Neural Network Basics and Representation Learning for 3D Shapes

Richard (Hao) Zhang

CMPT 464/764: Geometric Modeling in Computer Graphics

Lecture 7

Acknowledgment: some images taken from Michael Bronstein's GDL slides; some from Stanford UFLDL Tutorial

Assumed background and level of coverage

- Not a machine learning class, on “need-to-know” basis
- High-level coverage, with some key basics
- Will not assume much past ML experience, if at all

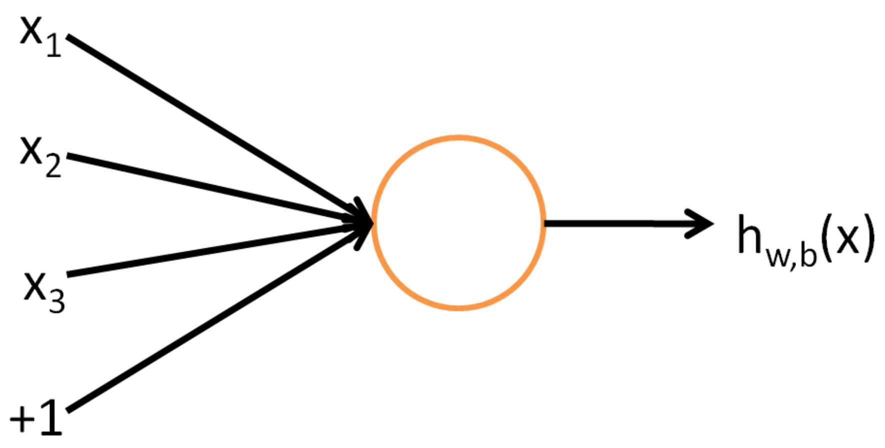
- Focus more on representation learning of 3D shapes
- Focus more on generative models, what graphics is about

What sort of learning are we talking about?

- Try to mimic or simulate how our brain functions
  - Neurons as elementary computational units
  - Use of artificial neural networks, which have a long history tracing back to at least early 1990’s
- Most machine learning or deep learning that people talk about today utilize neural networks
  - Deep learning uses deep neural networks

Neural network basics

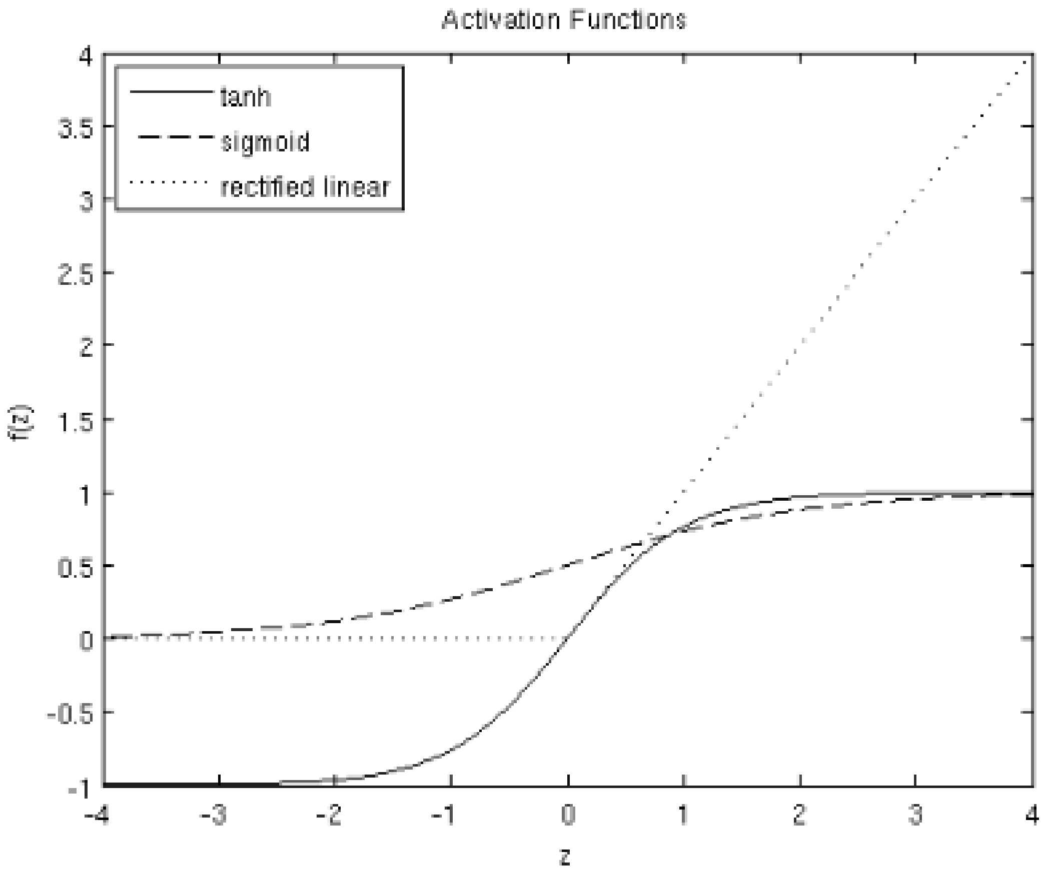
- Let us start with a single neuron to model a perceptron,



- $h_{W,b}(x) = f(W^T x) = f(\sum_{i=1}^{3} W_i x_i + b)$, where $f : \Re \mapsto \Re$ is called an activation function, which is often non-linear

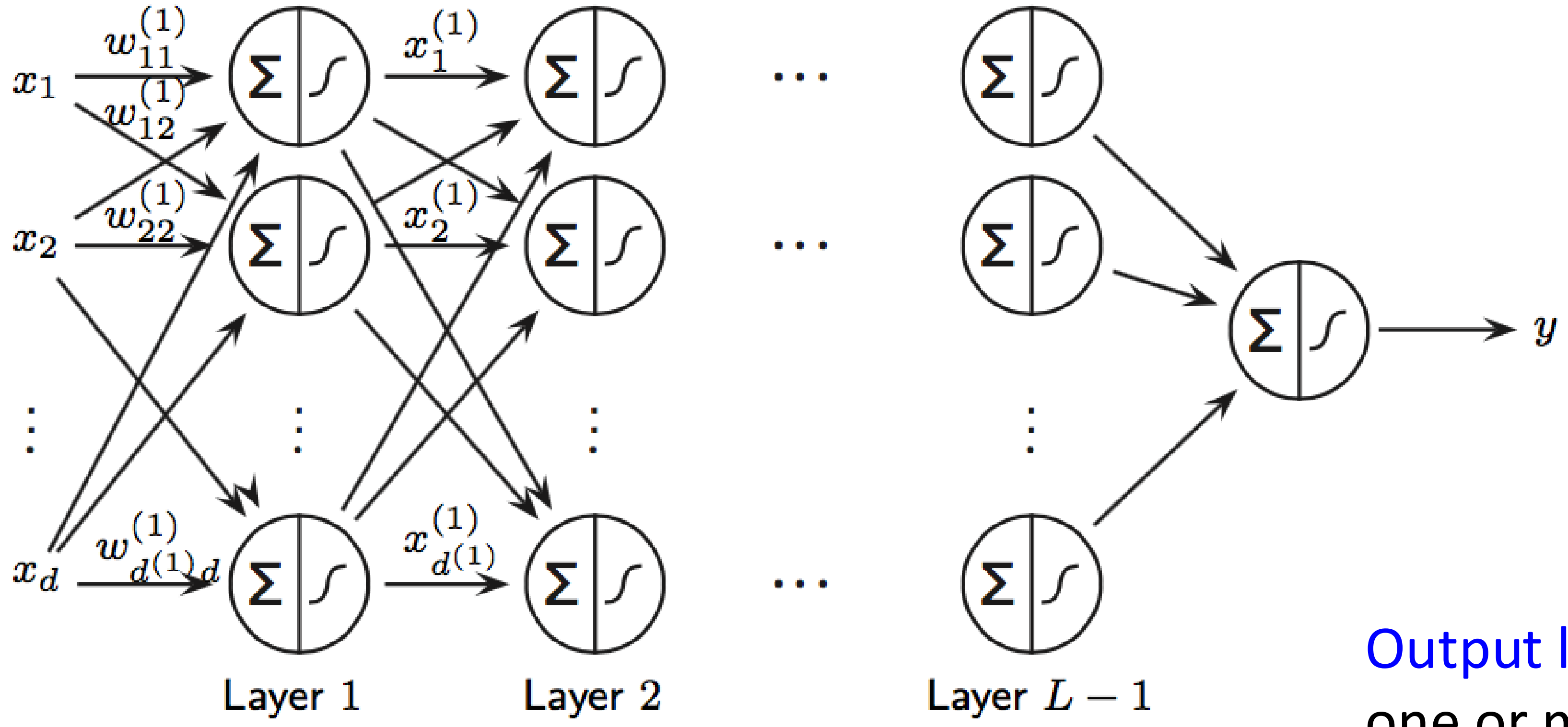
Activation function

- Desirable properties:
  - Nonlinearity: ensures universal approximation
  - Differentiability
  - Monotonicity
  - Etc.



Multi-Layer Perceptron (MLP)

- A feed-forward network: no loops or cycles

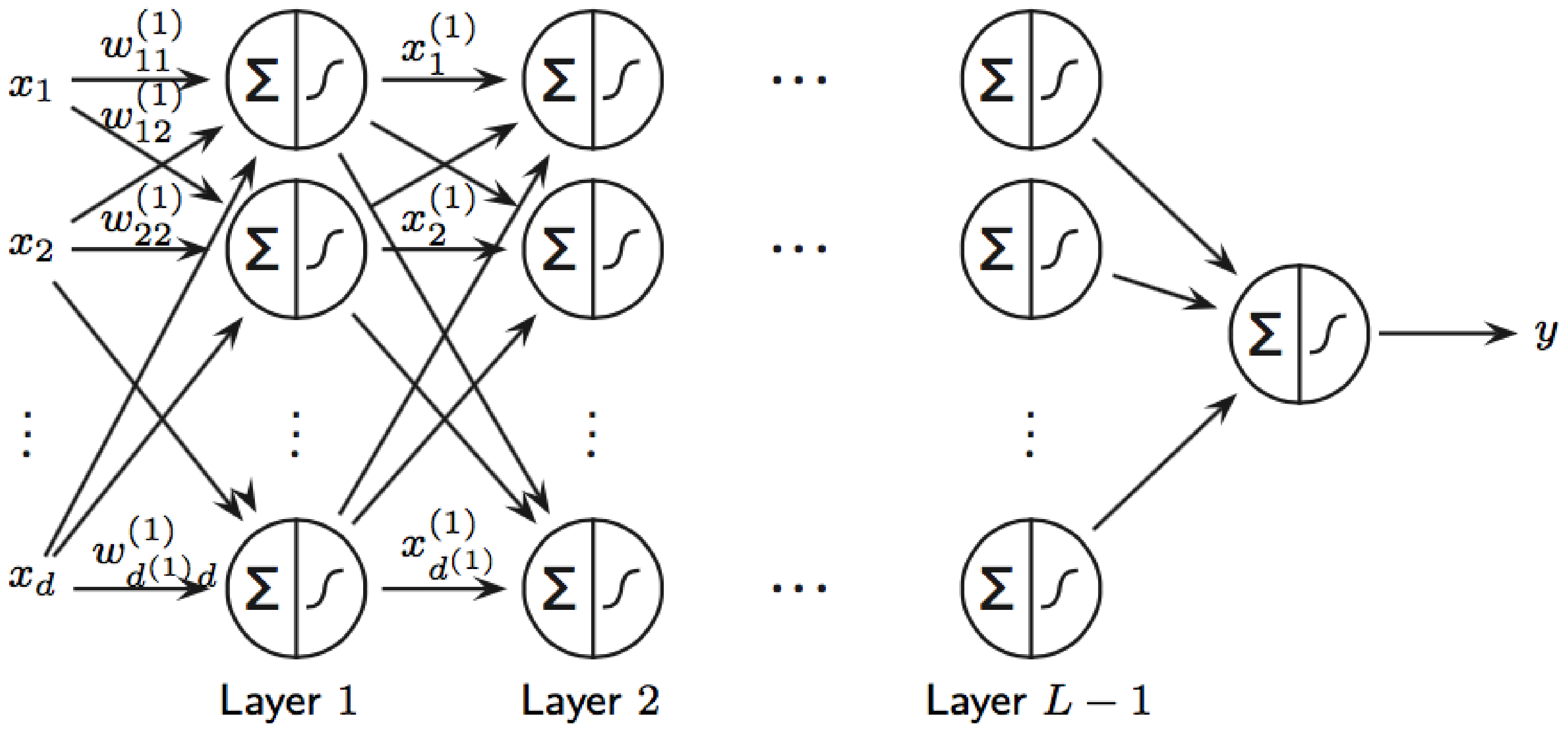


Input layer

Hidden layers

Output layer with one or more output values

“Train” a Network



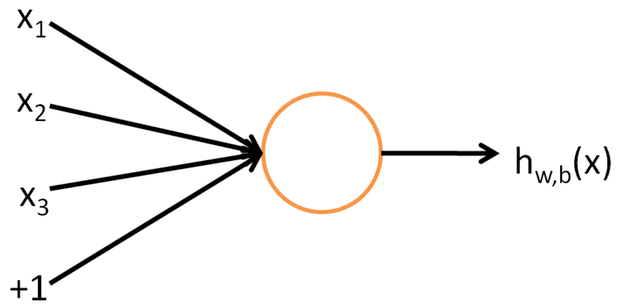
Determine the weight and biases so that the network delivers the assigned job

Loss function

A function quantizing the discrepancies between the current network performance and the actual goals

Like throwing a darts, knowing how far it is from the center (the loss) helps you (the network) tune your force (weight and bias)

Computing the network weights



- Using back-propagation and gradient decent
- Given training pairs $\{(x^{(i)}, y^{(i)})\}$, i=1,…,m, optimize weights W, b to minimize

$$J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m} J(W,b;x^{(i)},y^{(i)})\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2$$

$$= \left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\left\|h_{W,b}(x^{(i)}) - y^{(i)}\right\|^2\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2$$

See: http://ufldl.stanford.edu/tutorial/supervised/MultiLayerNeuralNetworks/

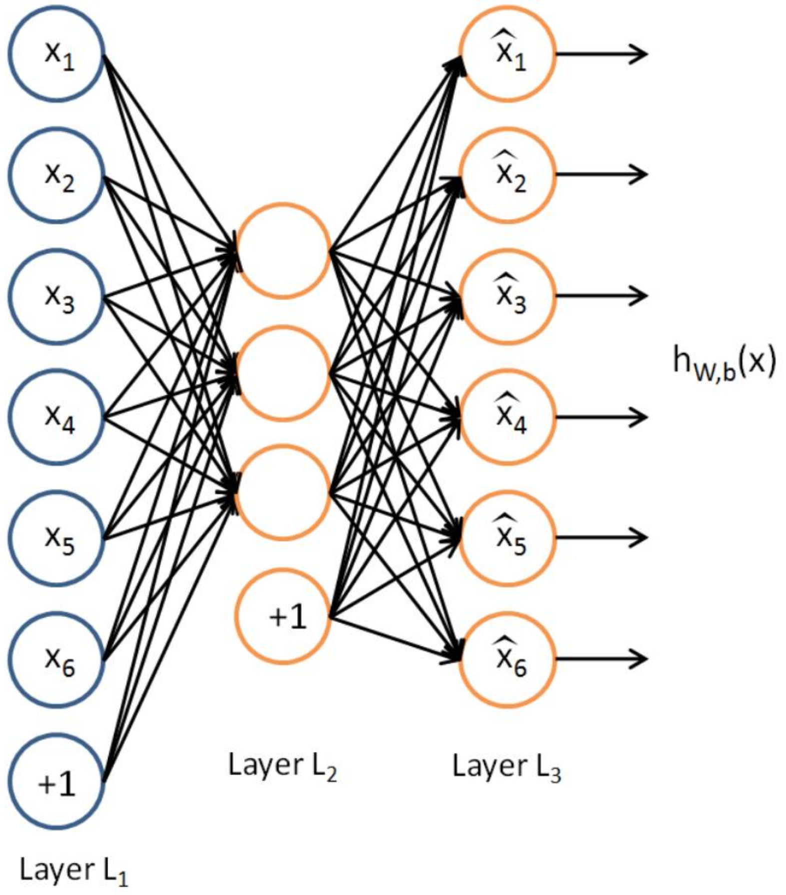
http://neuralnetworksanddeeplearning.com/chap2.html

Beyond supervised learning

- With the "training pairs", we have supervised learning
  - The input-output pairs serve as training or ground truth (GT) data
  - Optimize neural network weights to minimize loss against target outputs
- Unsupervised learning
  - Still optimize neural network weights to minimize some loss
  - But loss definition does not need target or GT data: can "self-supervise"
- Weakly supervised learning, e.g., one-shot learning
- Semi-supervised learning, e.g., active learning

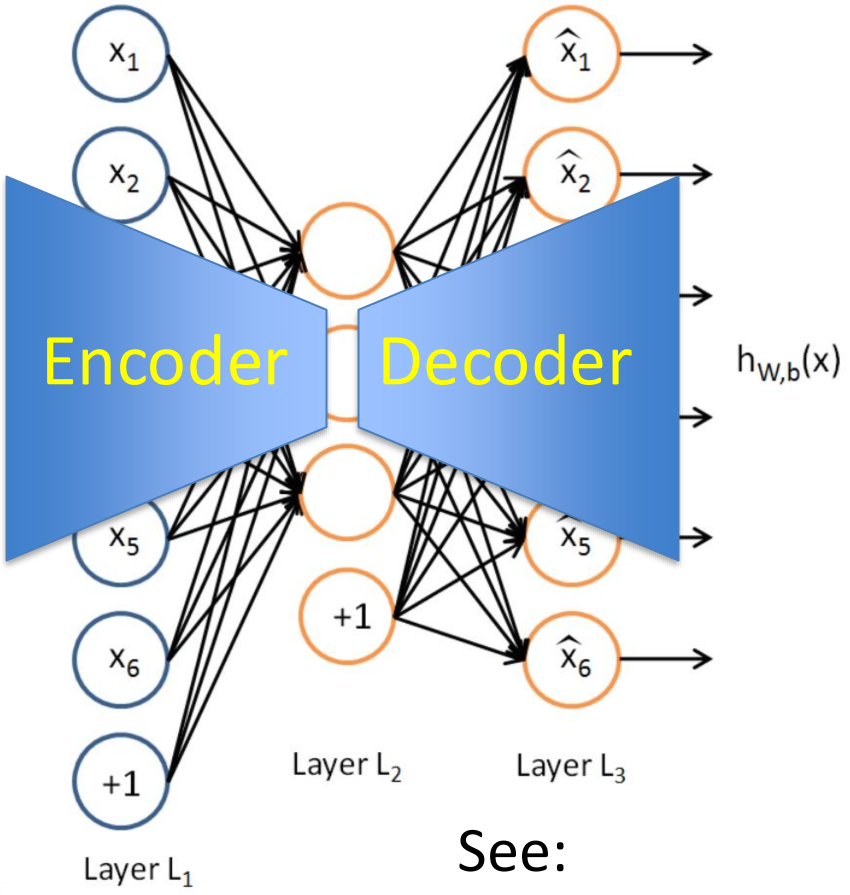
Autoencoder for unsupervised learning

- Network learns to reconstruct the input: learn identity



Autoencoder for unsupervised learning

- Network learns to reconstruct the input: learn identity

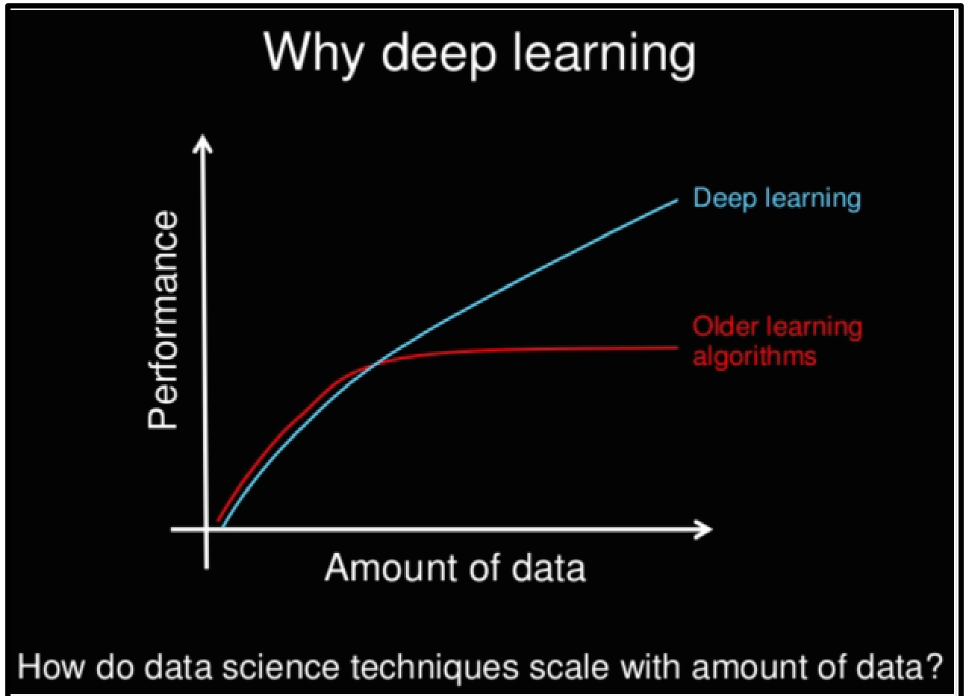


- By limiting the number of hidden units, it is forced to learn a compressed representation
- Representation learning
- Dimensionality reduction

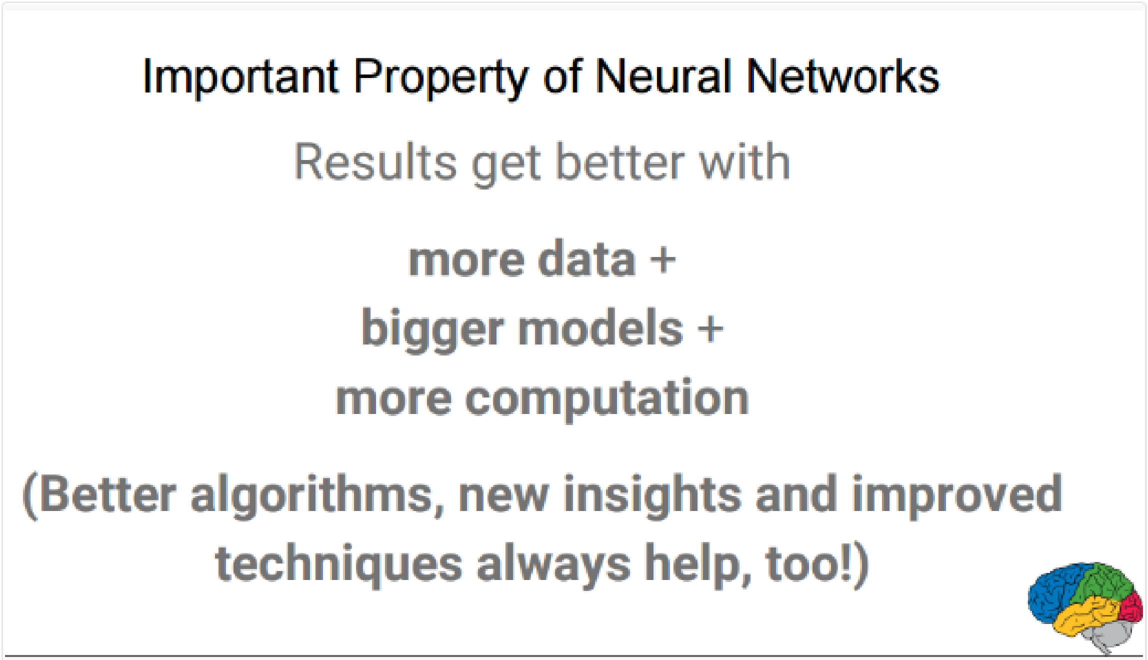
See: http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/

What is deep learning?

- Use of large and deep (many layers) neural networks
  - Many hidden layers and many weights: often deep and wide



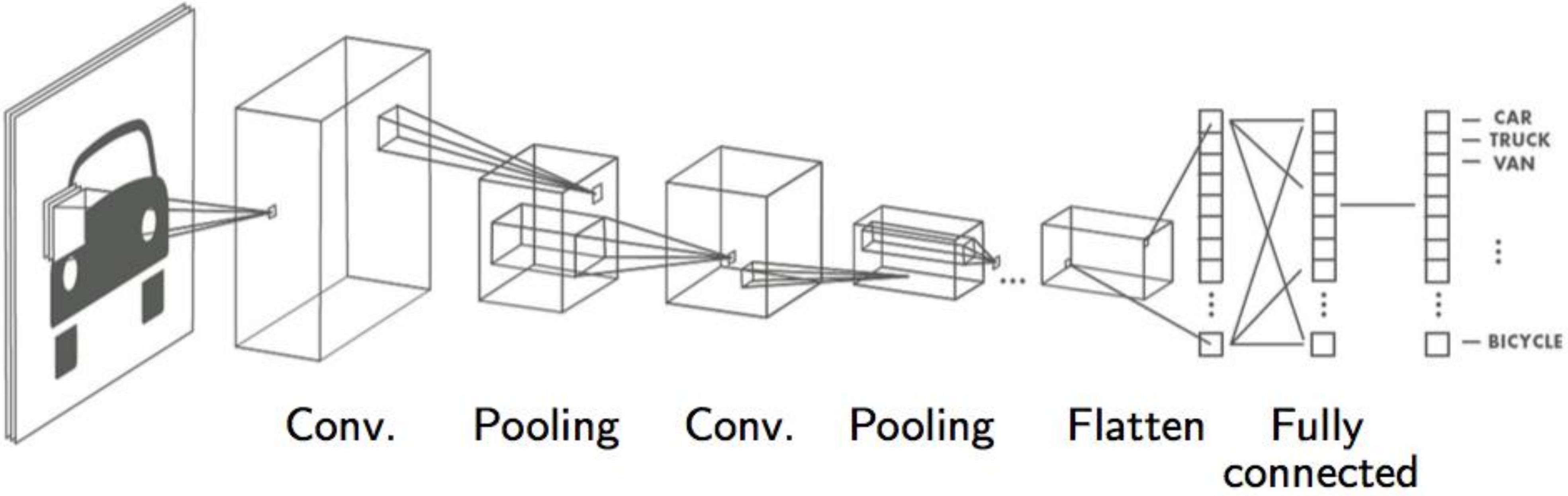
Why Deep Learning?
Slide by Andrew Ng, all rights reserved.



Results Get Better With More Data, Larger Models, More Compute
Slide by Jeff Dean, All Rights Reserved.

Convolutional neural network (CNN)

- A CNN designed for object classification from images

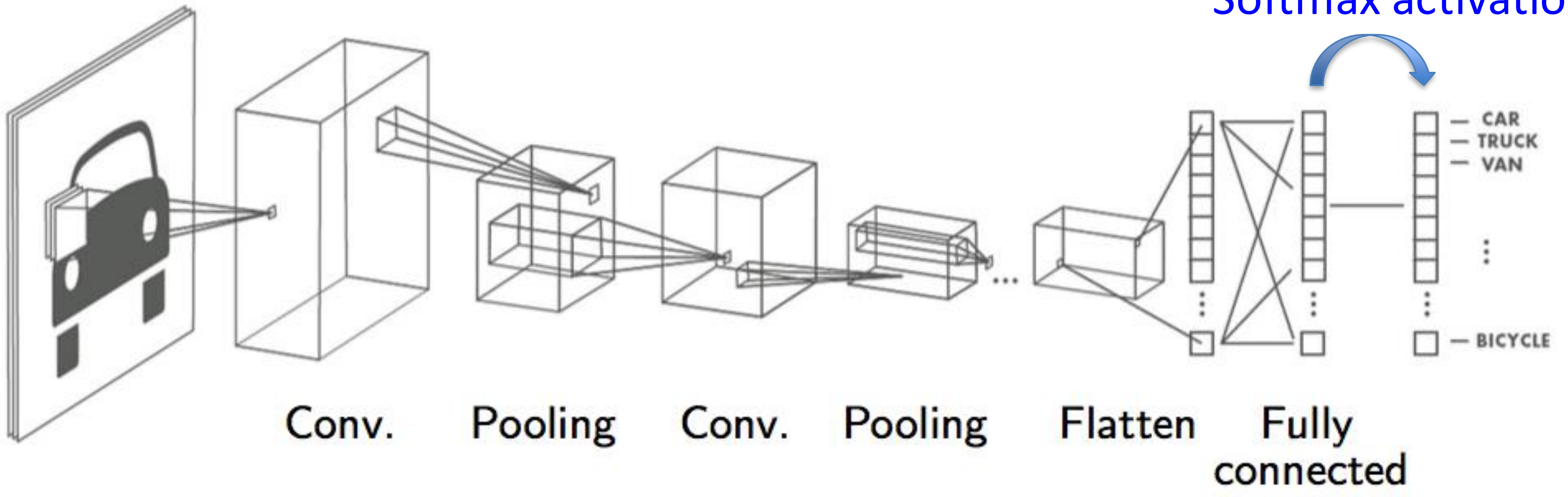


Convolutional neural network (CNN)

- A CNN designed for object classification from images

$$\boxed{\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}}$$

Softmax activation



Convolution: a “running” average

3	3	2	1	0	0
3	3	2	1	0	0
3	3	2	1	0	0
3	3	3	2	0	0
3	3	2	1	0	0
3	2	1	1	0	0

Input image

*

1	0	1
0	1	0
1	0	1

Filter

=

6	8	6	3	1	0
9	13	10	5	2	0
9	14	11	6	3	0
9	13	11	6	2	0
8	13	10	5	3	0
6	7	5	3	1	0

Feature map

Pooling: summarization to reduce spatial res

3	3	2	1	0	0
3	3	2	1	0	0
3	3	2	1	0	0
3	3	3	2	0	0
3	3	2	1	0	0
3	2	1	1	0	0

Input image

*

1	0	1
0	1	0
1	0	1

Filter

=

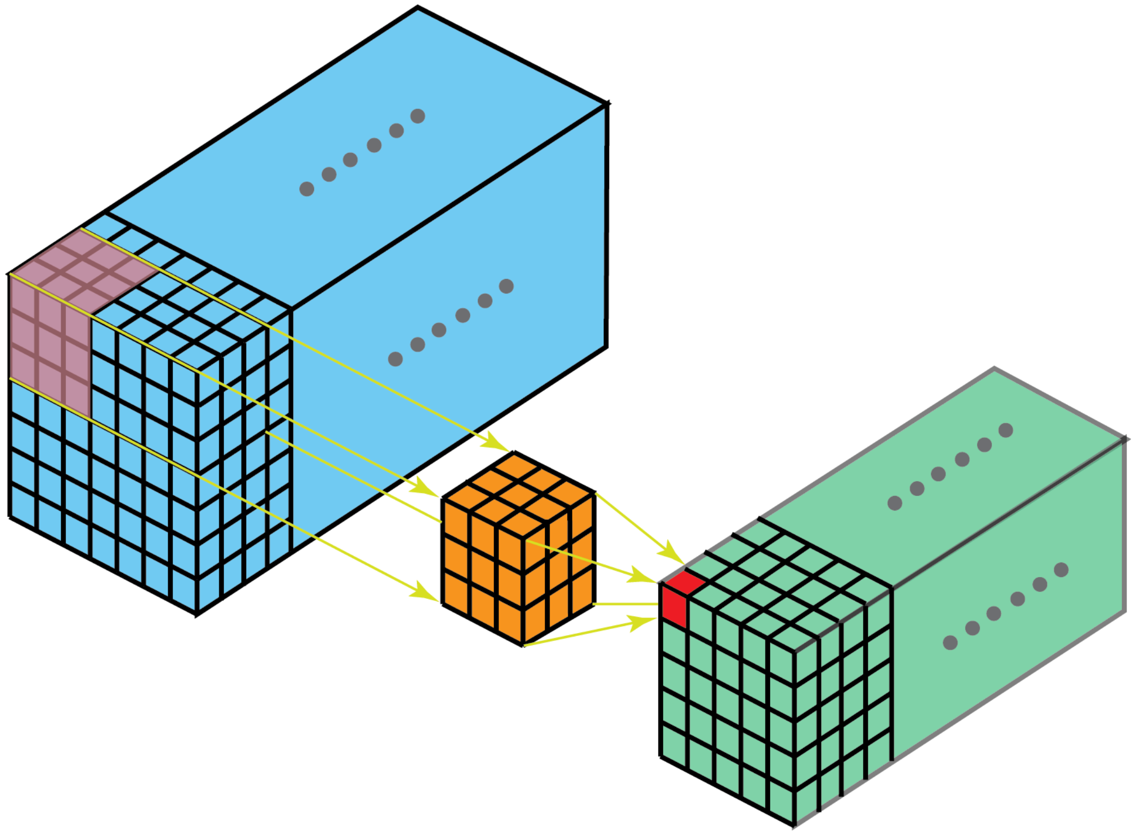
6	8	6	3	1	0
9	13	10	5	2	0
9	14	11	6	3	0
9	13	11	6	2	0
8	13	10	5	3	0
6	7	5	3	1	0

Feature map

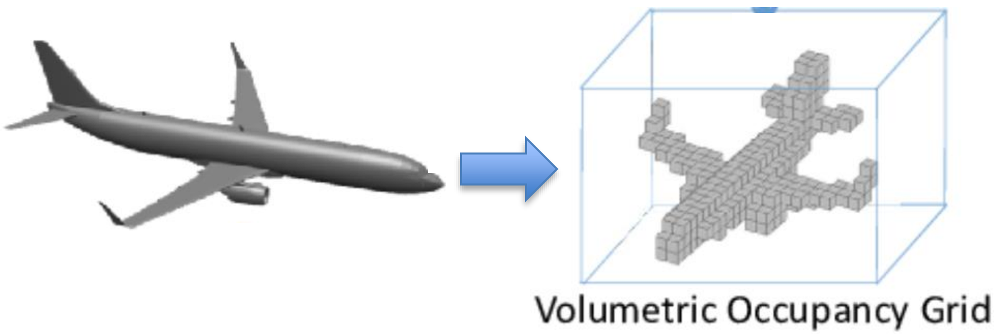
13	10	2
14	11	3
13	10	3

Max pooling

Volumetric convolution: 3D CNN

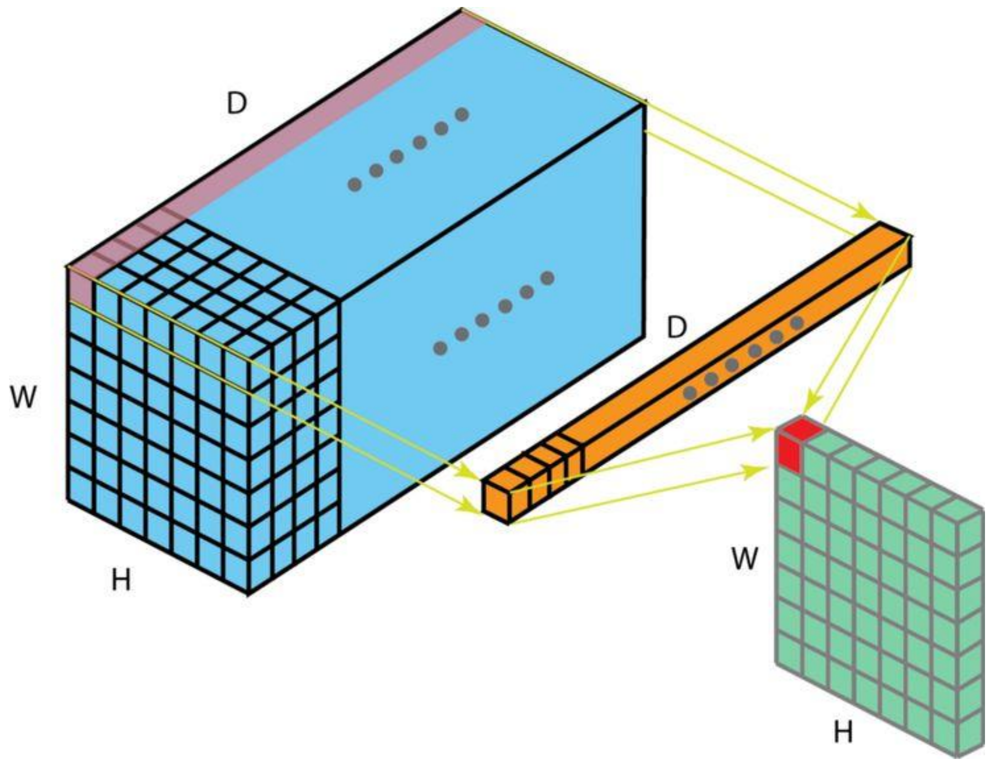


- Straightforward extension from image convolution
- Processes volumetric data, e.g., a 3D shape, or multi-channel image data



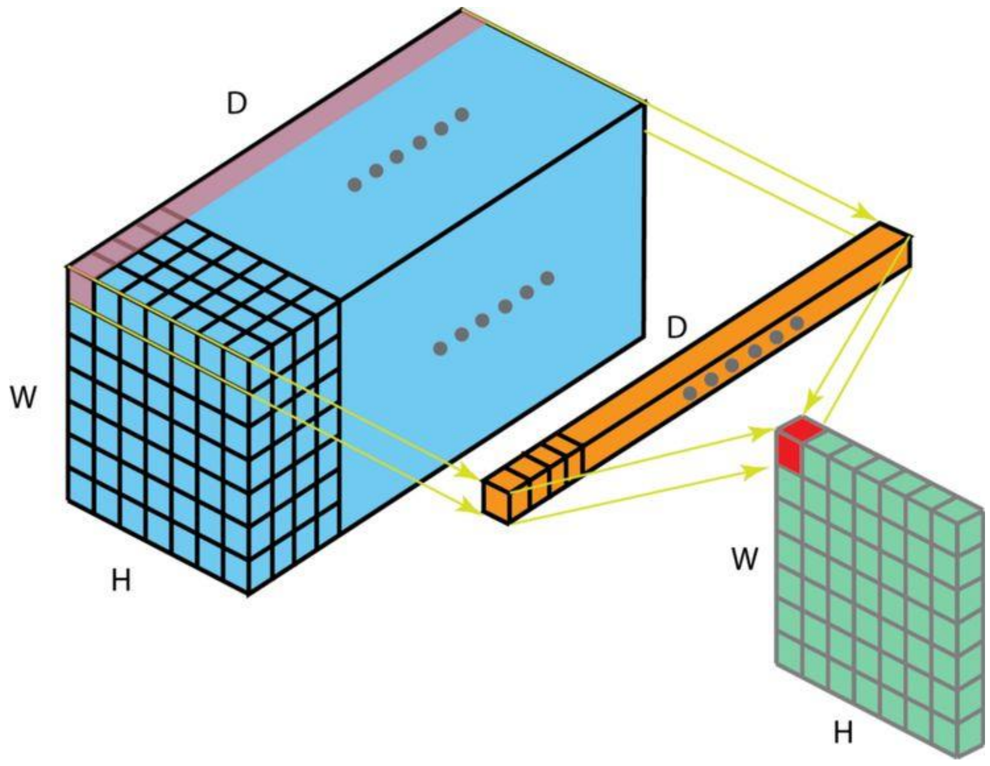
Other variants of convolution

- 1x1 convolution: reducing depth/channel resolution

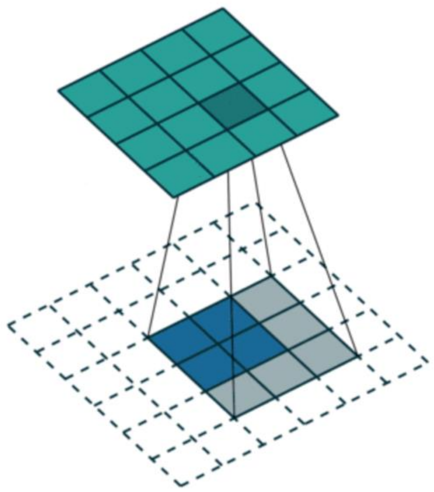


Other variants of convolution

- 1x1 convolution: reducing depth/channel resolution

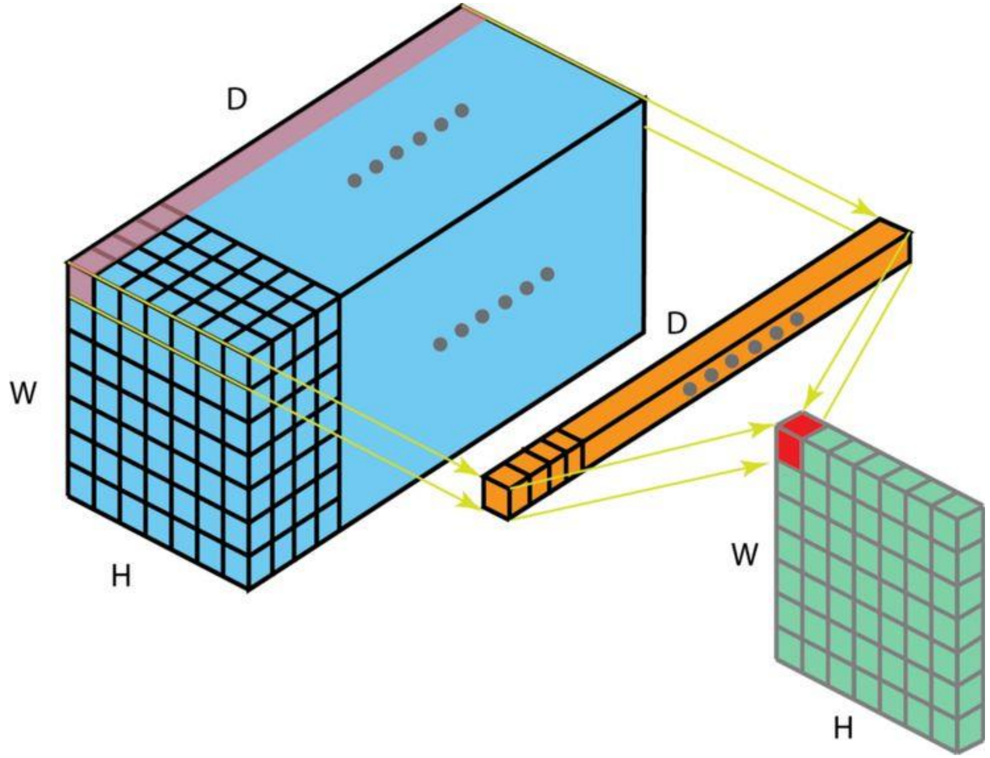


- Deconvolution: upsampling

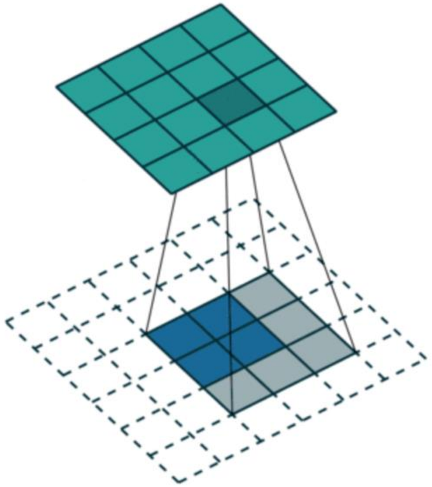


Other variants of convolution

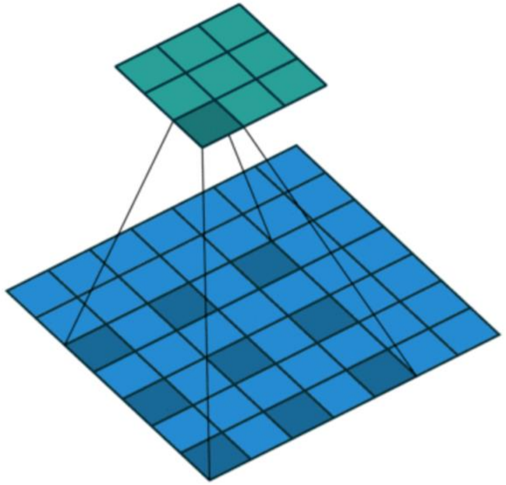
- 1x1 convolution: reducing depth/channel resolution



- Deconvolution: upsampling

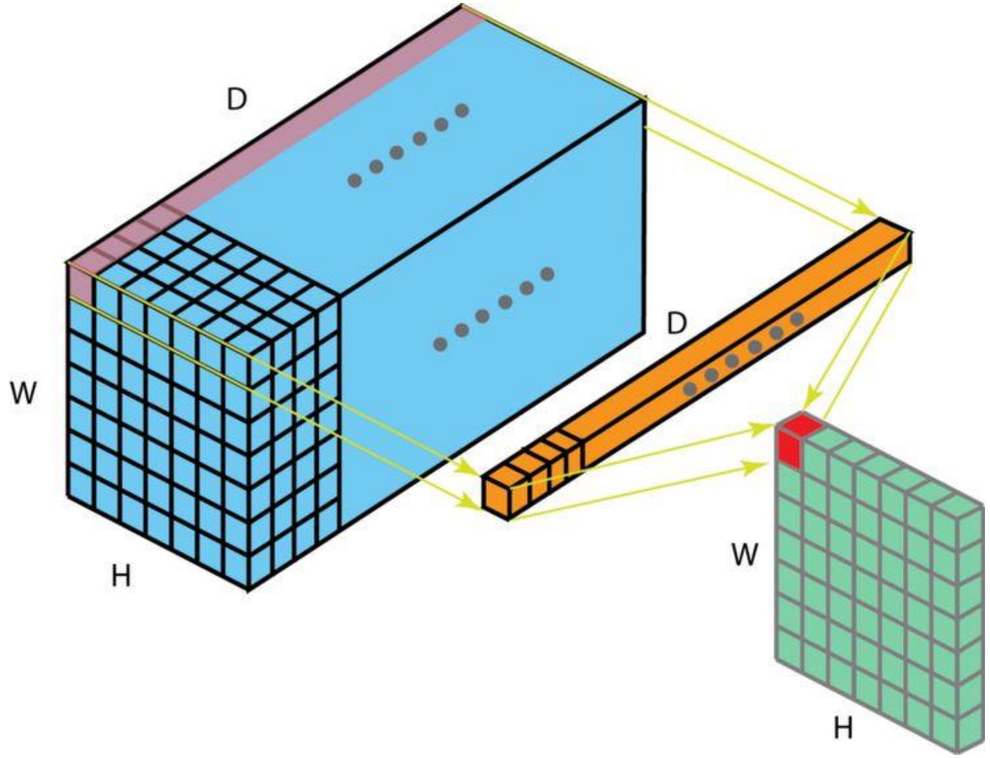


- Dilated convolution

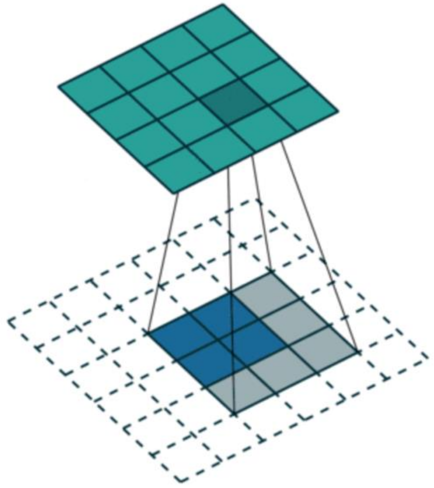


Other variants of convolution

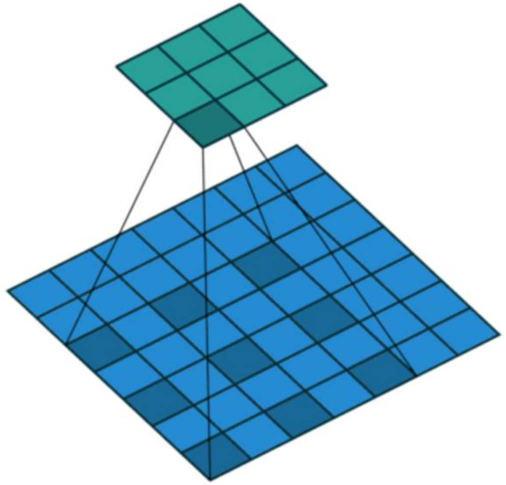
- 1x1 convolution: reducing depth/channel resolution



- Deconvolution: upsampling



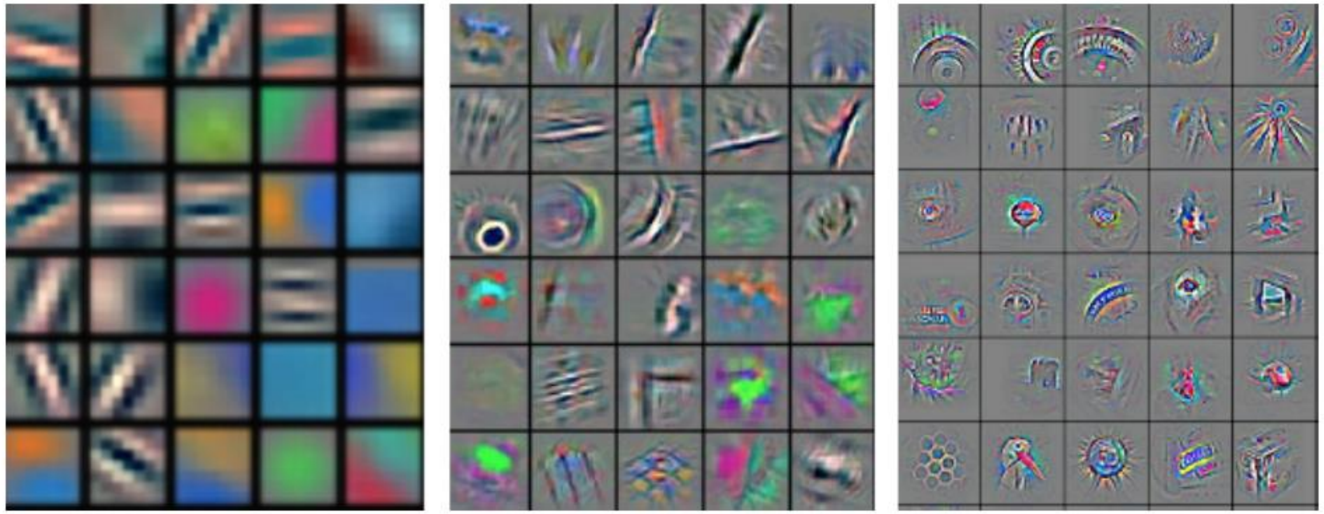
- Dilated convolution



- Graph convolution

Hand-crafted features vs. learned features

- Hand-crafted: e.g., total curvature, normal distance, etc.
- CNNs start with raw images and perform seemingly "uneducated" operations …
- Learned features are reflected in network weights: sensitivity or activation w.r.t. certain patterns in the images



Typical features learned by a CNN becoming increasingly complex at deeper layers

Some key questions about CNNs/DNNs

- How to design the network architecture?
  - From feature hand-crafting to hand-crafting network architecture

Some key questions about CNNs/DNNs

- How to design the network architecture?
  - From feature hand-crafting to hand-crafting network architecture
- How to design networks that generalize well to new data?
  - Avoid overfitting?

Some key questions about CNNs/DNNs

- How to design the network architecture?
  - From feature hand-crafting to hand-crafting network architecture
- How to design networks that generalize well to new data?
  - Avoid overfitting?
- How to ensure there is enough data?
  - Going to weak supervision or data augmentation

Some key questions about CNNs/DNNs

- How to design the network architecture?
  - From feature hand-crafting to hand-crafting network architecture
- How to design networks that generalize well to new data?
  - Avoid overfitting?
- How to ensure there is enough data?
  - Going to weak supervision or data augmentation
- How to improve training efficiency?

Some key questions about CNNs/DNNs

- How to design the network architecture?
  - From feature hand-crafting to hand-crafting network architecture
- How to design networks that generalize well to new data?
  - Avoid overfitting?
- How to ensure there is enough data?
  - Going to weak supervision or data augmentation
- How to improve training efficiency?
- Interpretability or explainability of the networks

Geometric deep learning

- Learn to discriminate or generate geometric data
- Apply deep learning to 3D data
- Can we replicate success of generative DNNs for images?

Remarkable progress on image generation

- Progressive GAN (Generative Adversarial Network) [Nvidia, 2016/17]

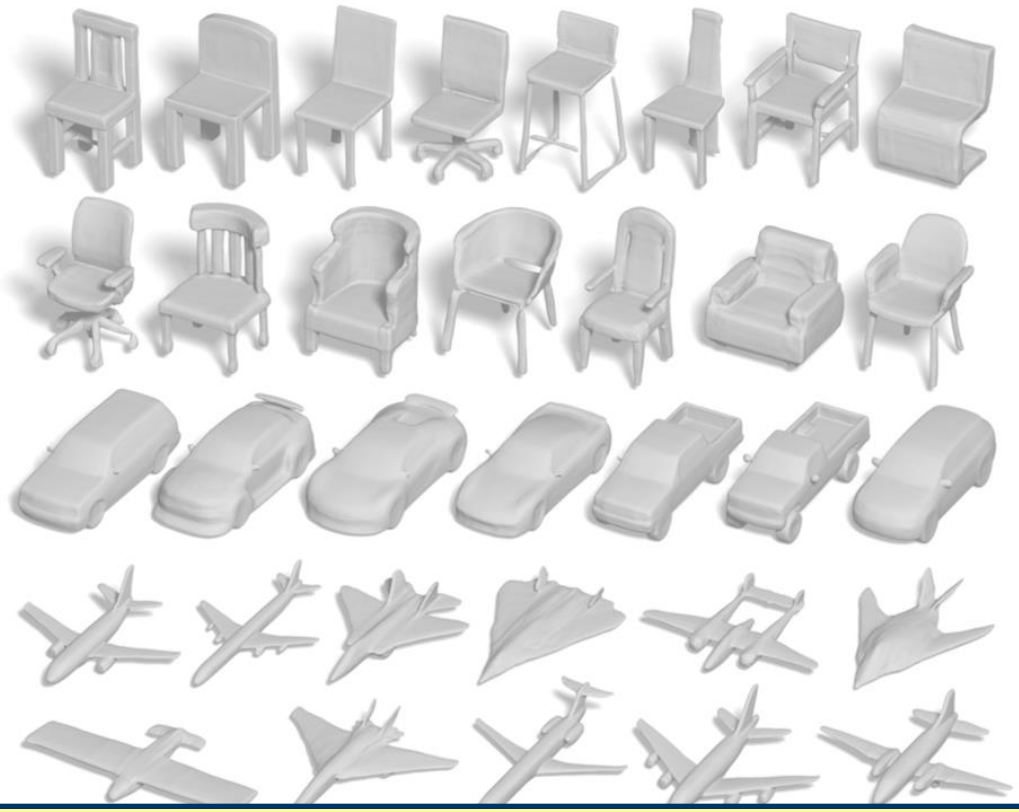
Remarkable progress on image generation

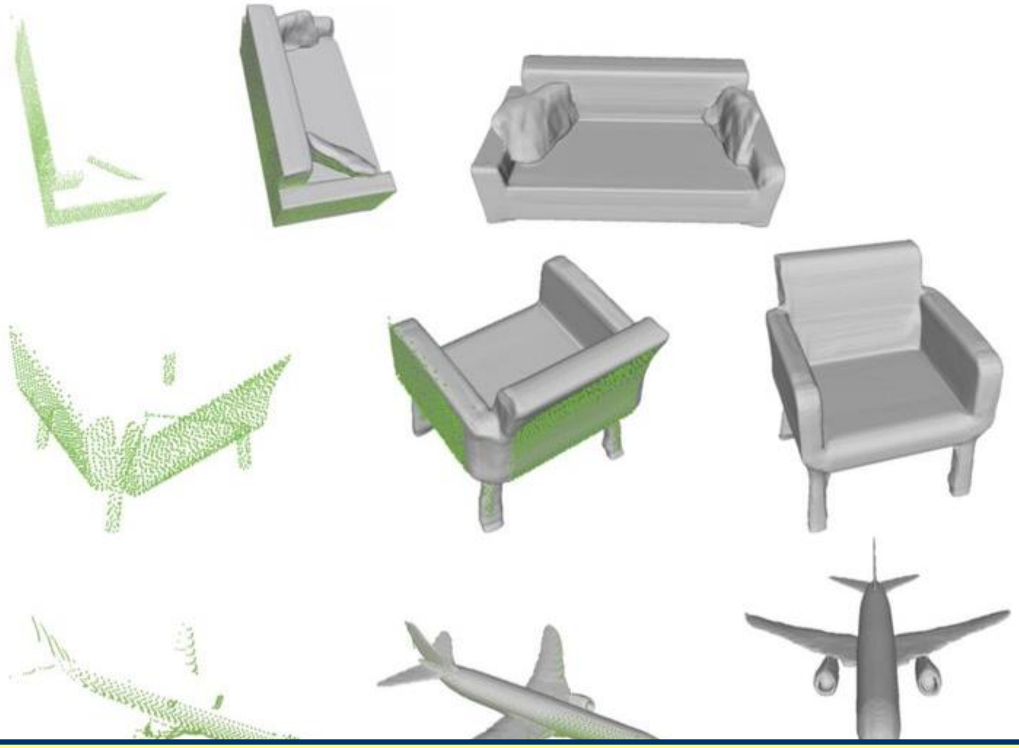
- Progressive GAN (Generative Adversarial Network) [Nvidia, 2016/17]

- Latest: BigGAN [Google Deepmind, 2018/19]

400 x 267 image resolution, using class conditionals

State of the art for 3D shape generation





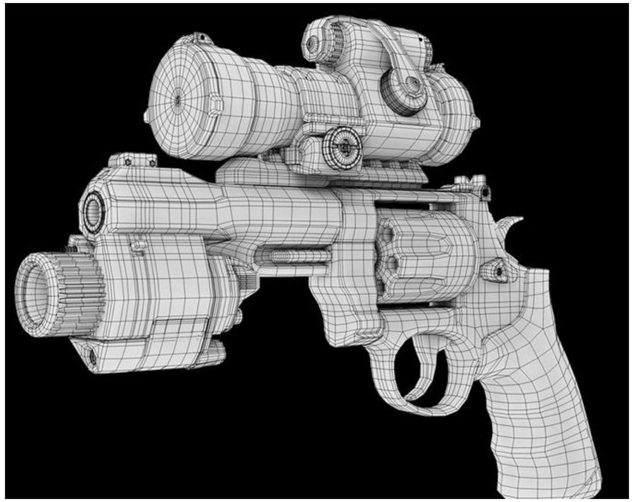
There are some unique challenges to training deep neural networks (DNNs) for 3D shape generation …

Unique challenge #1: representation

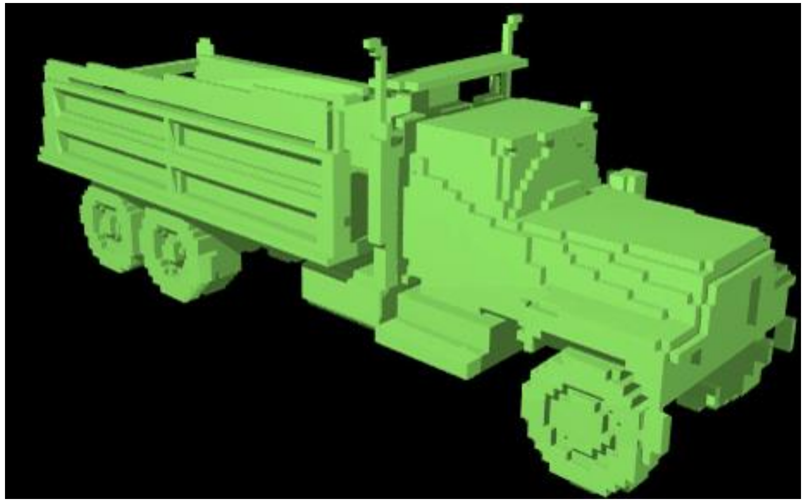
- Unlike images or speech, there is no universally accepted representation or encoding for 3D shapes

Unique challenge #1: representation

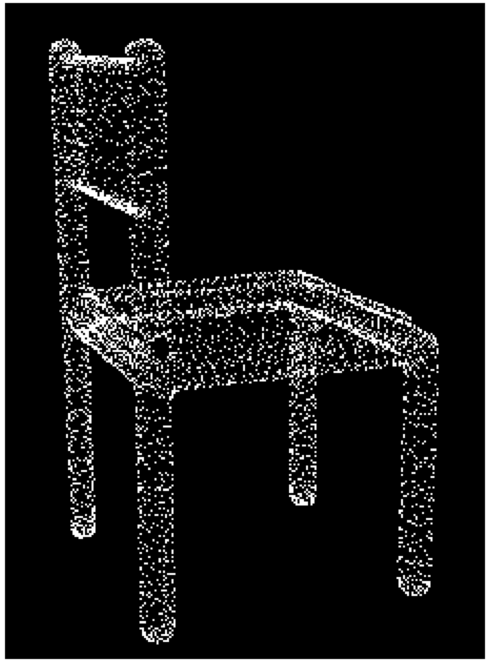
- Unlike images or speech, there is no universally accepted representation or encoding for 3D shapes
- Alternatives: low-level representations



Mesh: a set of triangles



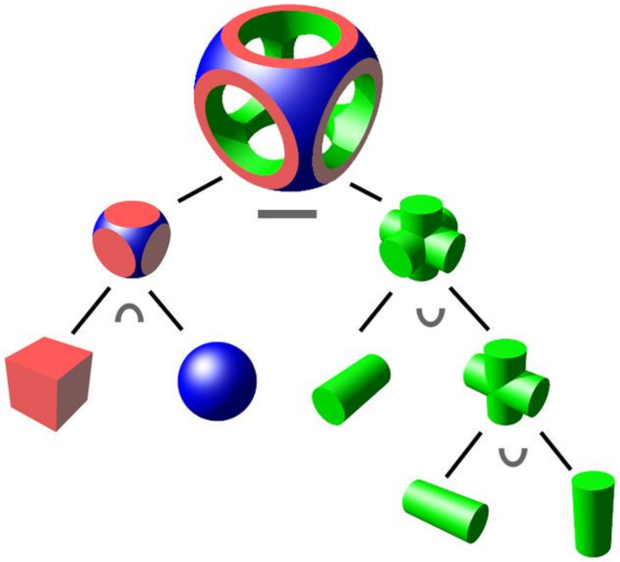
Volume: a grid of voxels



Point cloud: a set of points

Unique challenge #1: representation

- Many other representations



Procedural: e.g., CSG rep

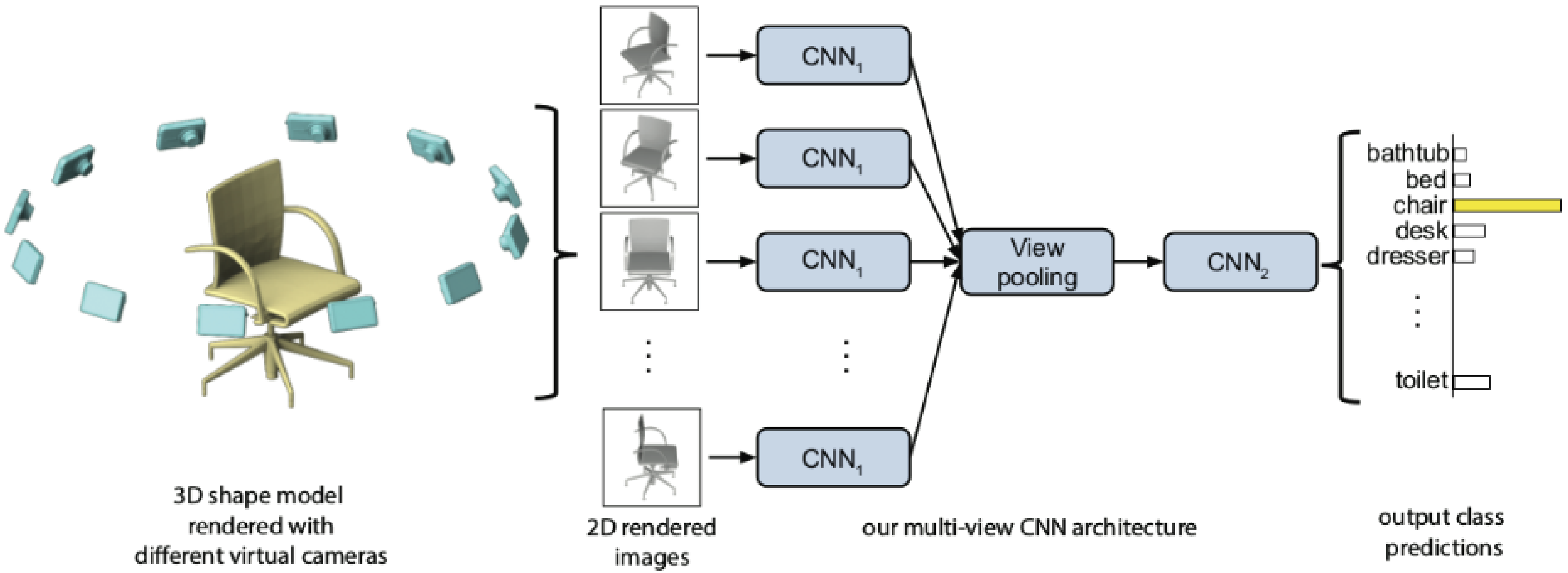
Structural: a set of parts

Multi-view images
in MVCNN [Su et al. 2015]

[Su et al. 2015]

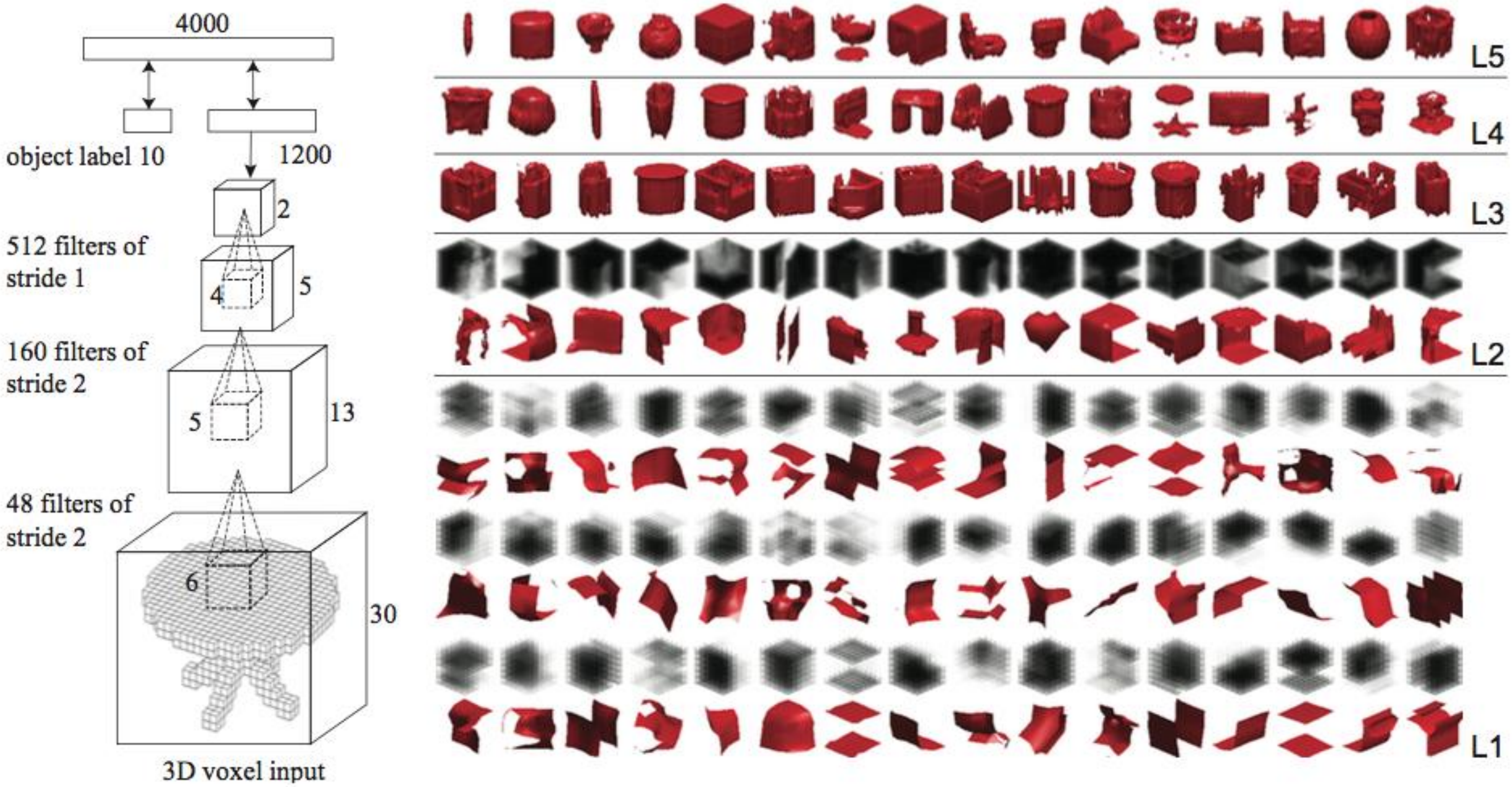
DNN examples: multi-view CNN (MVCNN)

- Reuse standard components of image-based CNNs
- Not geometric; designed for classification not generation



3D ShapeNet: voxel representations [Wu et al. 2015]

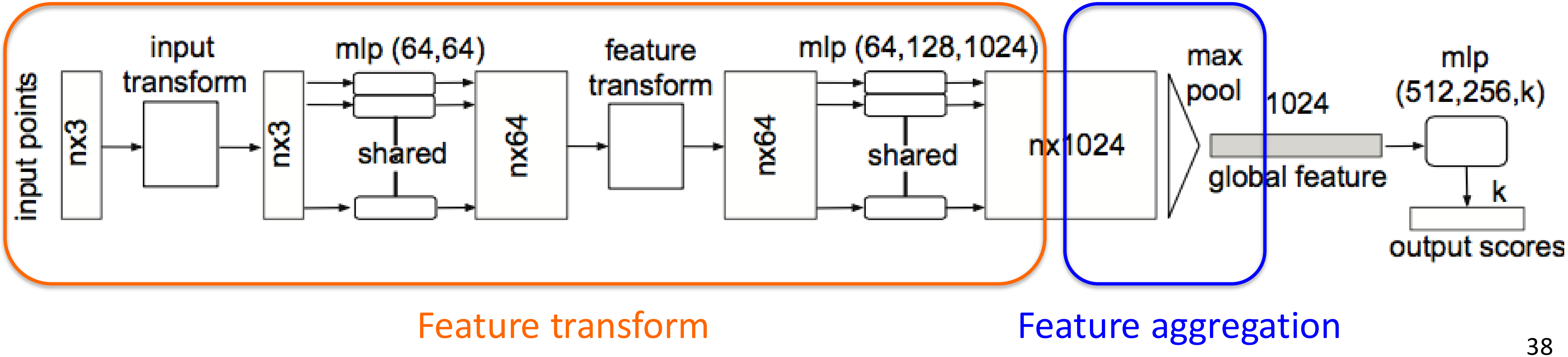
- Straightforward generation of image CNNs, for classification



PointNet

[Qi et al. 2017]

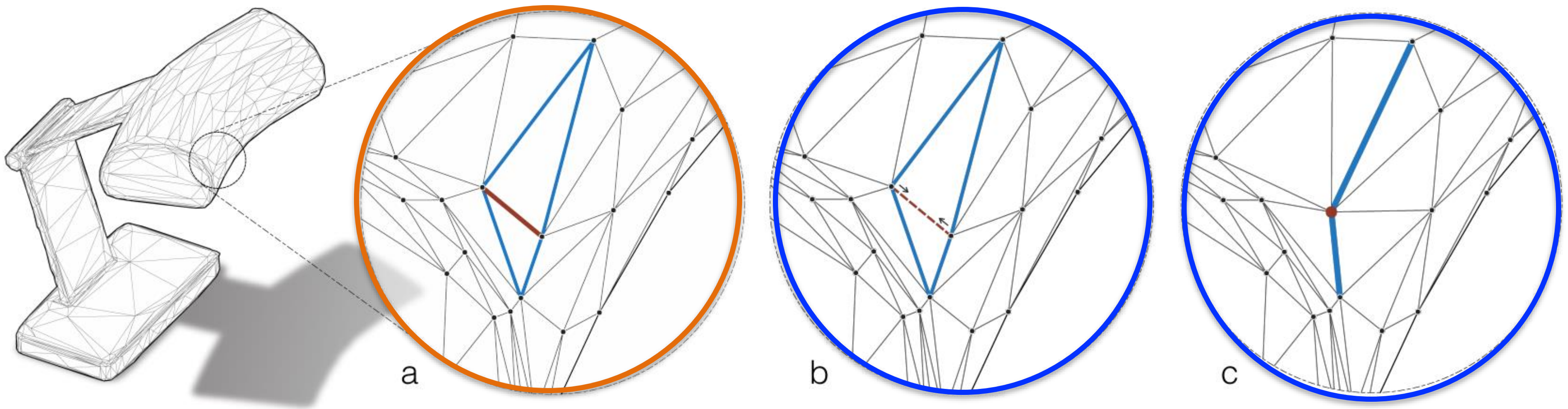
- First transform each 3D point into a high(1,024)-D feature.
- Then aggregate features into a signature for classification
- The max pooling ensures permutation invariance



MeshCNN

[Hanocka et al. 2019]

- Direct conv processing on irregular mesh connectivity
- Mesh edges act as pixels in an image
- Mesh pooling reduces mesh resolution via edge collapse

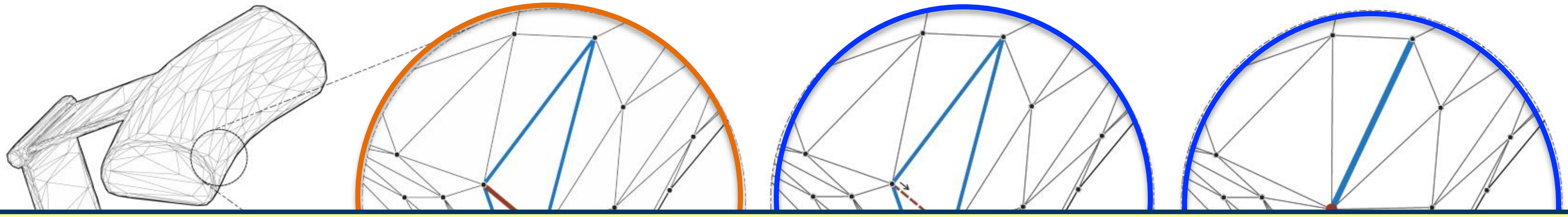


Convolve edge features   Mesh pooling via edge collapse

MeshCNN

[Hanocka et al. 2019]

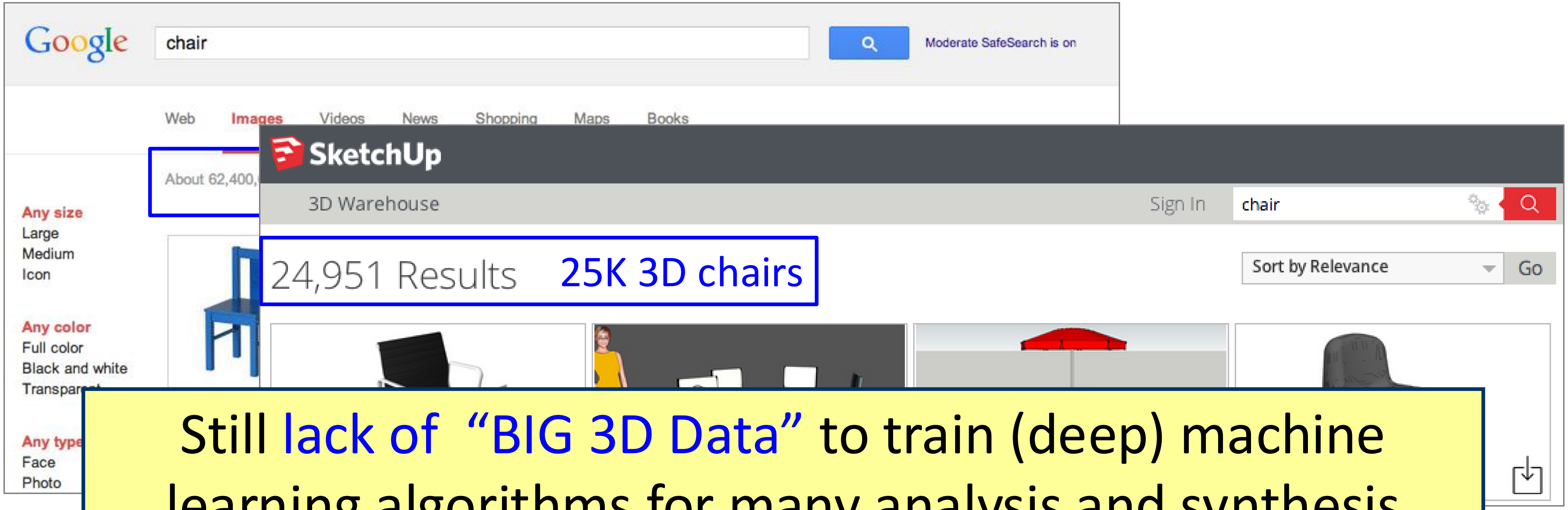
- Direct conv processing on irregular mesh connectivity
- Mesh edges act as pixels in an image
- Mesh pooling reduces mesh resolution via edge collapse



Most networks developed for these reps, in particular, those using convolutional neural networks (CNNs), are designed for discriminative analysis and recognition, not generation.

Unique challenge #2: 3D data challenge

- Acquisition of and interaction with 3D contents are hard



Still lack of "BIG 3D Data" to train (deep) machine learning algorithms for many analysis and synthesis tasks.

Unique challenge #3: affordance/functionality

- 3D objects or designs are meant to be used in real life
  - Not enough to just have the right parts

Unique challenge #3: affordance/functionality

- 3D objects or designs are meant to be used in real life
  - Not enough to just have the right parts
  - Not enough to just “look right” as an image or rendering

Ultimate goal of 3D shape generation

- We live in 3D world to interact with our surroundings
  - We do not just see and observe, we use and we act …

Ultimate goal of 3D shape generation

- We live in 3D world to interact with our surroundings
  - We do not just see and observe, we use and we act …
- Our understanding does not stop at what things are
- Ultimately, the understanding is about
  - how things are
  - how to use them

That is functionality!

3D shapes need to function properly

Early work: theory of affordance (aside)

- Affordance is what the environment offers or affords the individual
  - It presents opportunities for actions afforded by a specific object or environment
  - Agents = humans/hands

J. J. Gibson, *"The Ecological Approach to Visual Perception"*, *1979*

Affordance analysis in vision (aside)

CVPR 2011

What Makes a Chair a Chair?

Helmut Grabner[1] Juergen Gall[1] Luc Van Gool[1,2]

[1]Computer Vision Laboratory
ETH Zurich
{grabner,gall,vangool}@vision.ee.ethz.ch

[2]ESAT - PSI / IBBT
K.U. Leuven
luc.vangool@esat.kuleuven.be

Abstract

Many object classes are primarily defined by their functions. However, this fact has been left largely unexploited by visual object categorization or detection systems. We propose a method to learn an affordance detector. It identifies locations in the 3d space which "support" the particular function. Our novel approach "imagines" an actor performing an action typical for the target object class, instead of relying purely on the visual object appearance. So, function is handled as a cue complementary to appearance, rather than being a consideration after appearance-based detection. Experimental results are given for the functional category "sitting". Such affordance is tested on a 3d representation of the scene, as can be realistically obtained through SfM or depth cameras. In contrast to appearance-based object detectors, affordance detection requires only very few training examples and generalizes very well to other sittable objects like benches or sofas when trained on a few chairs.

1. Introduction

"An object is first identified as having important functional relations, [...] perceptual analysis is derived of the functional concept [...]."
Nelson, 1974, [17]

"Affordances relate the utility of things, events, and places to the needs of animals and their actions in fulfilling them [...]. Affordances themselves are perceived and, in fact, are the essence of what we perceive."
Gibson, 1982, [8, p. 60]

"There's little we can find in common to all chairs – except for their intended use."
Minsky, 1986, [16, p. 123]

"[...] objects like coffee cups are artifacts that were created to fulfill a function. The function of an object plays a critical role in processing that object [... for] categorization and naming."
Carlson-Radvansky *et al.*, 1999, [4]

Figure 1. The "chair-challenge" by I. and H. Bülthoff [3] (reprint with the author's permission).

These quotes emphasize that functional properties or affordances[1] are essential for forming concepts and learning object categories. Experiments (*e.g.* [18, 4]) have demonstrated that both appearance and function are strong cues for learning by infants. Initially they attend only to the form of an object. Later they use form and function and finally (by the age of 18 months) they attend to the relationships between form and function. Furthermore, Booth and Waxman [2] have identified two salient cues that facilitate categorization in infancy, namely (i) object functions and (ii) object names. Moreover, names of objects most often evolve on the basis of function[2].

Whereas all this is well known for a long time, it has been left mostly unused for object detection in computer vision. Taking a look at the results of the recent Pascal VOC Challenge [5], the performance still strongly depends

[1]"*Affordance:* A situation where an object's sensory characteristics intuitively imply its functionality and use. [...] A chair, by its size, its curvature, its balance, and its position, suggests sitting on it.", http://www.usabilityfirst.com/glossary/affordance, 2010/07/28. Introduced in 1979 by Gibson [9, p. 127] based on the verb *afford*.

[2]When considering the evolution of a word for an object, most of the time it is based on its function. For example the word "chair": PIE base **sed-* (to sit) → Latin *sedentarius* (sitting, remaining in one place) → *sedentary* (meaning "not in the habit of exercise") → *cathedral* → *chair*. http://www.etymonline.com, 2010/10/02.

- Fit canonical human poses into 3D scenes to detect sitting affordances

Affordance analysis in vision (aside)

CVPR 2011

What Makes a Chair a Chair?

Helmut Grabner[1] Juergen Gall[1] Luc Van Gool[1,2]

[1]Computer Vision Laboratory
ETH Zurich
{grabner,gall,vangool}@vision.ee.ethz.ch

[2]ESAT - PSI / IBBT
K.U. Leuven
luc.vangool@esat.kuleuven.be

Abstract

Many object classes are primarily defined by their functions. However, this fact has been left largely unexploited by visual object categorization or detection systems. We propose a method to learn an affordance detector. It identifies locations in the 3d space which "support" the particular function. Our novel approach "imagines" an actor performing an action typical for the target object class, instead of relying purely on the visual object appearance. So, function is handled as a cue complementary to appearance, rather than being a consideration after appearance-based detection. Experimental results are given for the functional category "sitting". Such affordance is tested on a 3d representation of the scene, as can be realistically obtained through SfM or depth cameras. In contrast to appearance-based object detectors, affordance detection requires only very few training examples and generalizes very well to other sittable objects like benches or sofas when trained on a few chairs.

1. Introduction

"An object is first identified as having important functional relations, [...] perceptual analysis is derived of the functional concept [...]."
Nelson, 1974, [17]

"Affordances relate the utility of things, events, and places to the needs of animals and their actions in fulfilling them [...]. Affordances themselves are perceived and, in fact, are the essence of what we perceive."
Gibson, 1982, [8, p. 60]

"There's little we can find in common to all chairs – except for their intended use."
Minsky, 1986, [16, p. 123]

"[...] objects like coffee cups are artifacts that were created to fulfill a function. The function of an object plays a critical role in processing that object [... for] categorization and naming."
Carlson-Radvansky *et al.*, 1999, [4]

Figure 1. The "chair-challenge" by I. and H. Bülthoff [3] (reprint with the author's permission).

These quotes emphasize that functional properties or affordances[1] are essential for forming concepts and learning object categories. Experiments (*e.g.* [18, 4]) have demonstrated that both appearance and function are strong cues for learning by infants. Initially they attend only to the form of an object. Later they use form and function and finally (by the age of 18 months) they attend to the relationships between form and function. Furthermore, Booth and Waxman [2] have identified two salient cues that facilitate categorization in infancy, namely (i) object functions and (ii) object names. Moreover, names of objects most often evolve on the basis of function[2].

Whereas all this is well known for a long time, it has been left mostly unused for object detection in computer vision. Taking a look at the results of the recent Pascal VOC Challenge [5], the performance still strongly depends

[1]"*Affordance:* A situation where an object's sensory characteristics intuitively imply its functionality and use. [...] A chair, by its size, its curvature, its balance, and its position, suggests sitting on it.", http://www.usabilityfirst.com/glossary/affordance, 2010/07/28. Introduced in 1979 by Gibson [9, p. 127] based on the verb *afford*.

[2]When considering the evolution of a word for an object, most of the time it is based on its function. For example the word "chair": PIE base **sed-* (to sit) → Latin *sedentarius* (sitting, remaining in one place) → *sedentary* (meaning "not in the habit of exercise") → *cathedral* → *chair*. http://www.etymonline.com, 2010/10/02.

- Fit canonical human poses into 3D scenes to detect sitting affordances

"An object is first identified as having important functional relations, […], perceptual analysis is derived of the functional concept […]

Nelson [1974]

Affordance analysis in vision (aside)

CVPR 2011

What Makes a Chair a Chair?

Helmut Grabner[1] Juergen Gall[1] Luc Van Gool[1,2]

[1]Computer Vision Laboratory
ETH Zurich
{grabner,gall,vangool}@vision.ee.ethz.ch

[2]ESAT - PSI / IBBT
K.U. Leuven
luc.vangool@esat.kuleuven.be

Abstract

Many object classes are primarily defined by their functions. However, this fact has been left largely unexploited by visual object categorization or detection systems. We propose a method to learn an affordance detector. It identifies locations in the 3d space which "support" the particular function. Our novel approach "imagines" an actor performing an action typical for the target object class, instead of relying purely on the visual object appearance. So, function is handled as a cue complementary to appearance, rather than being a consideration after appearance-based detection. Experimental results are given for the functional category "sitting". Such affordance is tested on a 3d representation of the scene, as can be realistically obtained through SfM or depth cameras. In contrast to appearance-based object detectors, affordance detection requires only very few training examples and generalizes very well to other sittable objects like benches or sofas when trained on a few chairs.

1. Introduction

"An object is first identified as having important functional relations, [...] perceptual analysis is derived of the functional concept [...]."

Nelson, 1974, [17]

"Affordances relate the utility of things, events, and places to the needs of animals and their actions in fulfilling them [...]. Affordances themselves are perceived and, in fact, are the essence of what we perceive."

Gibson, 1982, [8, p. 60]

"There's little we can find in common to all chairs – except for their intended use."

Minsky, 1986, [16, p. 123]

"[...] objects like coffee cups are artifacts that were created to fulfill a function. The function of an object plays a critical role in processing that object [... for] categorization and naming."

Carlson-Radvansky *et al.*, 1999, [4]

Figure 1. The "chair-challenge" by I. and H. Bülthoff [3] (reprint with the author's permission).

These quotes emphasize that functional properties or affordances[1] are essential for forming concepts and learning object categories. Experiments (*e.g.* [18, 4]) have demonstrated that both appearance and function are strong cues for learning by infants. Initially they attend only to the form of an object. Later they use form and function and finally (by the age of 18 months) they attend to the relationships between form and function. Furthermore, Booth and Waxman [2] have identified two salient cues that facilitate categorization in infancy, namely (i) object functions and (ii) object names. Moreover, names of objects most often evolve on the basis of function[2].

Whereas all this is well known for a long time, it has been left mostly unused for object detection in computer vision. Taking a look at the results of the recent Pascal VOC Challenge [5], the performance still strongly depends

[1]"*Affordance*: A situation where an object's sensory characteristics intuitively imply its functionality and use. [...] A chair, by its size, its curvature, its balance, and its position, suggests sitting on it.", http://www.usabilityfirst.com/glossary/affordance, 2010/07/28. Introduced in 1979 by Gibson [9, p. 127] based on the verb *afford*.

[2]When considering the evolution of a word for an object, most of the time it is based on its function. For example the word "chair": PIE base **sed-* (to sit) → Latin *sedentarius* (sitting, remaining in one place) → *sedentary* (meaning "not in the habit of exercise") → *cathedral* → *chair*. http://www.etymonline.com, 2010/10/02.

- Fit canonical human poses into 3D scenes to detect sitting affordances

"There's little we can find in common to all chairs – except for their intended use."

Minsky [1986]

How to define affordance/functionality?

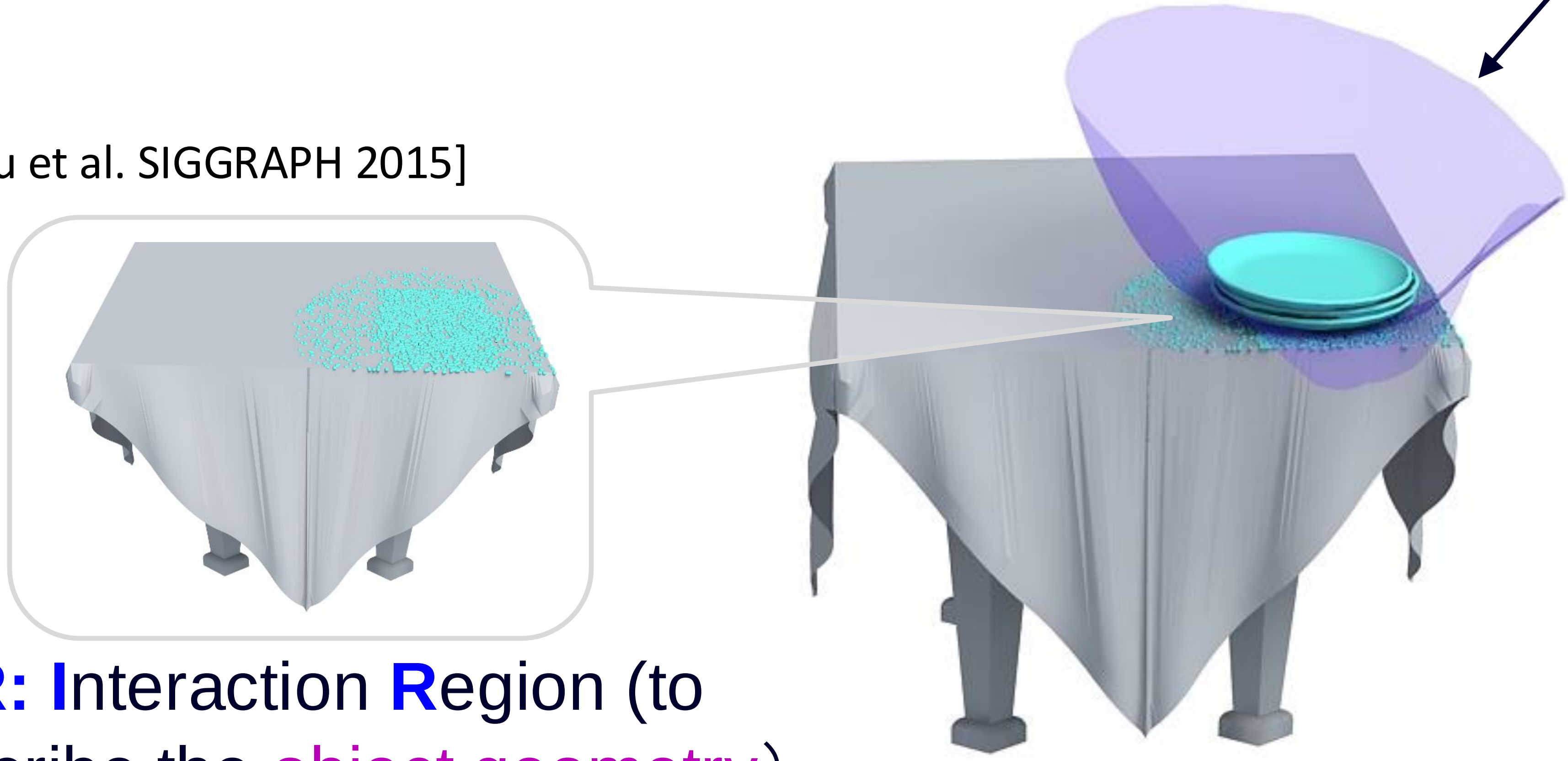
- Interactions between a 3D object with other objects (the agents) in a given scene context reflects its functionality
- Agents can be
  - Humans or hands
  - Other 3D objects

How to represent object-object interactions

IBS: Intersection **B**isector **S**urface (to describe the interaction)

[Zhao et al. TOG 2014]

[Hu et al. SIGGRAPH 2015]



IR: Interaction **R**egion (to describe the object geometry)

Before functionality …

- Results of 3D generative NNs still visually unsatisfactory
  - Low resolution and geometric/structural/topological noise

It is important to find the right shape representations for training DNNs to generate quality 3D shapes

What is a shape?

shape

/SHāp/

noun

1. the external form or appearance characteristic of someone or something; the outline of an area or figure.
   "she liked the shape of his nose"
   synonyms: form, appearance, configuration, formation, structure; More

Shape vs. image

vs.

shape

/SHāp/

noun

1. the external form or appearance characteristic of someone or something; the outline of an area or figure.
   "she liked the shape of his nose"
   synonyms: form, appearance, configuration, formation, structure; More

Shape vs. image

shape

/SHāp/

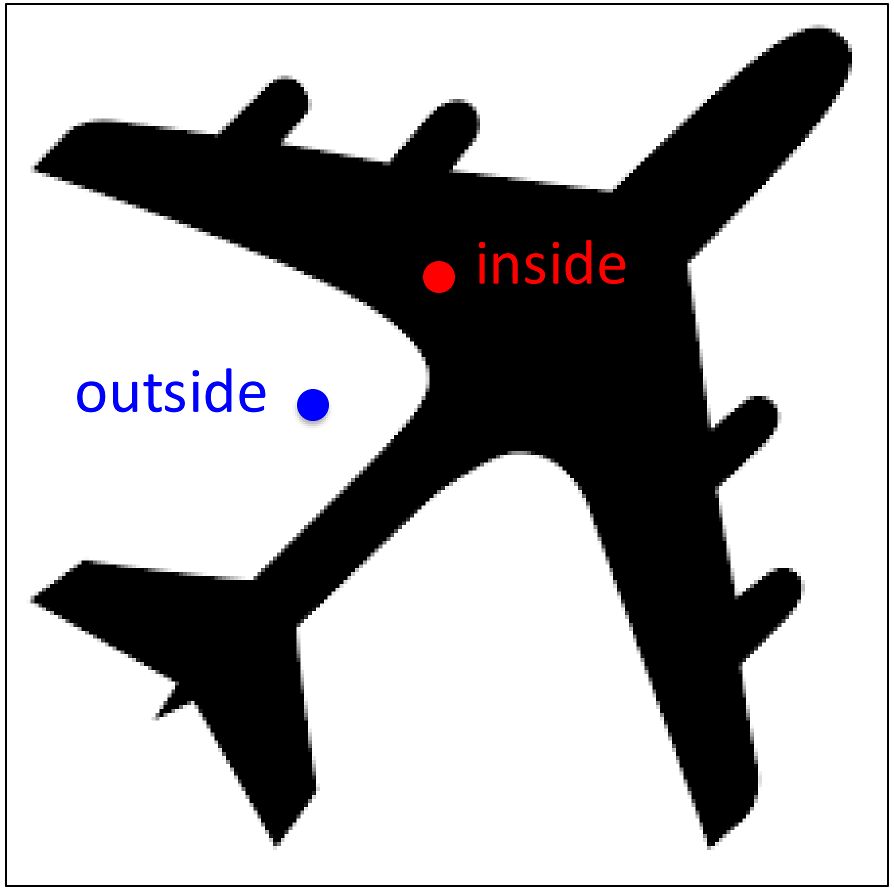
noun

1. the external form or appearance characteristic of someone or something; the outline of an area or figure.
   "she liked the shape of his nose"
   synonyms: form, appearance, configuration, formation, structure; More

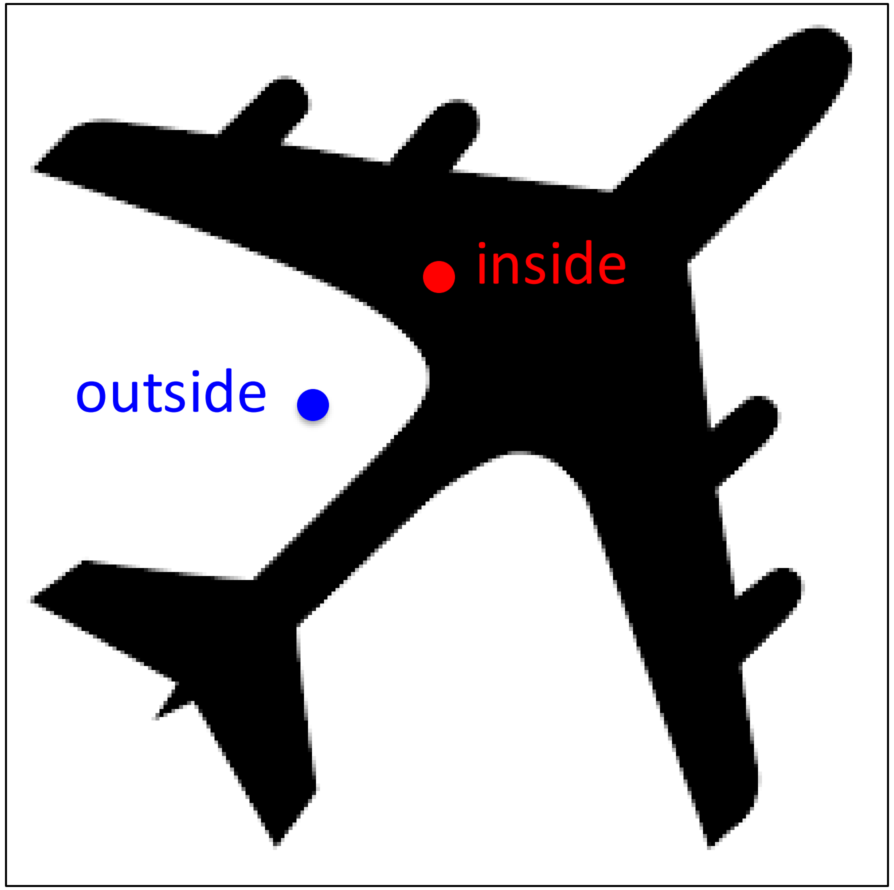
vs.

- A shape is defined/characterized by its boundary/outline
- Image boundary is artificial: it is because we had to crop

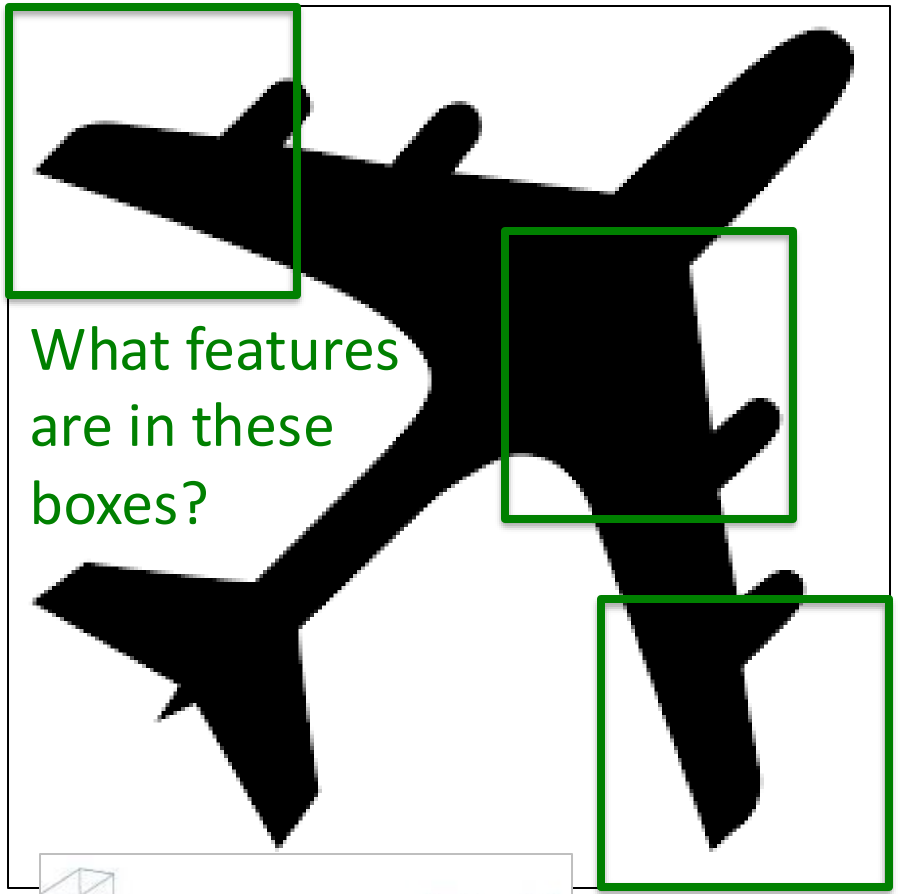
Shape boundary is about what is inside/outside

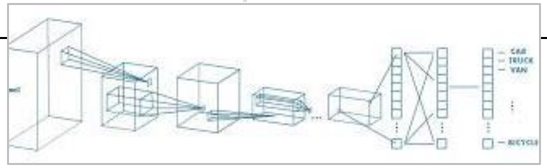


It is not really about feature inference



vs. image convolution via CNNs





CNNs “see” textures, humans see shapes (aside)

(a) Texture image
81.4% **Indian elephant**
10.3% indri
8.2% black swan

(b) Content image
71.1% **tabby cat**
17.3% grey fox
3.3% Siamese cat

(c) Texture-shape cue conflict
63.9% **Indian elephant**
26.4% indri
9.6% black swan

“ImageNet-trained CNNs are biased towards texture; Increasing shape bias improves accuracy and robustness” [Geirhos et al. ICLR 2019]

To learn to generate shapes, we should ask …

Is this point inside the shape?

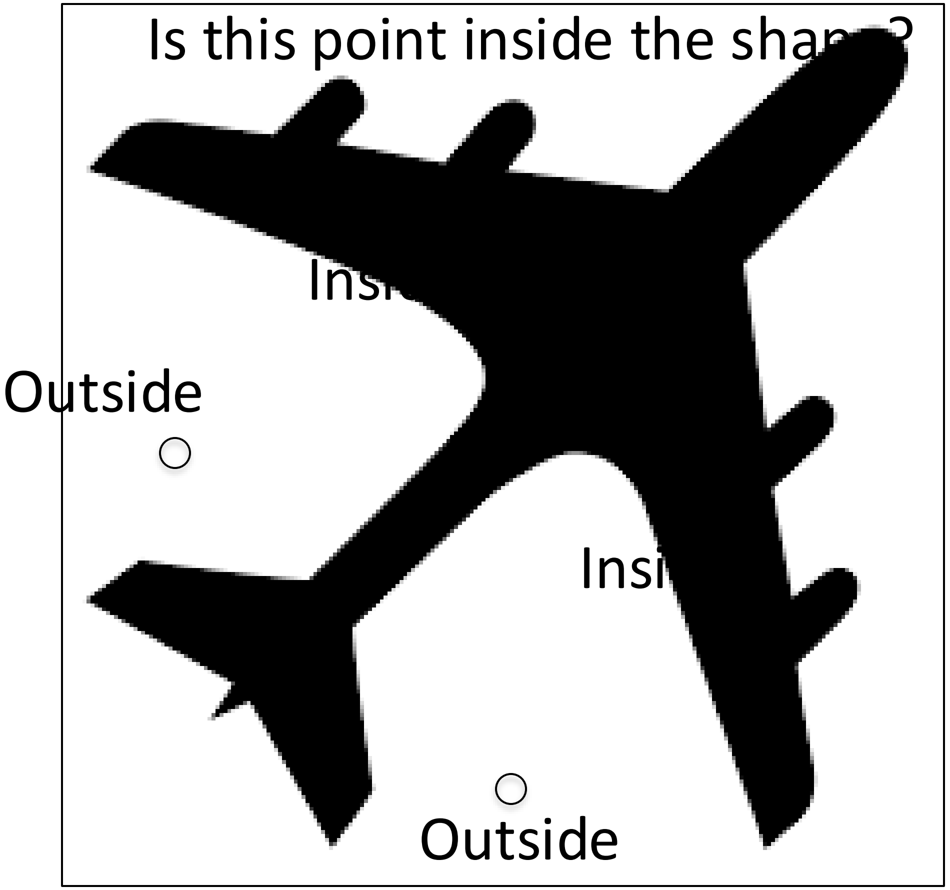
Inside

Outside

Inside

Outside

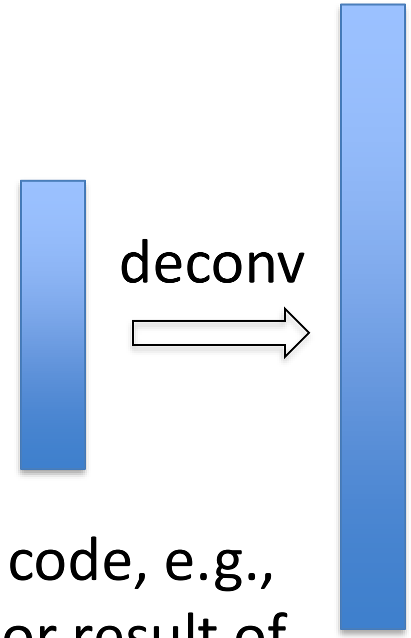
To learn to generate shapes, we should ask …



A typical CNN-based shape generator

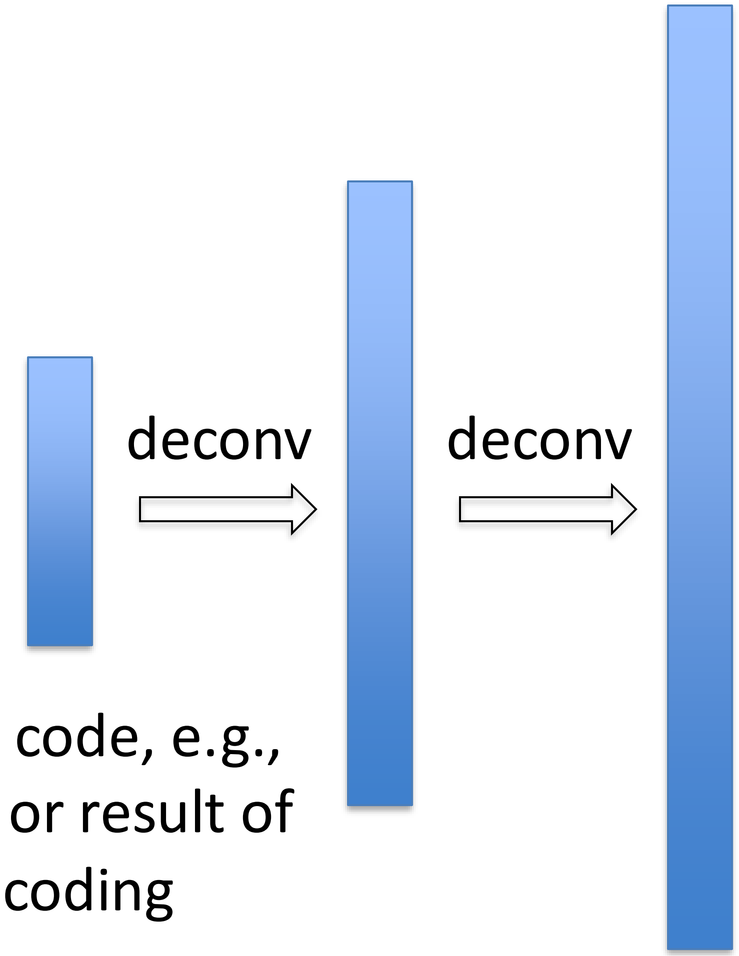
Some code, e.g., noise or result of AE encoding

A typical CNN-based shape generator



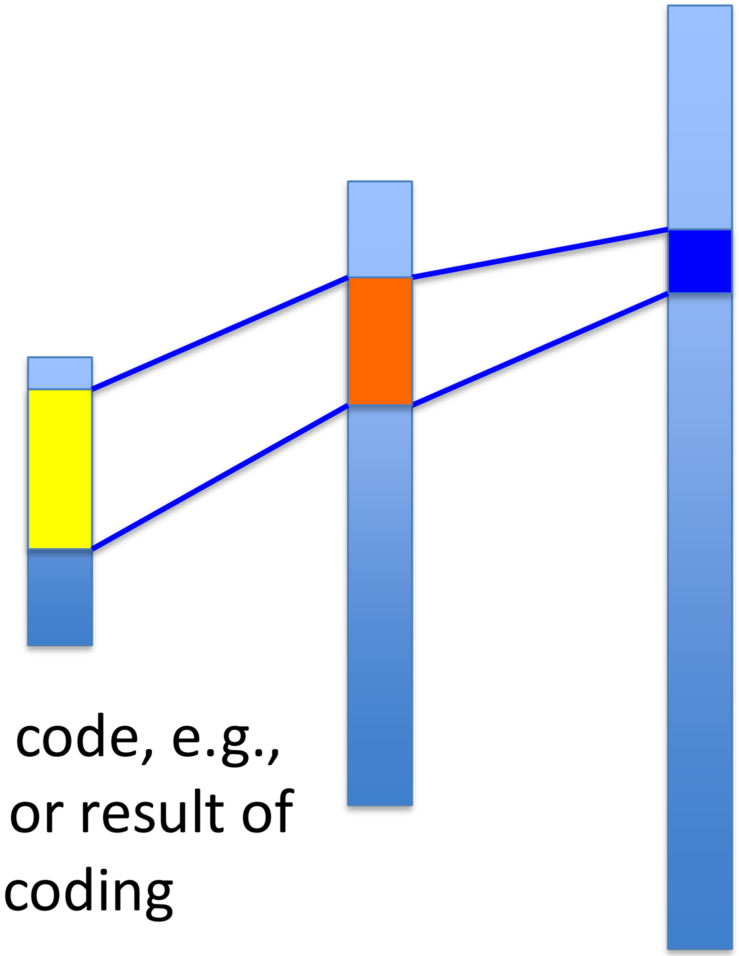
Some code, e.g.,
noise or result of
AE encoding

A typical CNN-based shape generator



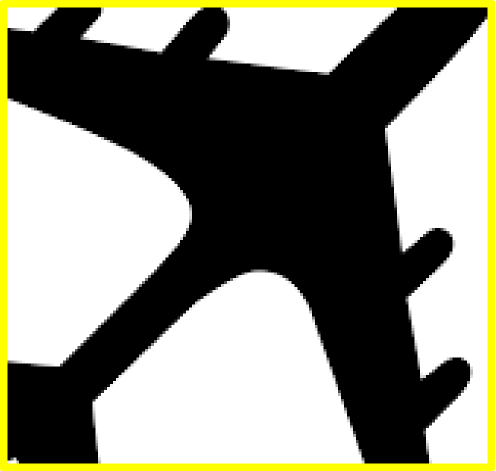
Some code, e.g.,
noise or result of
AE encoding

A mapping from features to voxel values



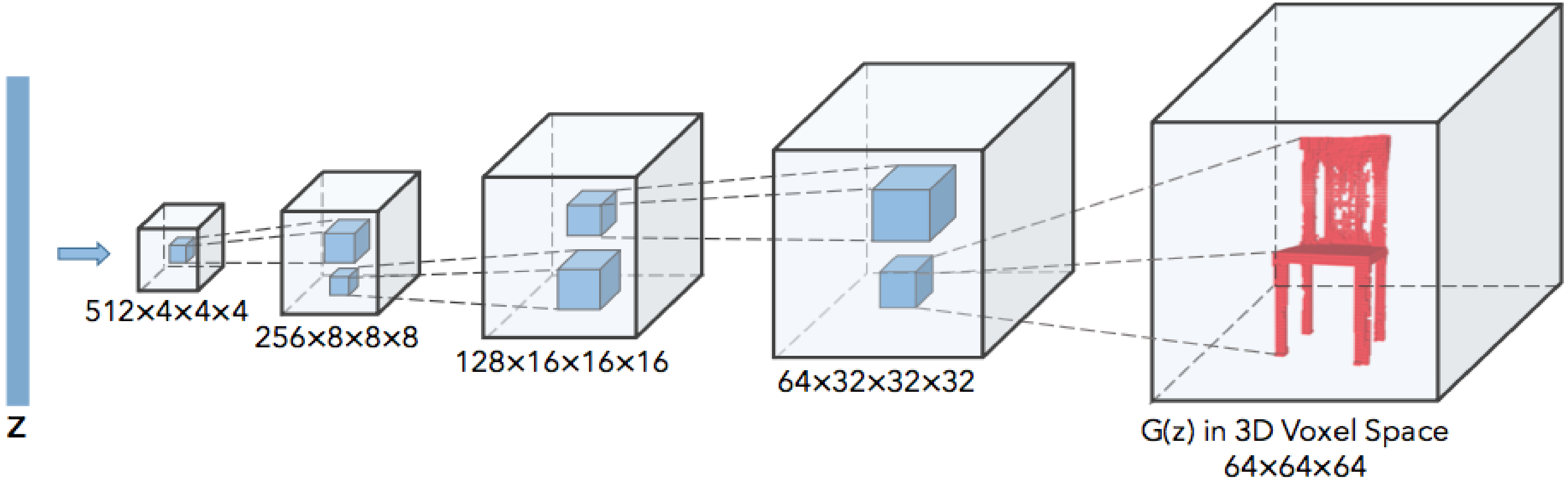
= *f*( ) = mapping from some features, e.g., image features (= pixel intensities) via an AE encoding

Some code, e.g., noise or result of AE encoding



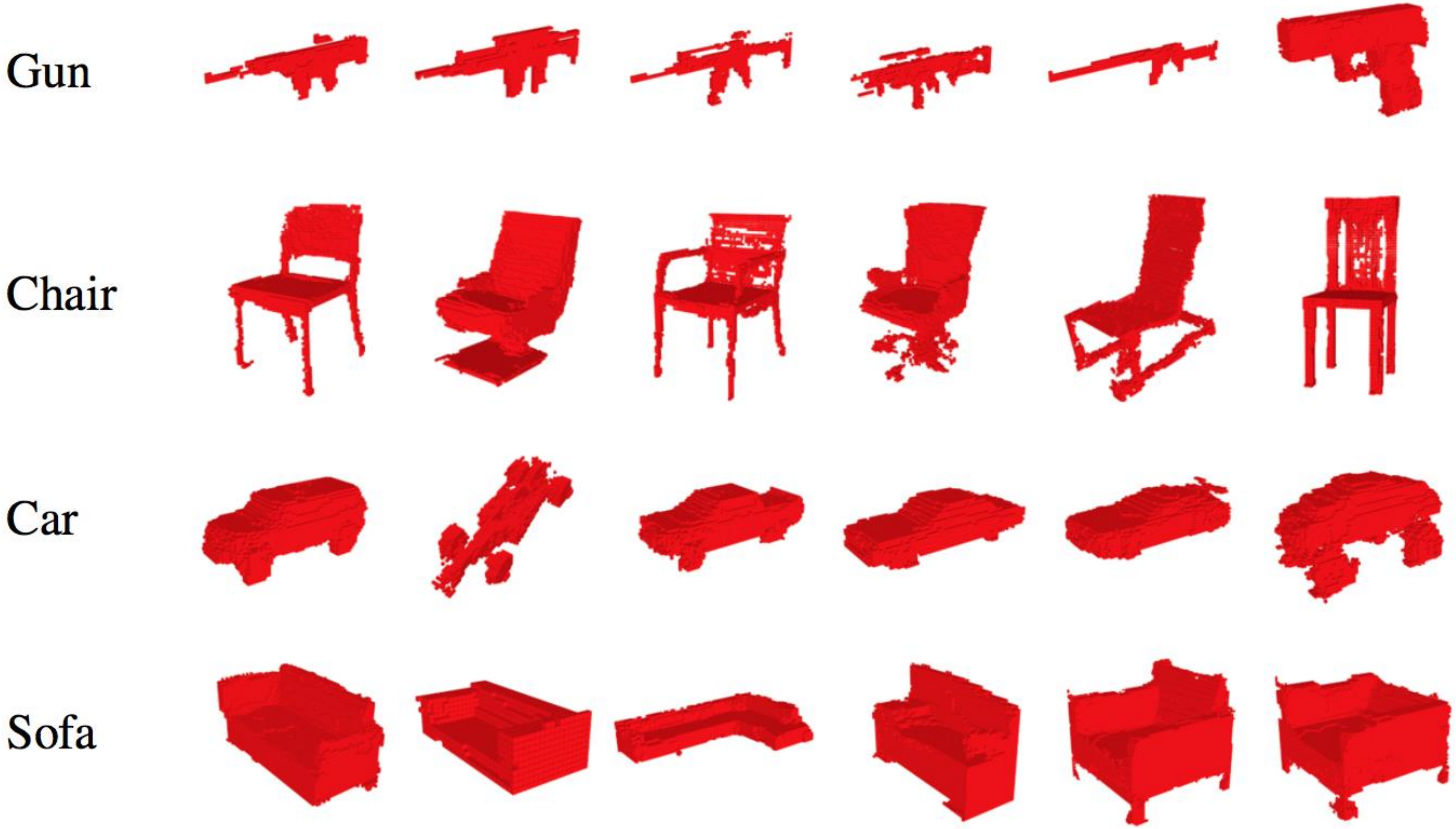
3D generative adversarial network (3D-GAN)

- 3D shape as voxels: combine volumetric CNN and GAN
- Generative network maps 200d vector to 64^3 volume



3D-GAN [Wu et al. NIPS 2016]

Shape generation results by 3D-GAN



3D volumetric shapes generated from random latent vectors

Let us learn the right mapping …

$\boxed{f(p, S) \text{ : is point } p \text{ inside or outside shape } S}$

Let us learn the right mapping …

$f(p, S)$: is point p inside or outside shape S

e.g.,

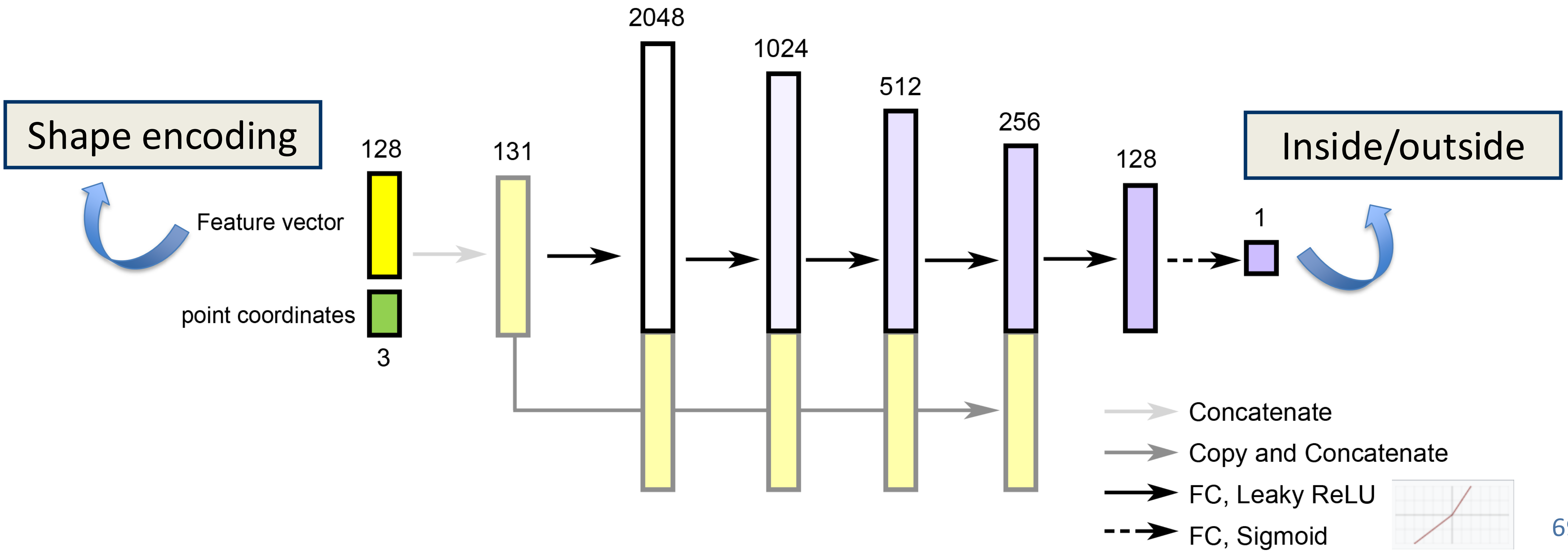
$f(p, S) = 1$, if p is outside S

$= 0$, otherwise

- This is an implicit representation: a shape is composed of the set of all points satisfying an equation $f(x) = 0$
- Point p can be in R^3, so it is a continuous representation

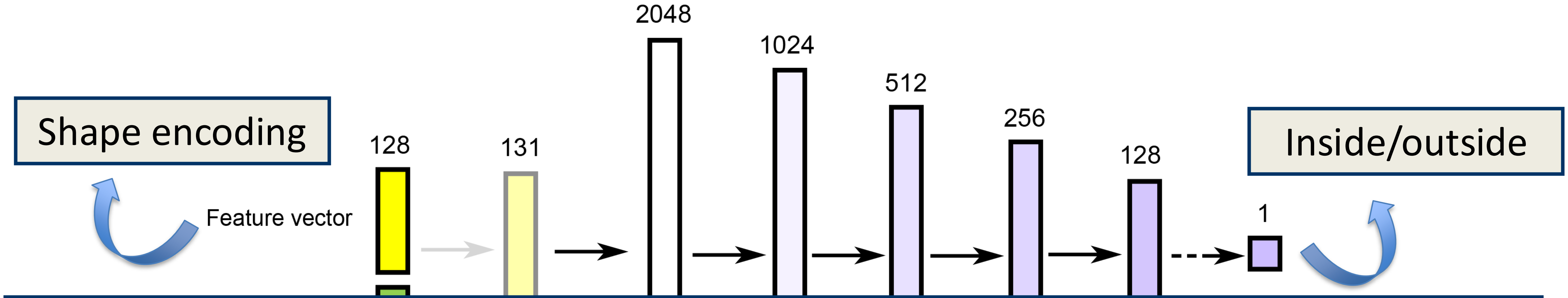
IM-NET: an implicit field generator

- Learn mapping from a 3D point (x, y, z) to inside/outside status with respect to a 3D shape



IM-NET: an implicit field generator

- Learn mapping from a 3D point (*x*, *y*, *z*) to inside/outside status with respect to a 3D shape



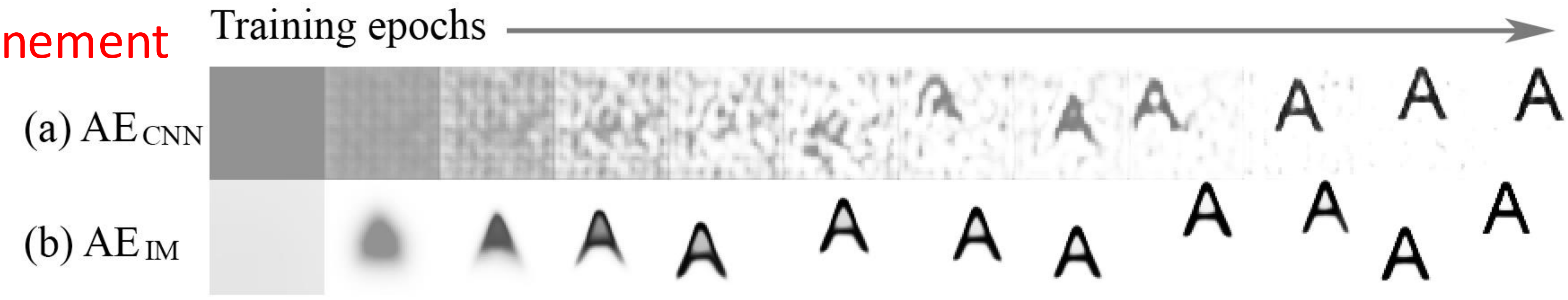
IM-NET: trained on point-value pairs and learns shape boundaries

Traditional CNN-based decoders learn feature-to-voxel mappings.
The features are voxel intensity distributions inside various boxes

CNN vs. IM-NET for shape generations

- CNN vs. implicit decoders on learning to generate A's
- Networks were trained on same letter shape A, with white background, in different locations

Image intensity refinement

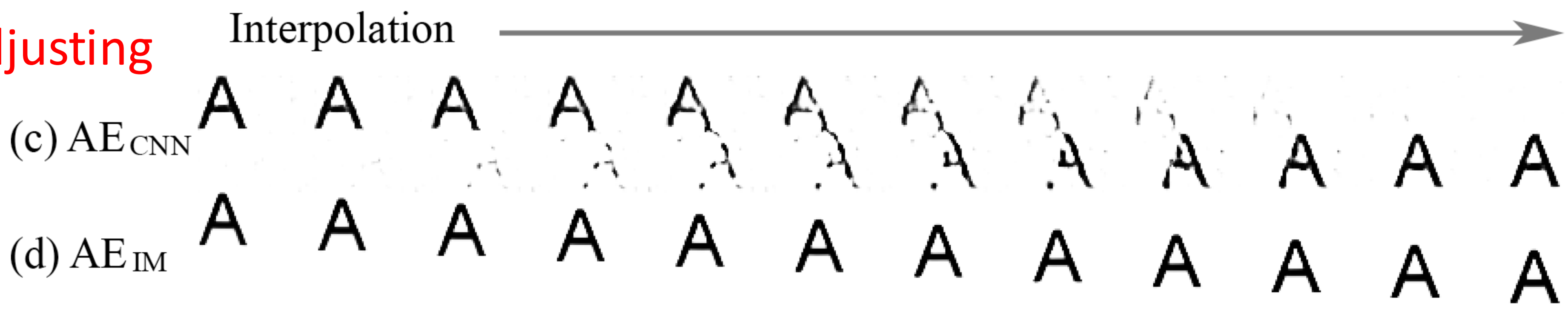


Shape boundary refinement

CNN vs. IM-NET for shape interpolation

- Networks were trained in the same way
- In-between results were generated from linearly interpolated latent codes between source and target

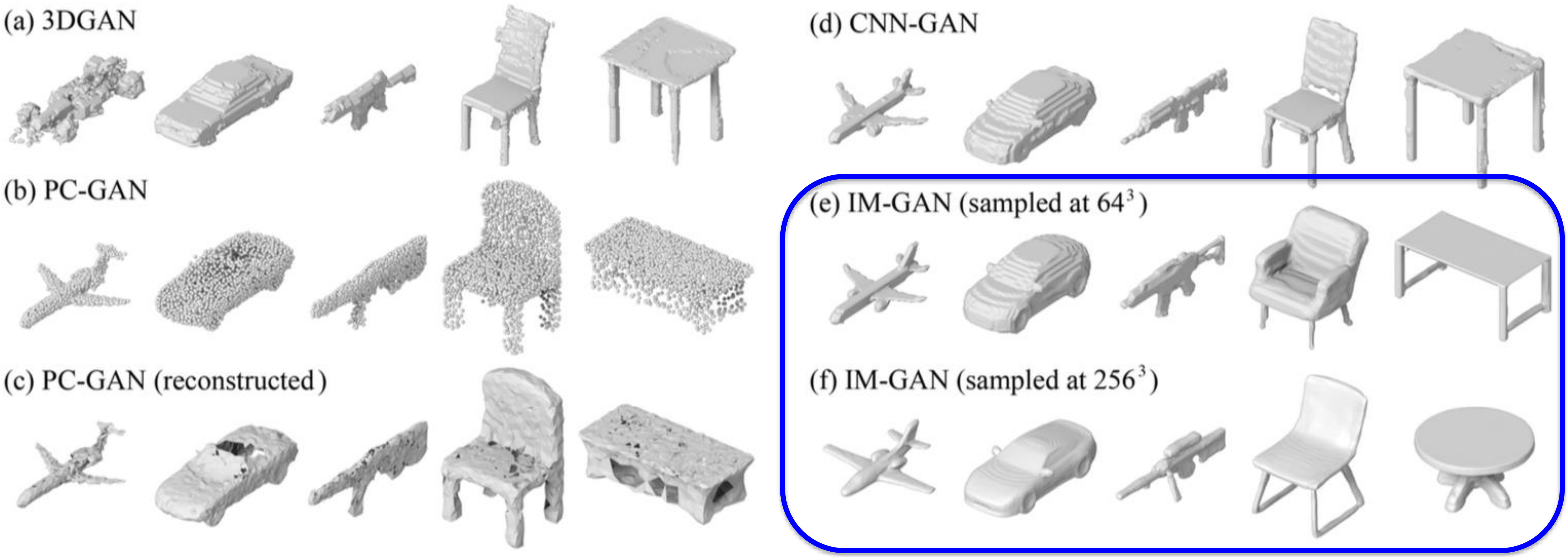
Interpolate by adjusting image intensities



Interpolate by moving the shapes: the right way

Comparing 3D shape generation results

[Chen and Zhang, CVPR 2019]



Comparing 3D shape interpolation results

[Chen and Zhang, CVPR 2019]

2019: the start of neural implicits

Learning Implicit Fields for Generative Shape Modeling

Zhiqin Chen, Hao Zhang

(Submitted on 6 Dec 2018 (v1), last revised 5 Apr 2019 (this version, v3))

Occupancy Networks: Learning 3D Reconstruction in Function Space

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, Andreas Geiger

(Submitted on 10 Dec 2018)

DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, Steven Lovegrove

(Submitted on 16 Jan 2019)

Deep Level Sets: Implicit Surface Representations for 3D Shape Inference

Mateusz Michalkiewicz, Jhony K. Pontes, Dominic Jack, Mahsa Baktashmotlagh, Anders Eriksson

(Submitted on 21 Jan 2019)

Learning Shape Templates with Structured Implicit Functions

Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T. Freeman, Thomas Funkhouser

(Submitted on 12 Apr 2019)