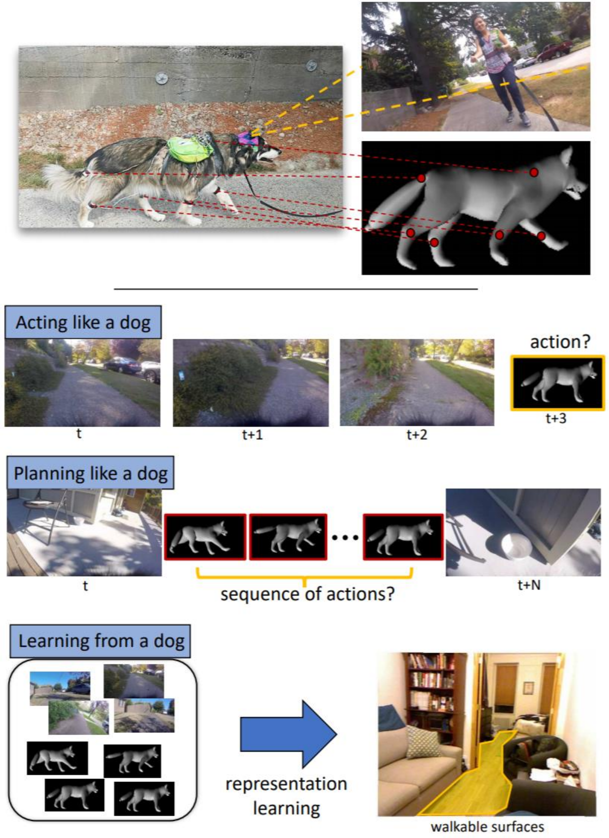
Fully convolutional
(without classification branch)



Unrelated but funny:
Who Let The Dogs Out? Modeling Dog Behavior From Visual Data

By the YOLO author, Joseph Redmon

- Learning visual features and behavior by observing an egocentric dog camera



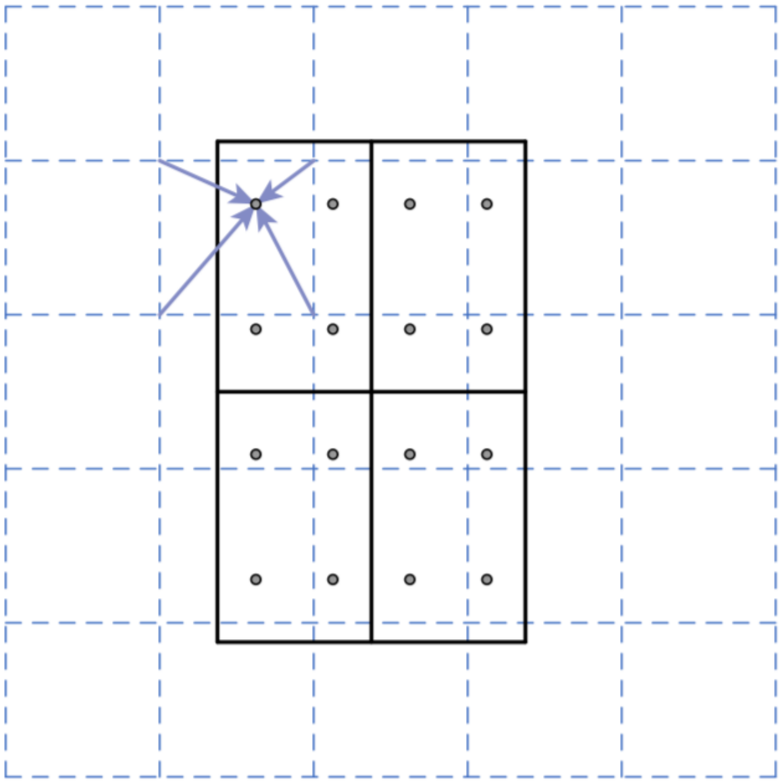
RoIAlign pooling

Goal: attain a fixed-size localized feature map

- compute grid points for target bins
- four locations in each RoI bin
- bilinear interpolation for each sample
- average or max polling within each bin

It is a variant of STNs

- differentiable with respect to position



Mask R-CNN

A joint model for

- object detection
- instance segmentation
- extending Region-based CNN (R-CNN)

Advantages

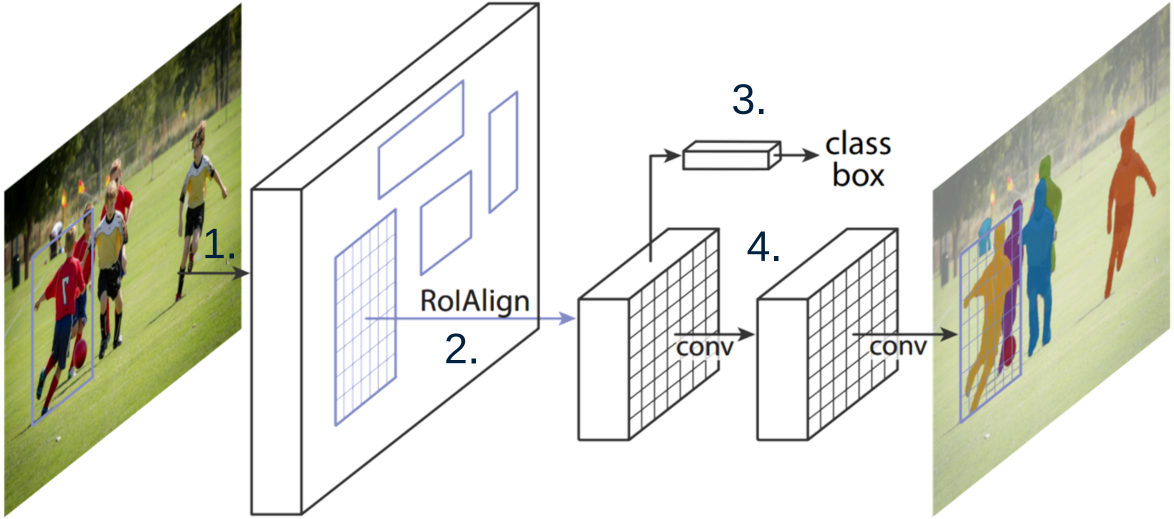
- fast
- accurate
- simple

[He et al, Mask R-CNN]

Mask R-CNN details

A multi-stage process

1. backbone network to extract feature maps
2. RoIAlign pooling per object candidate
3. separate classification branch
4. instance segmentation
   (one channel per class)



Perspective spatial transformer

Goal: self-supervised training of reconstruction

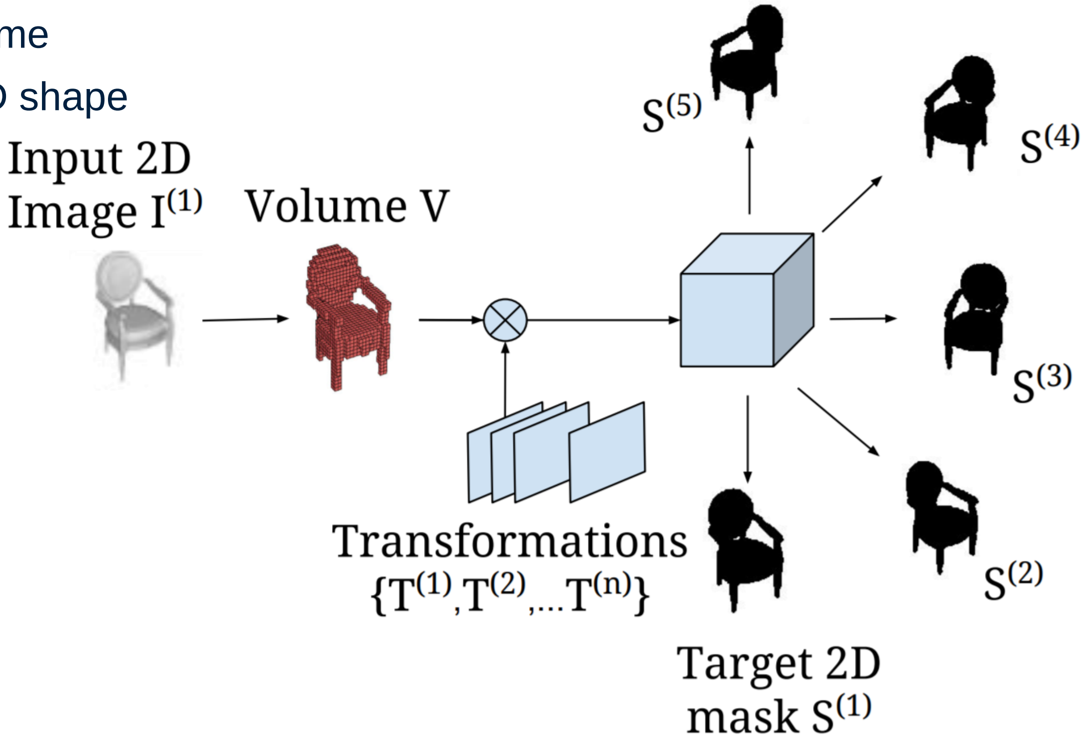
Given: set of multi-view images at training time

Training: a neural network that predicts a 3D shape

- consistent with all views
- using silhouette constraints

Requires:

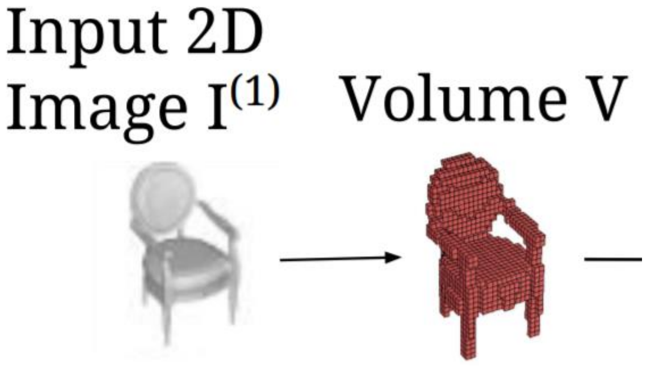
- 2D to 3D correspondences
- a perspective 3D spatial transformer

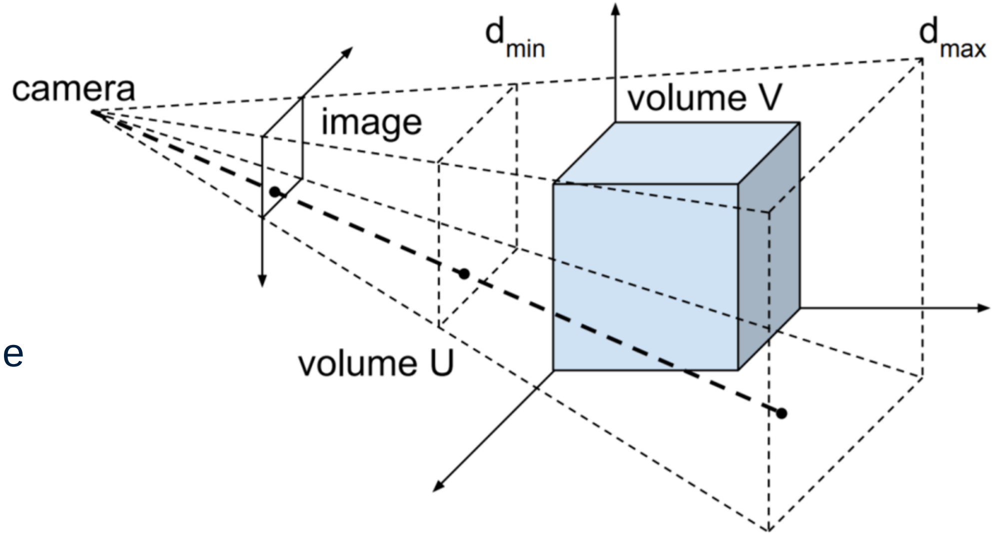


Perspective spatial transformer, details

Concept:

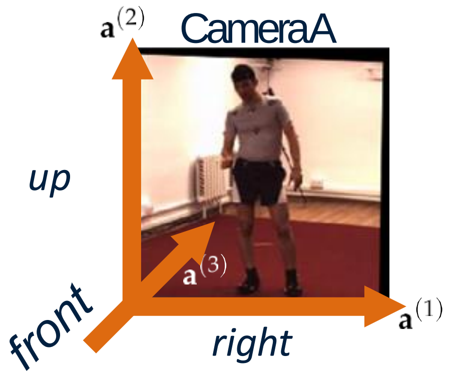
- predict a 3D occupancy grid given the input view
- construct a N 3D grids (one for each reference view)
  - pyramidal form, with
    - position and orientation of reference cameras
  - models the perspective effect
- sample the 3D volume
  - as for 2D spatial transformers, but
    by trilinear interpolation
- take the maximum along the depth direction
  - models projection
- minimize the distance of this projection to the reference image silhouette (see prev. slide)

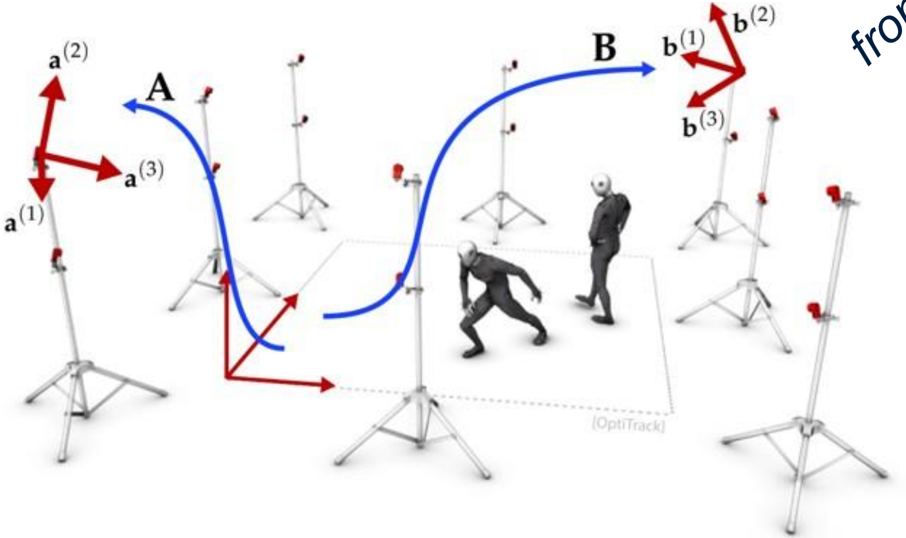




Recap: Linear transformations

- Used in computer graphics and computer vision
- A chain of linear maps is a linear map
  - rotation
  - scaling
  - shear and mirror



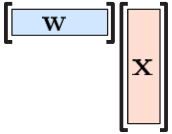


Recap: Affine transformations & augmented matrix and vector

- Can express rigid transformations
  - **Translation**
  - Scale
  - Rotation
  - Shear and mirror

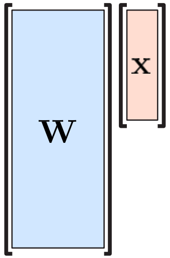
Linear

$$f(\mathbf{x}) = \sum_i \mathbf{w}_i \mathbf{x}_i$$
$$= \mathbf{w} \cdot \mathbf{x}$$



Multidimensional

$$f(\mathbf{x}) = \mathbf{W}\mathbf{x}$$



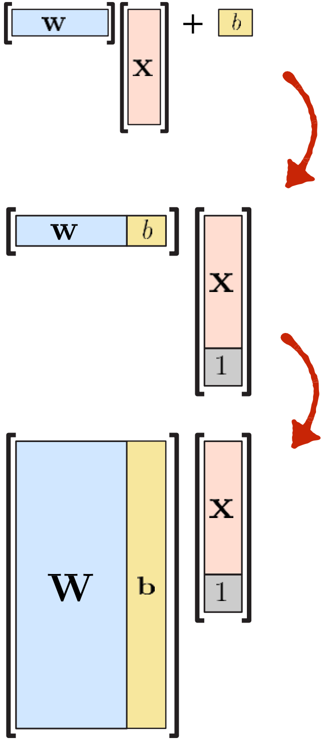
$$f(\mathbf{x}) = \sum_i \mathbf{w}_i \mathbf{x}_i + b$$
$$= \mathbf{w} \cdot \mathbf{x} + b$$
$$= \tilde{\mathbf{w}} \cdot \tilde{\mathbf{x}}$$

$$\text{with } \tilde{\mathbf{w}} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n, b)$$
$$\text{and } \tilde{\mathbf{x}} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, 1)$$

$$f(\mathbf{x}) = \tilde{\mathbf{W}} \cdot \tilde{\mathbf{x}}$$

$$\text{with } \tilde{\mathbf{W}} = \begin{pmatrix} \mathbf{w}_{1,1} & \mathbf{w}_{1,2} & \cdots & \mathbf{w}_{1,n} & b_1 \\ \mathbf{w}_{2,1} & \mathbf{w}_{2,2} & \cdots & \mathbf{w}_{2,n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \end{pmatrix}$$

Affine

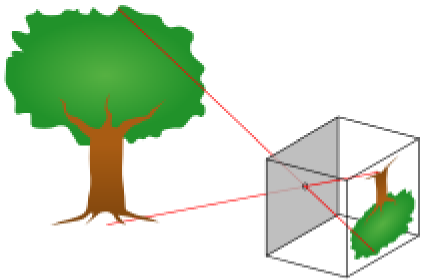


Recap: Projective transformation & Homogeneous coordinates

Equivalence in homogeneous coordinates

- Compared to the Euclidean space, points are not unique:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{m-1} \\ x_m \end{bmatrix} = \begin{bmatrix} x_1\lambda \\ x_2\lambda \\ \vdots \\ x_{m-1}\lambda \\ x_m\lambda \end{bmatrix} = \begin{bmatrix} x_1/x_m \\ x_2/x_m \\ \vdots \\ x_{m-1}/x_m \\ 1 \end{bmatrix}$$



Pinhole camera model

[https://en.wikipedia.org/wiki/Pinhole_camera_model]

- Able to model perspective transformations (projection) as a linear transformation

$$\begin{pmatrix} y_1 \\ y_2 \\ 1 \end{pmatrix} \sim \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix}$$

Projection in Homogeneous coordinates

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = -\frac{f}{x_3} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

Projection in Euclidean coordinates

Summary

- 11 Lectures (Weeks 1 – 6)
  - Introduction
  - Deep learning basics and best practices
  - Network architectures for image processing
  - Representing images and sparse 2D keypoints
  - Representing dense and 3D keypoints
  - Representing geometry and shape
  - Representation learning I (deterministic)
  - Representation learning II (probabilistic)
  - Sequential decision making
  - Unpaired image translation
  - Attention models

- 3x Assignments
  - Playing with pytorch (5% of points)
  - Pose estimation (10% of points)
  - Shape generation (10% of points)
- 1x Project (40 % of points)
  - Project pitch (3 min, week 6)
  - Project presentation (10 min, week 14)
  - Project report (8 pages, April 14)
- 1x Paper presentation (Weeks 8 – 13)
  - Presentation, once per student (25% of points) (20 min + 15 min discussion, week 8-13)
  - Read and review one out of the two papers presented per session (10% of points)

except on the day of your presentation, please submit your slides instead (PDF)

Course project proposal

Project proposal

- 3-minute pitch
  - answer three questions
    - what, why, how?
  - 2-3 slides should be enough
    - keep it high level
  - submit slides on canvas
- written proposal (one page, 11pt font)
  - research idea
  - possible algorithmic contributions
  - outline of the planned evaluation

W4	Jan 28	Representation learning I (deterministic) lecture slides - principal component analysis (PCA) - auto-encoder (AE) Homework 2 due. ~~Homework 3 release~~	PCA face model Deep Learning Book - Chapter 14
	Jan 30	Representation learning II (probabilistic) lecture slides - variational autoencoder (VAE) - generative adversarial network (GAN) Homework 3 release Assignment3.zip (posted Feb. 1)	Deep Learning Book - Chapter 20
W5	Feb 4	Sequential decision making - Monte Carlo methods - reinforcement learning	Deep Learning Book - Chapter 17
	Feb 6	Unpaired image translation - cycle consistency - style transfer ~~Homework 3 due~~	Cycle Gan Style transfer
W6	Feb 11	Attention models - spatial transformers, RoI pooling, attention maps - camera models and multi-view Homework 3 due (new deadline)	RoI pooling, Spatial Transformer Multi-view Geometry
	Feb 13	Project Pitches (3 min pitch) Project proposal due	
W7		Midterm Break (no class)	-
W8	Feb 25	Conditional content generation Park et al., Semantic Image Synthesis with Spatially-Adaptive Normalization paper Li et al., Putting Humans in a Scene: Learning Affordance in 3D Indoor Environments paper	
	Feb 27	Motion transfer Chan et al, Everybody Dance Now paper Gao et al., Automatic Unpaired Shape Deformation Transfer paper	

Course project schedule

- 14 projects
  - 16 persons teamed up
  - 5 single teams
- 0.5 minute setup
- 3 minute pitch
- 2 minutes comments
  - makes 77 minutes
- 3 minutes slack
- Submit PDF slides till Thursday, 7 am
  - on Canvas

#group	Time
a)	9:30
b)	9:35:30
c)	9:41
d)	9:46:30
e)	9:52
f)	9:57:30
g)	10:03
h)	10:08:30
i)	10:14
j)	10:19:30
k)	10:25
l)	10:30:30
m)	10:36
n)	10:41:30
o)	10:47

Course Project Signup 3	Michelle Appel	Full 2 / 2 students
Course Project Signup 4	Tianxin Tao	Full 2 / 2 students
Course Project Signup 5	Dingqing Yang	Full 2 / 2 students
Course Project Signup 15	Zikun Chen	1 / 2 students
Course Project Signup 16	MONA FADAVIARDAKANI	Full 2 / 2 students
Differentiable Shadow Rendering	Jerry Yin	Full 2 / 2 students
Improving Visual Quality of Unsupervise...	SHANE SIMS	Full 2 / 2 students
Killer Whale Identification	Matheus Ulhoa Avelar Stolet	Full 2 / 2 students
Knots Detection Based on Timber Board I...	Shenyi Pan	Full 2 / 2 students
Methods from Neuroevolution to improv...	EGE UNLU	1 / 2 students
Pose-Guided Visual Commonsense Reaso...	Zicong Fan	Full 2 / 2 students
Rethinking Visual Classifiers using the M...	Peyman Bateni	1 / 2 students
spatial embeddings to improve the gener...	Weidong Yin	1 / 2 students
Virtual Keyboard	Willis Peng	1 / 2 students

Done.