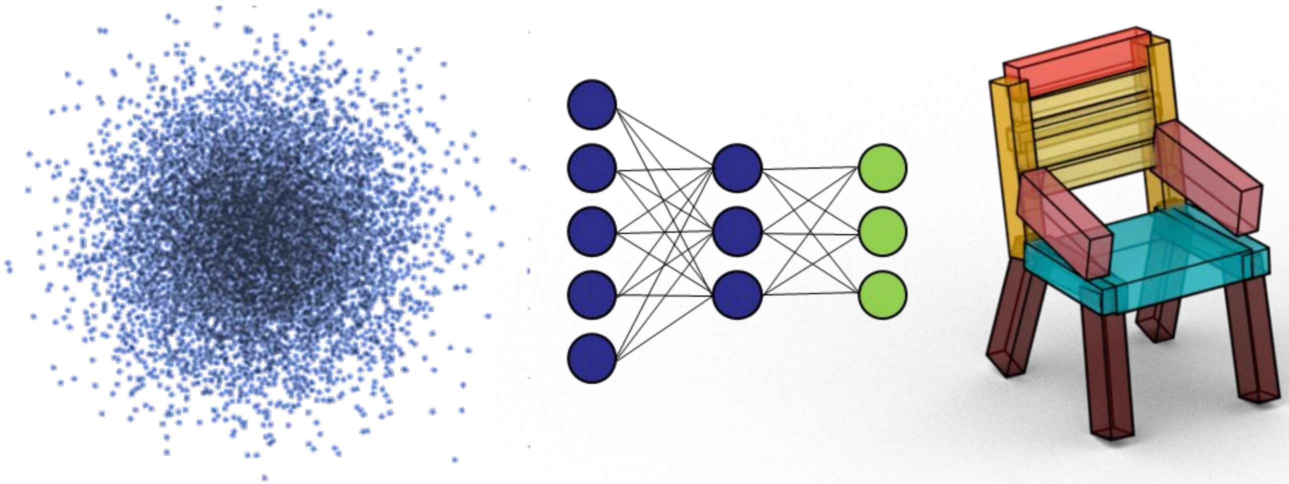
CS348n: Neural Representations and Generative Models for 3D Geometry



Leonidas Guibas
Kaichun Mo (TA / Lecturer Today)
Computer Science Department
Stanford University

Leonidas Guibas Laboratory
Geometric Computing

01-10_DEEP_ARCH

Recap: Class Logistics

Immediate ToDos

- Sign up for Piazza
  - https://piazza.com/stanford/winter2022/cs348n
- Sign up for your class presentation session:
  - Google form: https://forms.gle/xNzWptSzfngzmuGs7
  - **Deadline: Jan 11, noon PDT** – otherwise we will randomly assign you
  - You can sign up as a team (at max 3 students per team)
  - One team per day, covering all the required papers
- For access to class lecture slides:
  - Use credentials:
    - user: neural
    - passwd: creation

Paper Presentations

- Similar to a paper reading group
  - Staff students will give an example on Jan 19
- 2-4 papers form the literature:
  - provide context and relate them to the material in the previous class
  - relate them to each other, if this makes sense
  - discuss:
    - the problem being solved and its significance
    - the method(s) used
    - the evaluation(s) used and your assessment of the paper's merits and drawbacks
- Can work as teams of up to three students
- 25 mins total time

Course Project

- Homeworks address the “how” – learning specific methods
- Project addresses the “what” – to imagine what you can do with the class tools

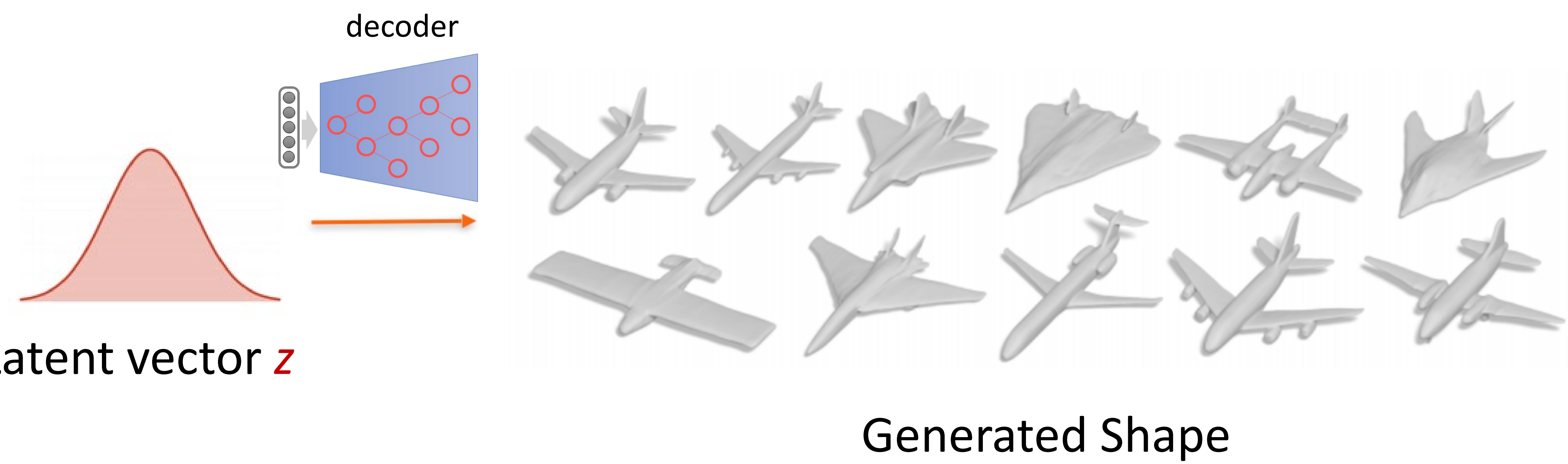
- Can be an extension of one of the homeworks (we’ll provide some suggestions)
- Can be related to or useful for some other research you are doing
- Can work in groups of up to three students

- Need to turn in a group write up and give a brief demo at the end of the class

Recap: 3D Geometry Representations

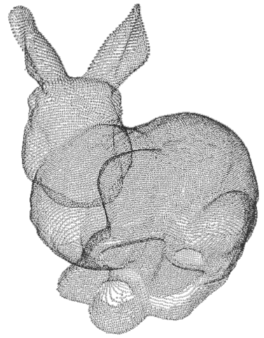
(Last Lecture)

ML Decoding/Generation from Latent Vectors

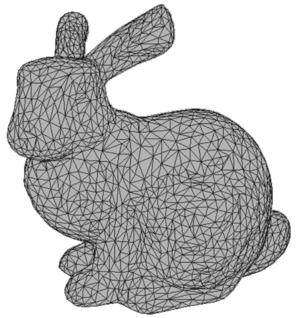


Generator/Decoder: generating shapes from latent vectors via deep networks – but in what format?

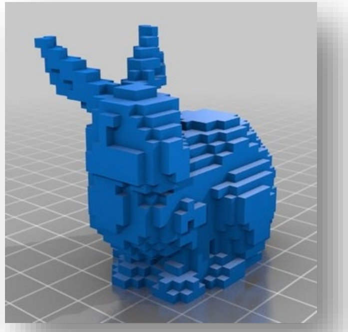
In 3D, There is Representation Diversity



Point Cloud



Mesh

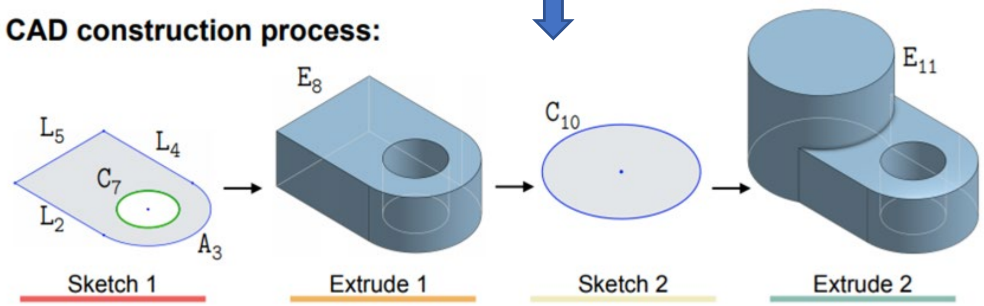


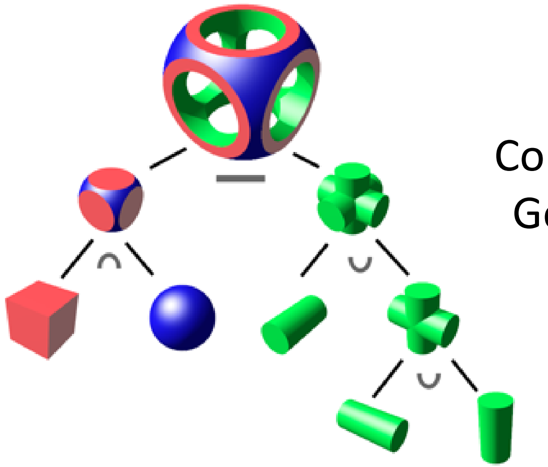
Voxels

Multiple View Images
RGB(D)

These are irregular representations – and the ones most commonly used in 3D apps

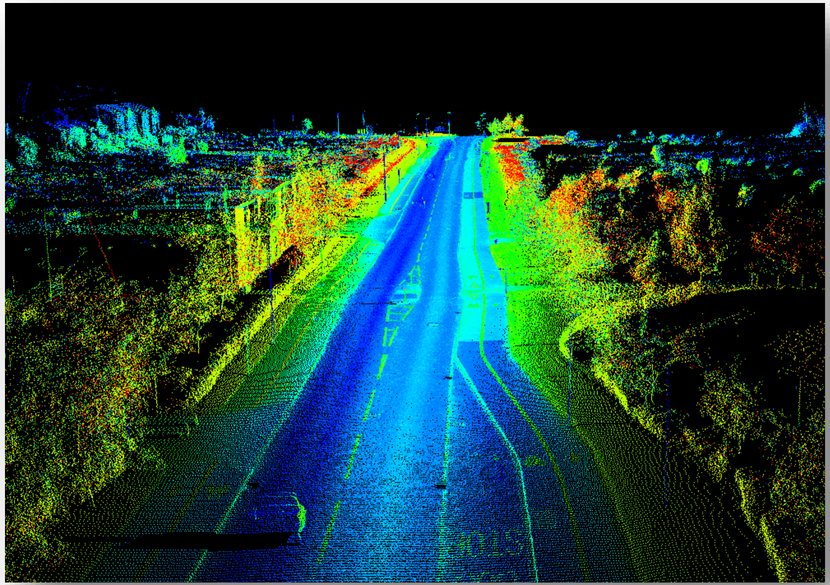
Sketch-Extrude



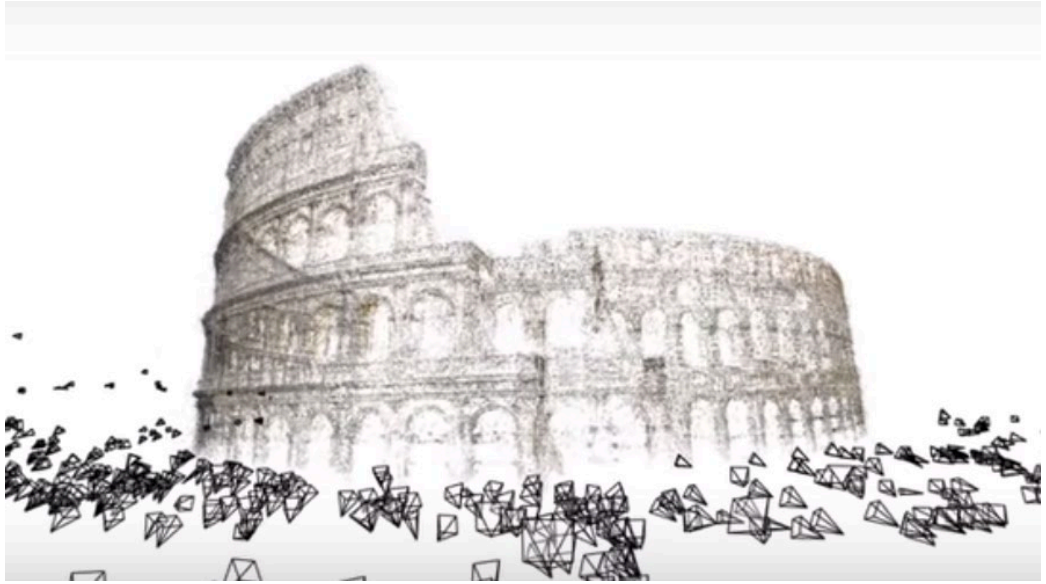


Constructive Solid Geometry (CSG)

3D Point Clouds from Many Sensors



Lidar point clouds (LizardTech)



Structure from motion (Microsoft)

Depth camera (Intel, Microsoft, Google)

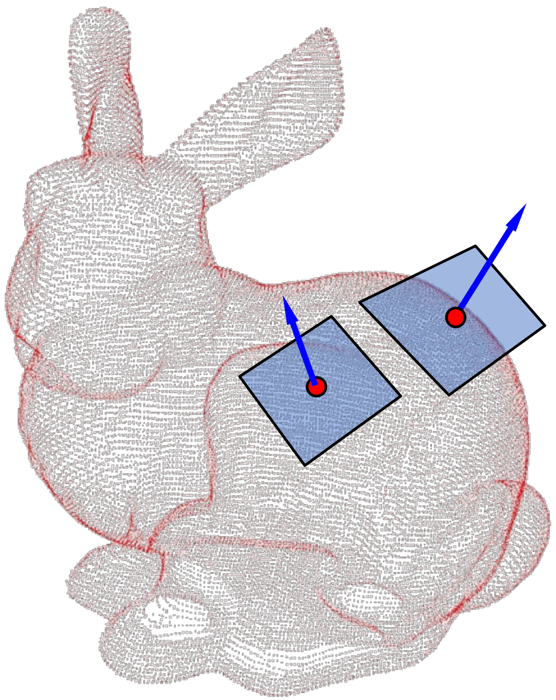
Points			
p0	x0	y0	z0
p1	x1	x1	z1
p2	x2	y2	z2
p3	x3	y3	z3
p4	x4	y4	z4
p5	x5	y5	z5
p6	x6	y6	z6
...	...	...	...

In addition, normals, colors, etc.

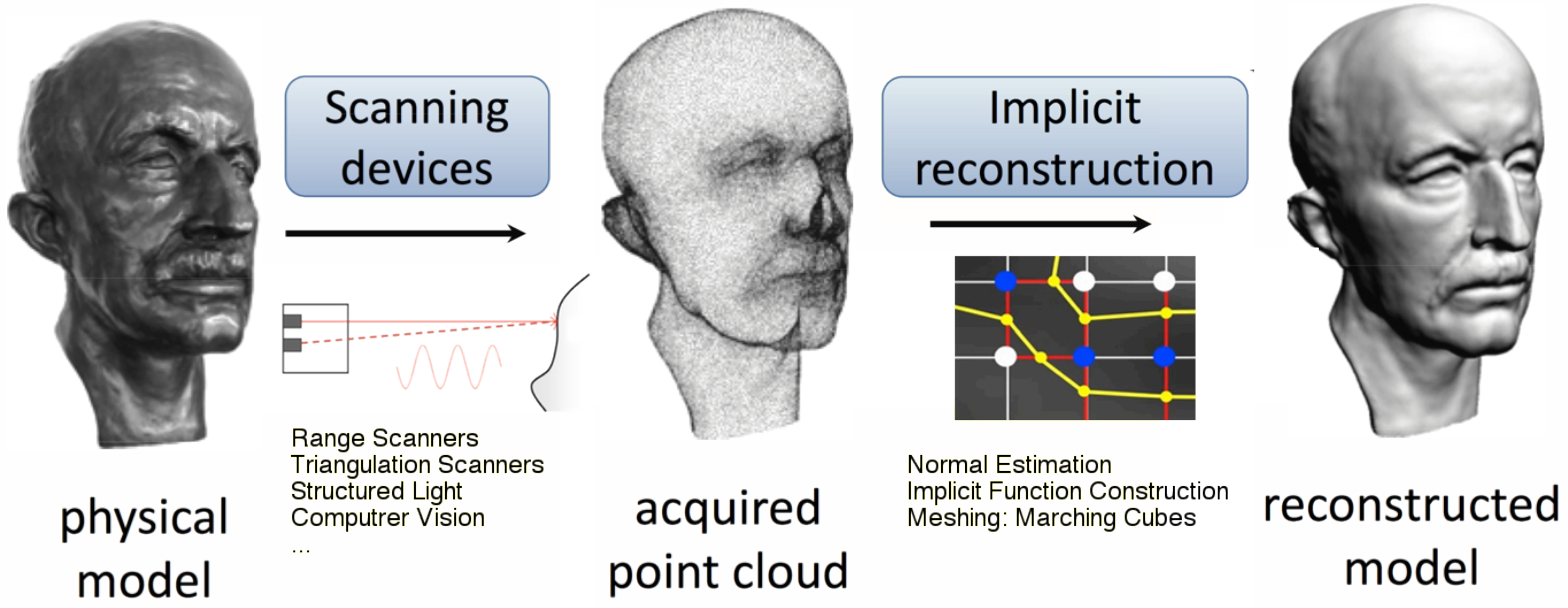
Normal Estimation and Outlier Removal

Fundamental problems in point cloud processing.

Although seemingly very different, can be solved with the same general approach – look at the “shape of neighborhoods” …



3D Point Cloud Processing

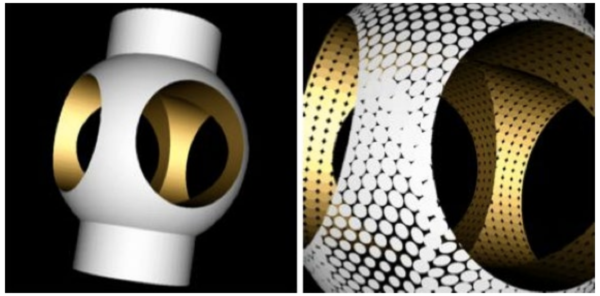


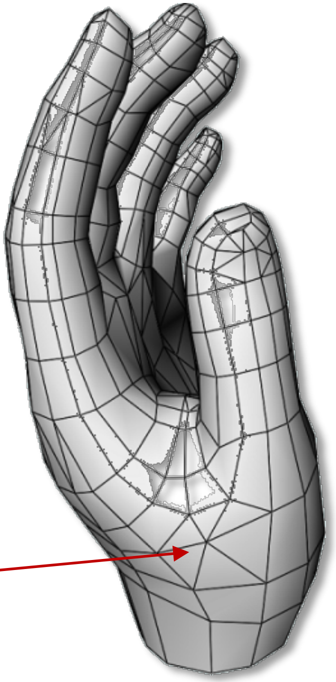
Traditional 3D Acquisition Pipeline

We’ll see how to apply ML directly on Point Cloud Data

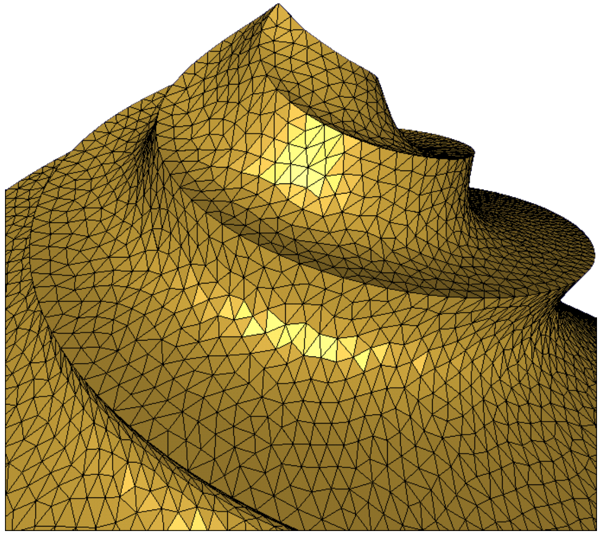
B-Reps: Low-Level Elements

- Triangle meshes
- Quad meshes

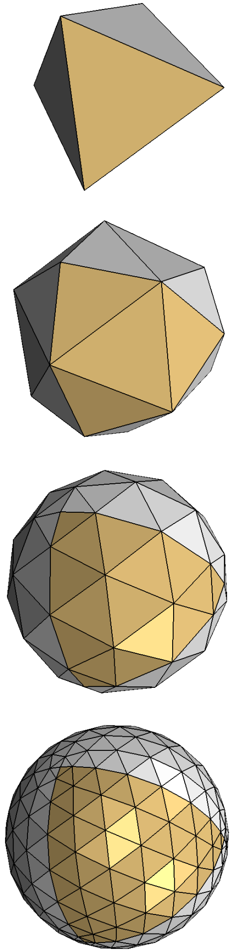




quad meshes can be formed by grid-like quads, but there will almost always be extraordinary (singular) vertices



trade-offs



trade-offs

Simple Data Structures: Triangle List

- Used in formats

  OBJ, OFF, WRL

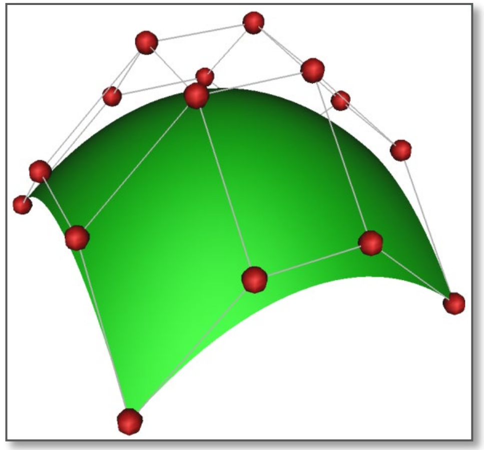
- Storage
  - Vertex: position
  - Face: vertex indices
  - 12 bytes per vertex
  - 12 bytes per face
  - 36**v* bytes for the mesh

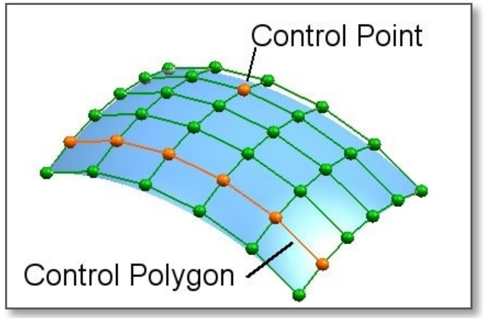
- No explicit neighborhood info

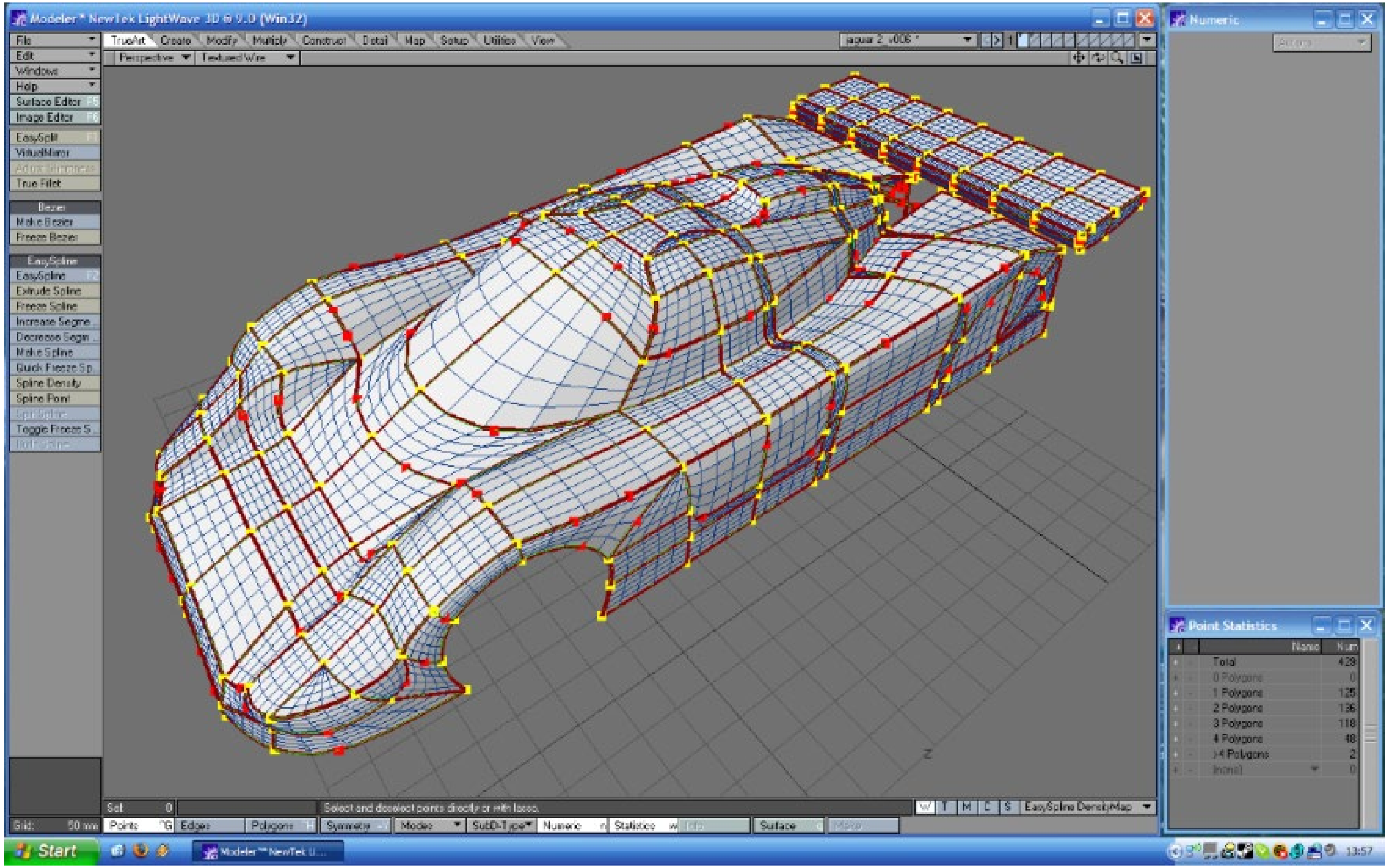
Vertices			
v0	x0	y0	z0
v1	x1	x1	z1
v2	x2	y2	z2
v3	x3	y3	z3
v4	x4	y4	z4
v5	x5	y5	z5
v6	x6	y6	z6
...	...	...	...

Triangles			
t0	v0	v1	v2
t1	v0	v1	v3
t2	v2	v4	v3
t3	v5	v2	v6
...	...	...	...

Parametric Surfaces



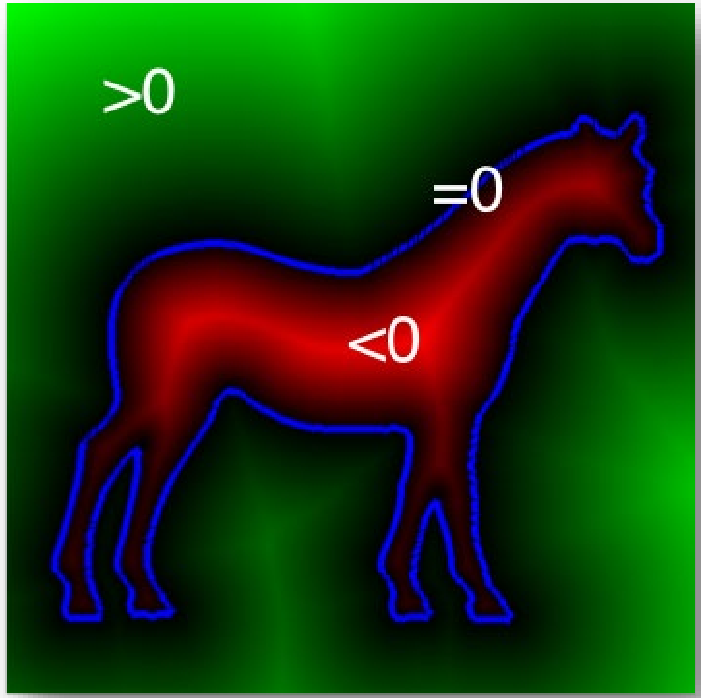




Implicit Curves and Surfaces

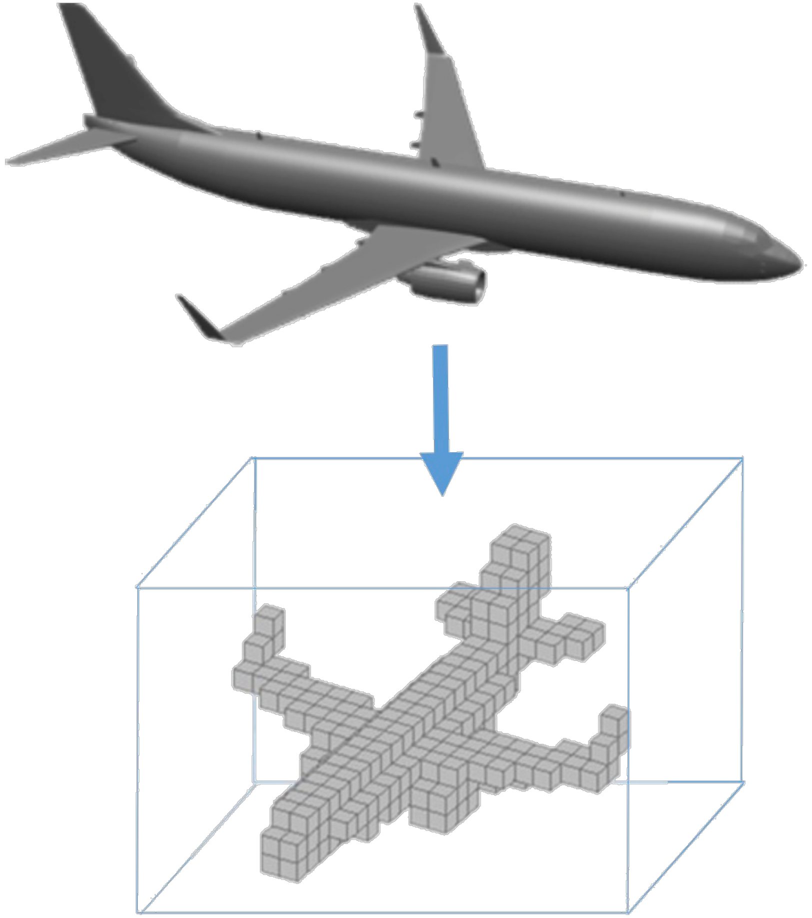
- Kernel of a scalar function $f : \mathbb{R}^m \to \mathbb{R}$
  - Curve in 2D: $S = \{x \in \mathbb{R}^2 | f(x) = 0\}$
  - Surface in 3D: $S = \{x \in \mathbb{R}^3 | f(x) = 0\}$

- Zero level set of signed distance function



V-Rep: Volumetric Grids

- Binary volumetric grids
- Can be produced by thresholding the distance function, or from the scanned points directly
- N^3 gets expensive fast



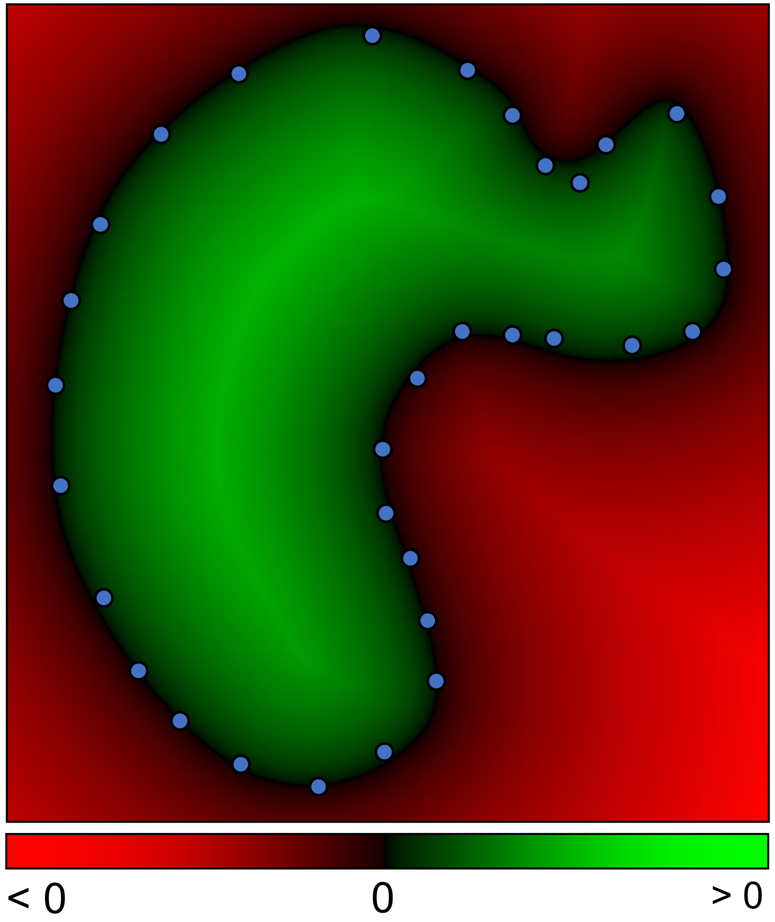
Also represents space of little informational value

Representation Conversion: Points --> Implicit

- Given a point cloud
- Define a function $f : R^3 \rightarrow R$

with value > 0 outside the shape and < 0 inside

Example: signed distance function (SDF) to the shape surface

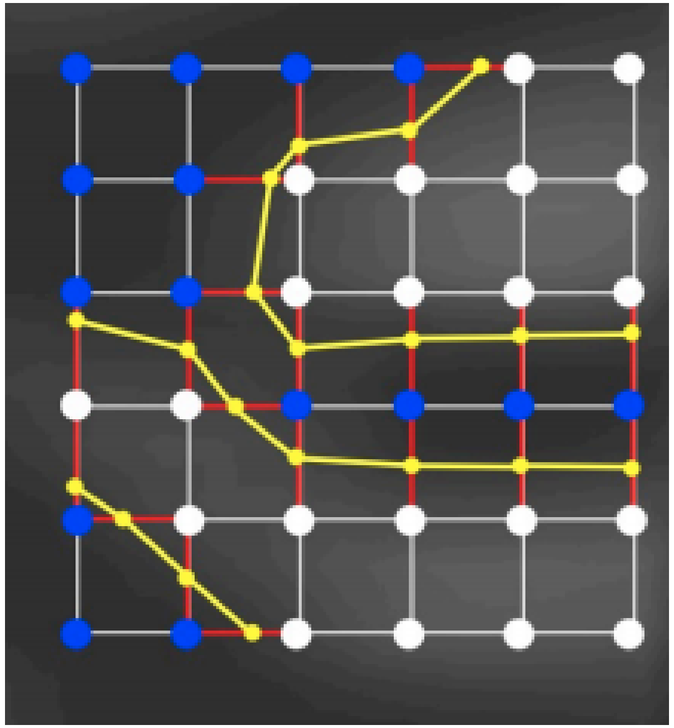


Representation Conversion: Implicit --> Mesh

Given a function: $f(x)$

- $f(\mathbf{x}) < 0$ inside
- $f(\mathbf{x}) > 0$ outside

1. Discretize space.
2. Evaluate $f(x)$ on a grid.
3. Classify grid points (+/-)
4. Classify grid edges
5. Compute intersections
6. Connect intersections

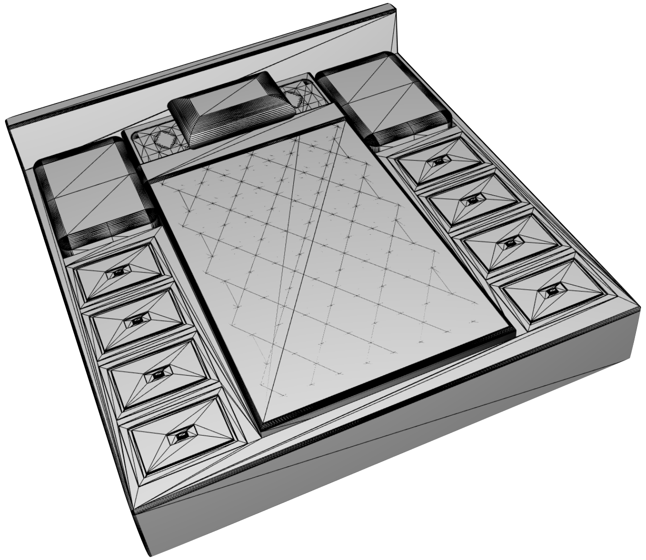


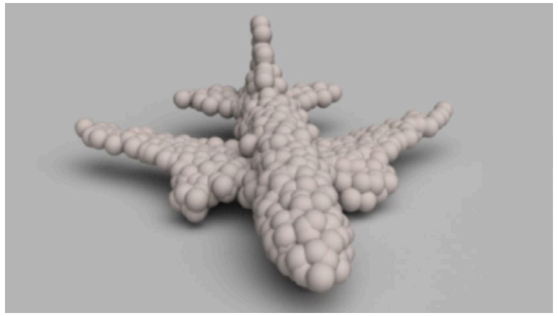
Marching Squares (2D)
Marching Cubes (3D)

Representation Conversion: Mesh --> Points

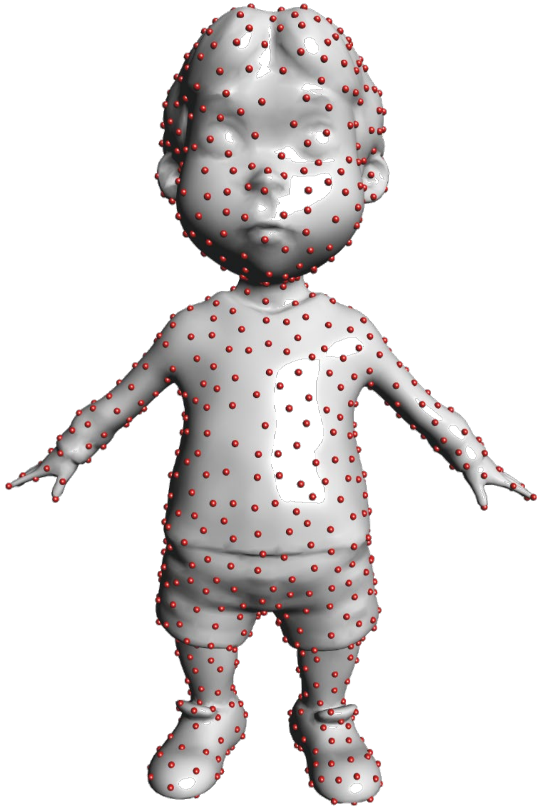
- Points are simple but expressive!
  - Few points can suffice
- Flexible, unstructured, few constraints
- Also: ML applications!

Furthest Point Sampling (FPS)



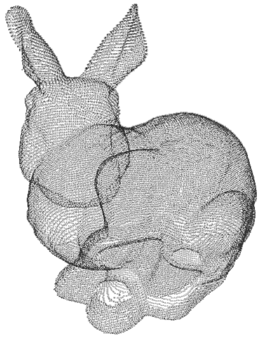


CAD meshes:
many components
bad triangles
connectivity problems

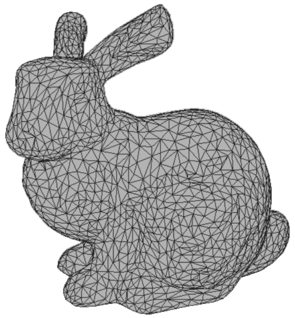


the problem:
sampling the mesh

Today: DL / NNs for Images and 3D Voxel Data



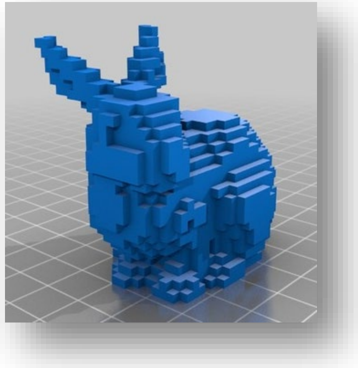
Point Cloud



Mesh

These are irregular representations – and the ones most commonly used in 3D apps

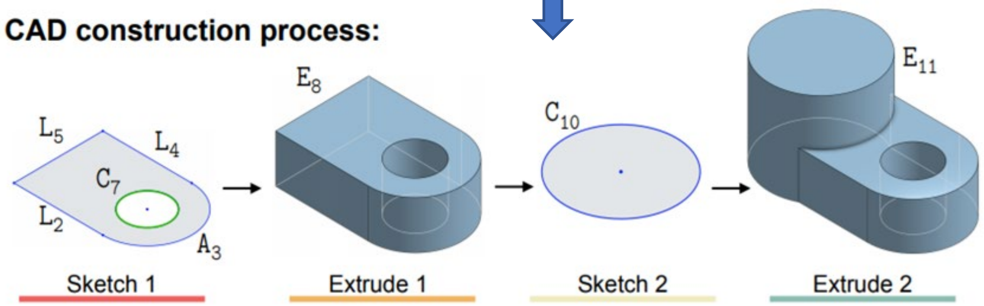
Today!!!

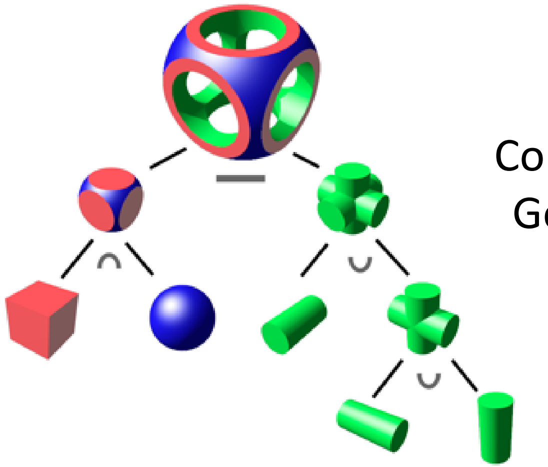


Voxels

Multiple View Images
RGB(D)

Sketch-
Extrude





Constructive Solid Geometry (CSG)

Brief Review: ML, DL, Deep Nets, CNNs, Transformers

(get prepared with DL and NN basics)

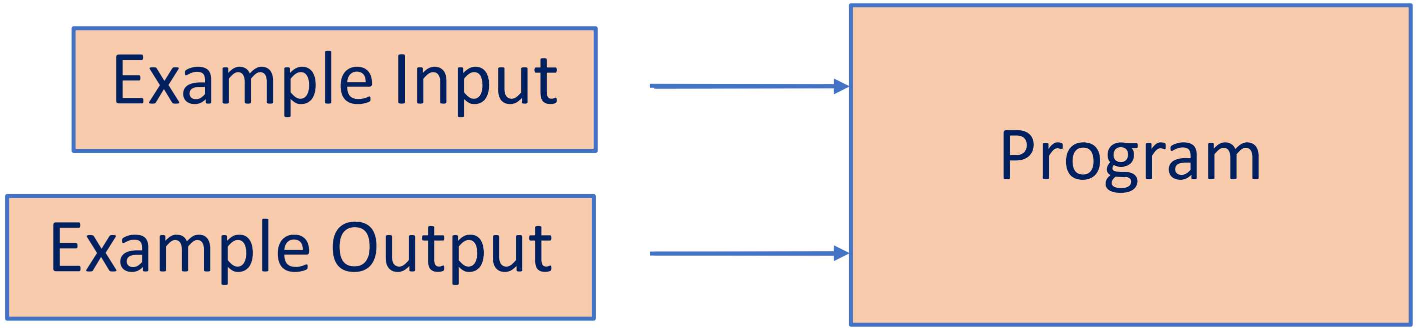
Machine Learning

- Traditional Programming

Machine Learning

- Traditional Programming

- Machine Learning

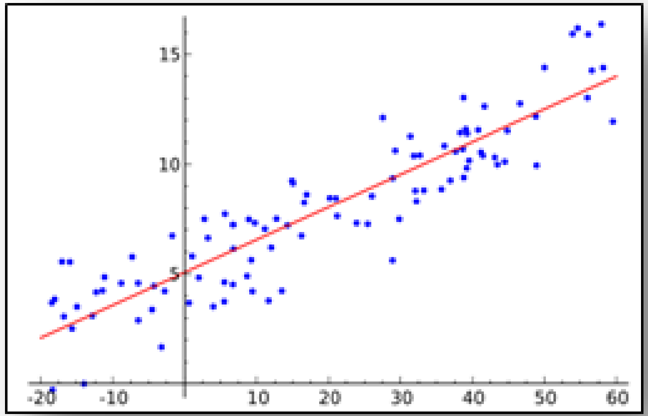


Machine Learning

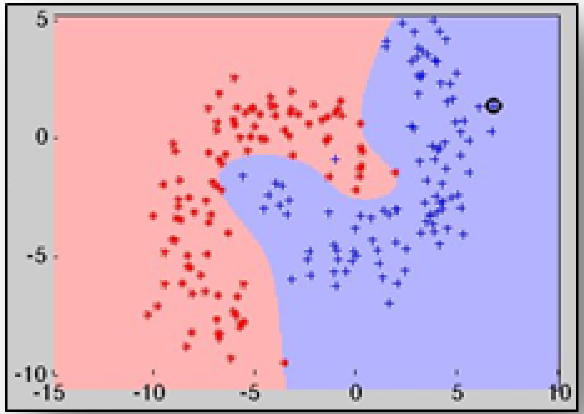
f()=cat

Parametrized by many
learnable parameters

Model Fitting



Regression

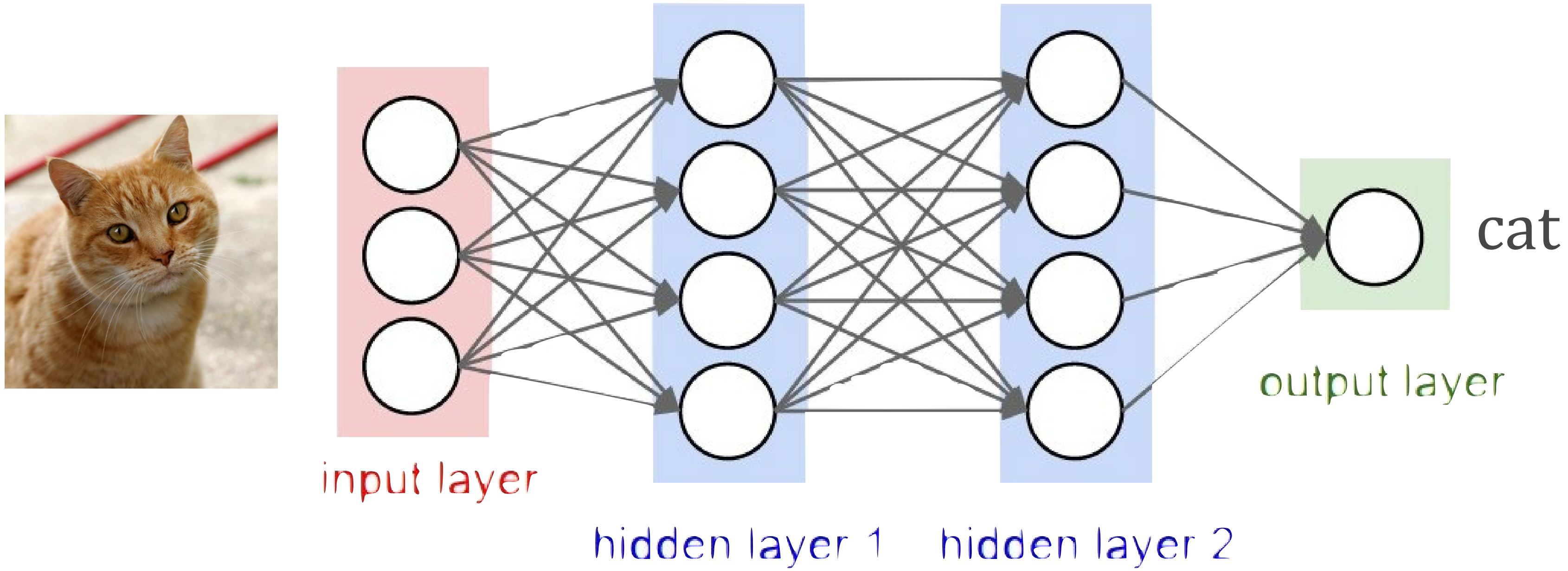


Classification

Deep Learning

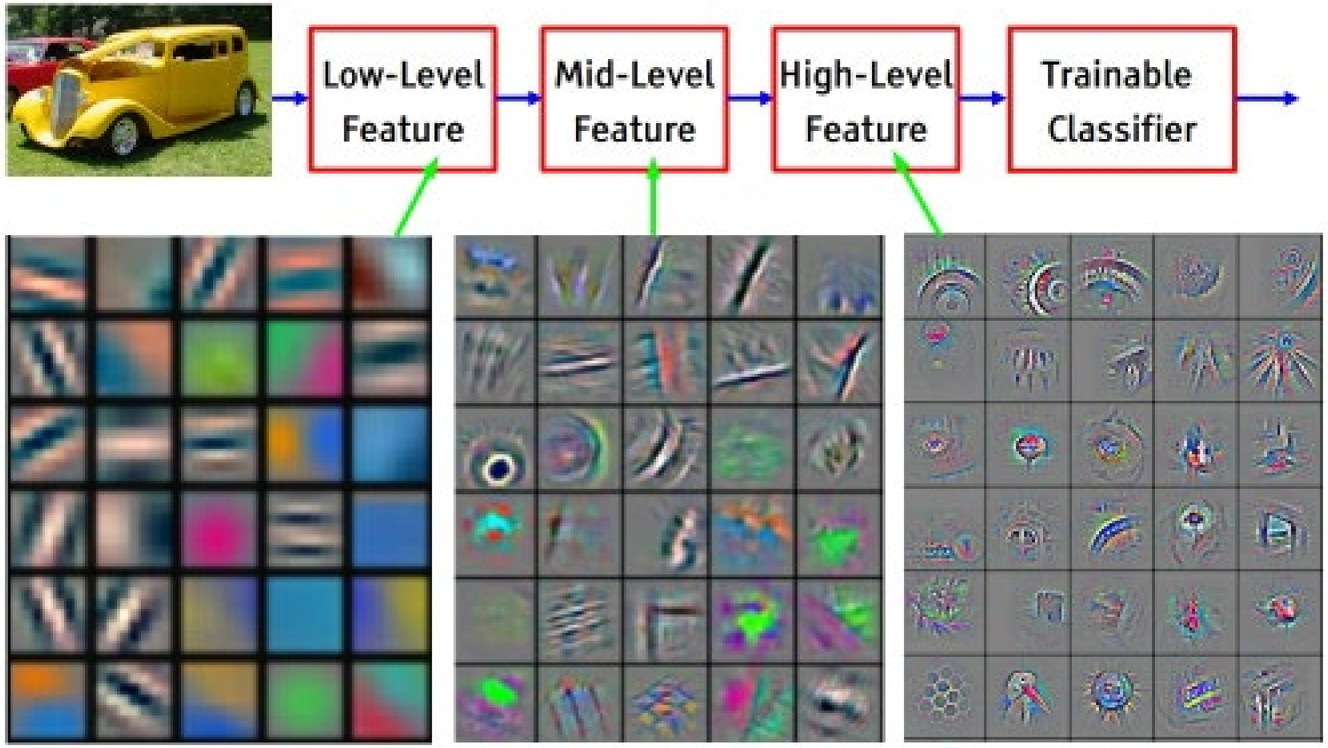
- Deep learning allows computational models that are composed of **multiple processing layers** **to learn representations of data** with **multiple levels of abstraction**.

Deep Learning by Y. LeCun et al. Nature 2015



Neural Networks as Feature Extractors

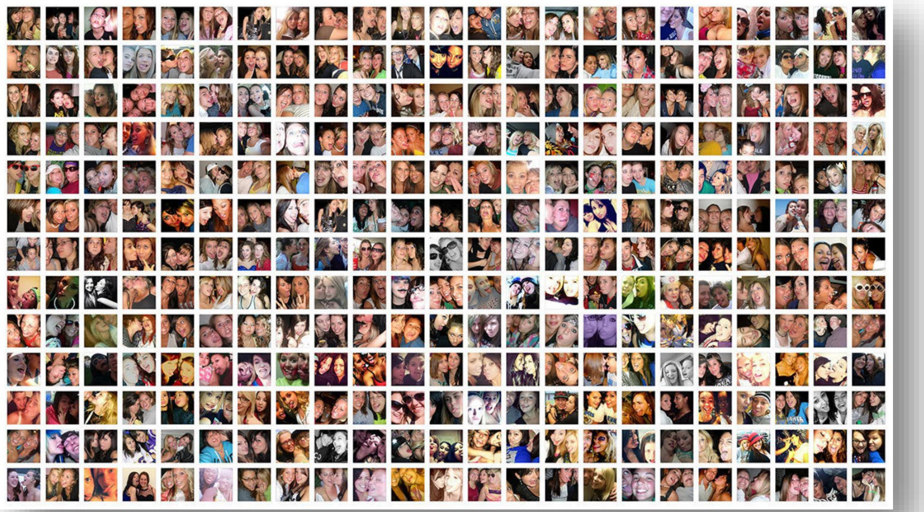
- Neural networks extract powerful features from data



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Image Credits: Yan LeCun

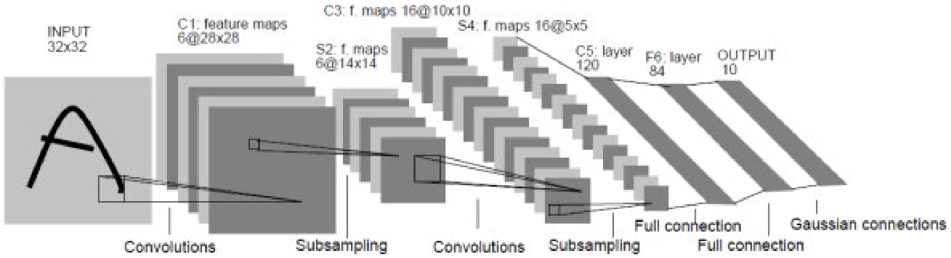
Deep Learning: Powered By

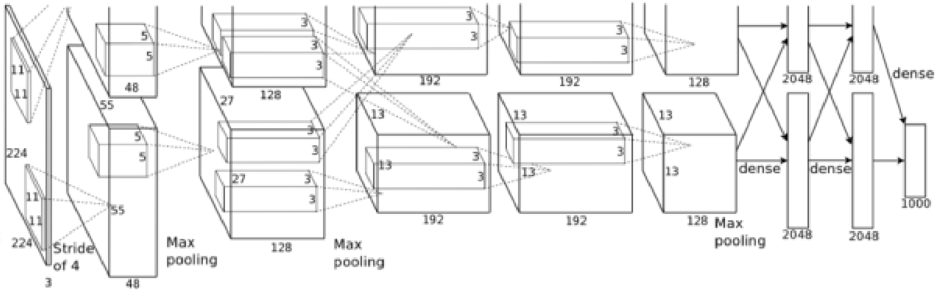


Lots of data

Lots of computing power

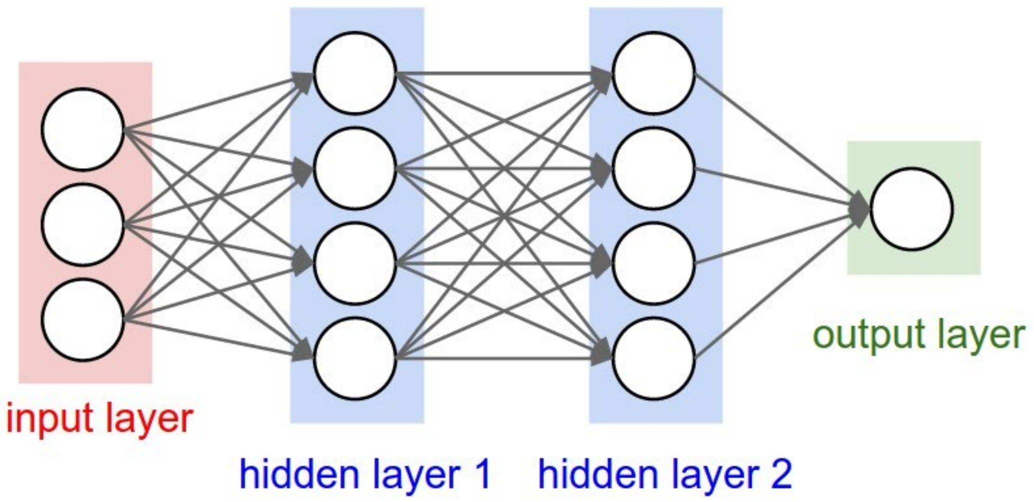
+





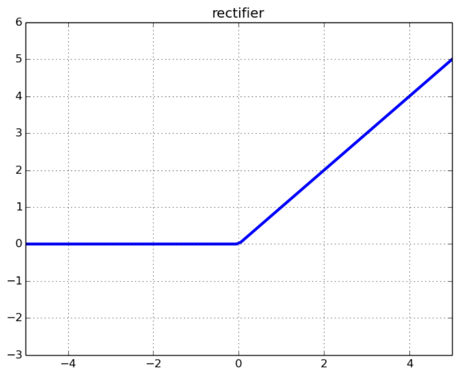
Powerful Neural Network Architectures

Neural Networks



f: non-linear activation function

$$\mathrm{ReLU}(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases}$$



Model: Multi-Layer Perceptron (MLP)

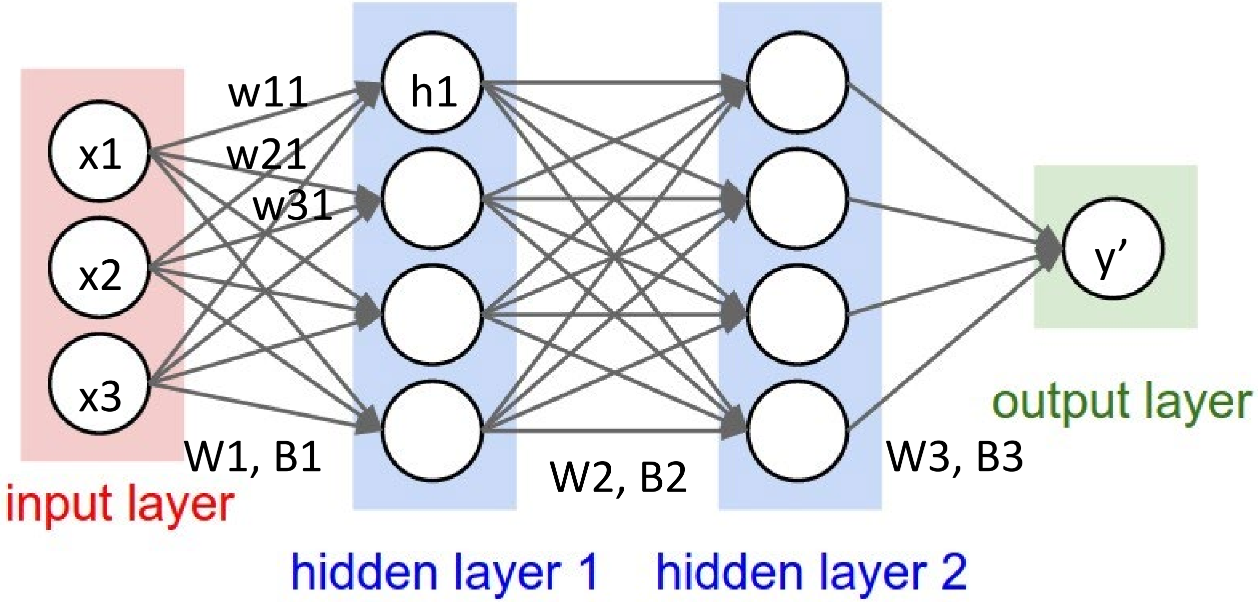
$$y' = W_3 f(W_2 f(W_1 x + b_1) + b_2) + b_3$$

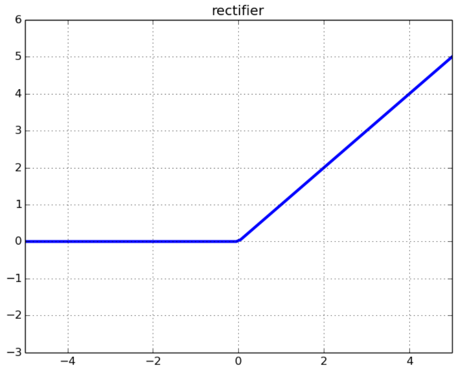
Neural Networks

h1=f(w11 * x1 + w21 * x2 + w31 * x3 + b1)

f: non-linear activation function

$$\mathrm{ReLU}(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases}$$





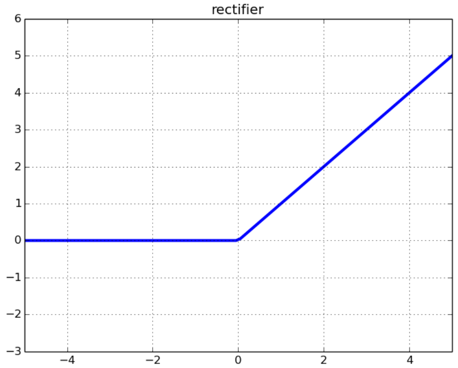
Model: Multi-Layer Perceptron (MLP)

$$y' = W_3 f(W_2 f(W_1 x + b_1) + b_2) + b_3)$$

Neural Networks

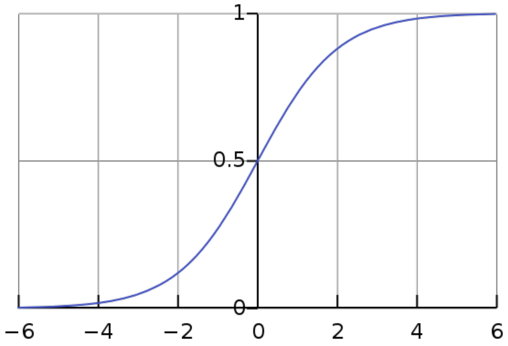
f: non-linear activation function

$$\text{ReLU}(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases}$$

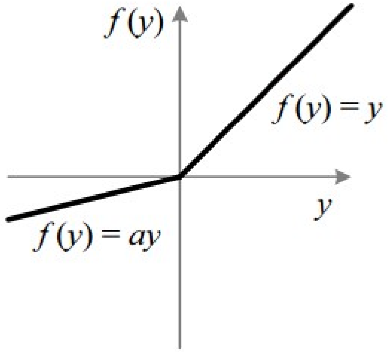


Sigmoid Function

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}.$$



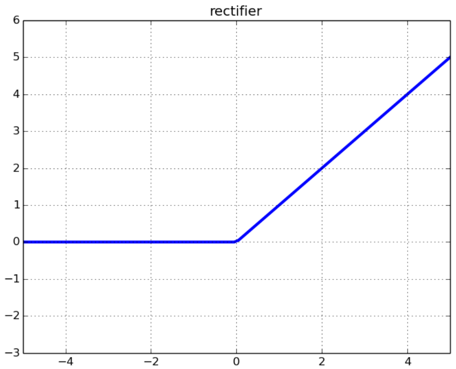
Leaky ReLu

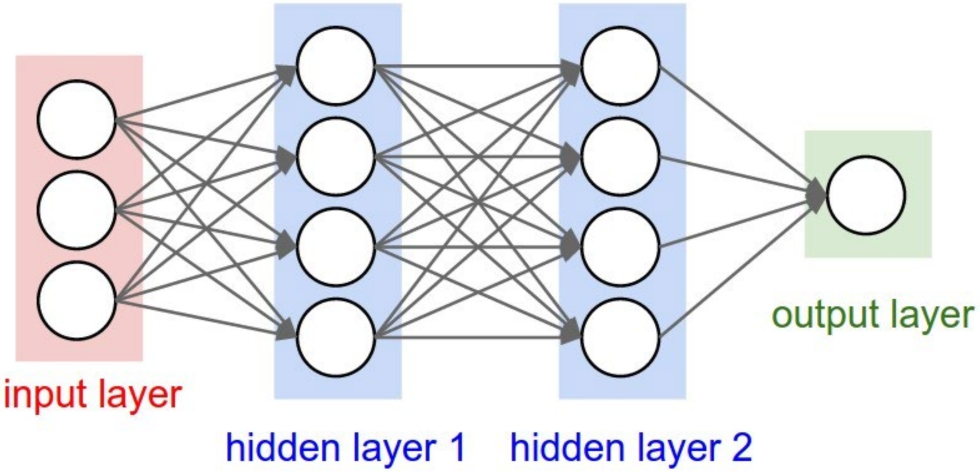


Neural Networks

f: non-linear activation function

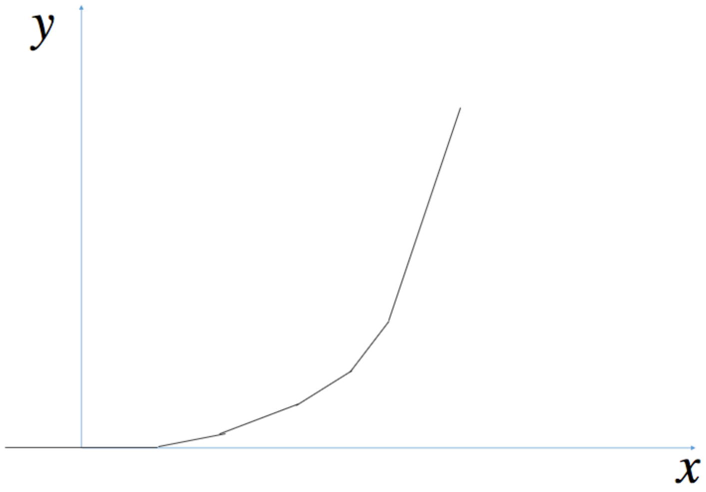
$$\mathrm{ReLU}(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases}$$





Piece-wise Approximation

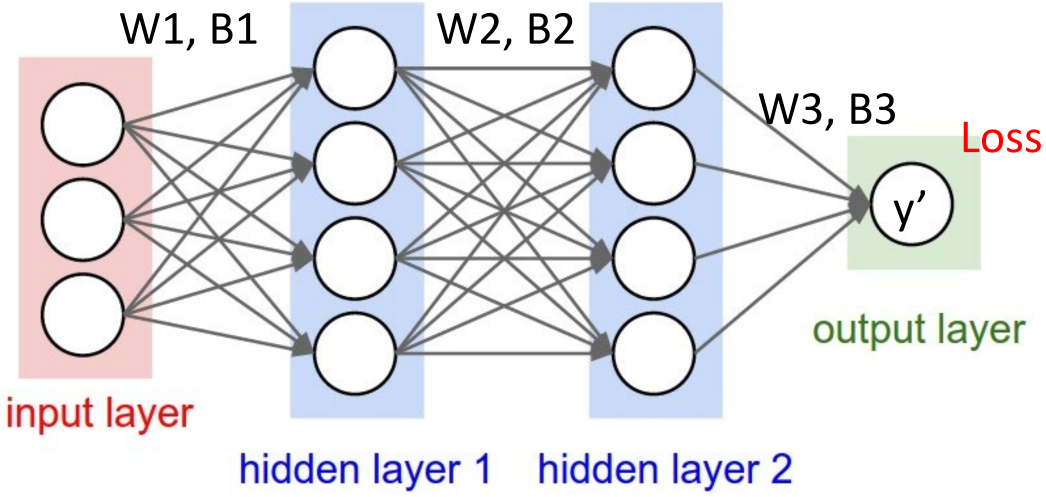
y = x^2



Neural Networks

W = {W1, W2, W3, B1, B2, B3}

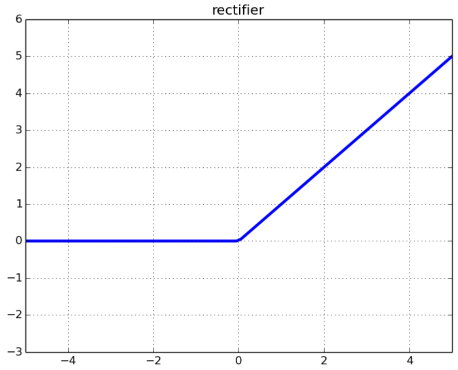
x



y

f: non-linear activation function

$$\mathrm{ReLU}(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases}$$



Model: Multi-Layer Perceptron (MLP)

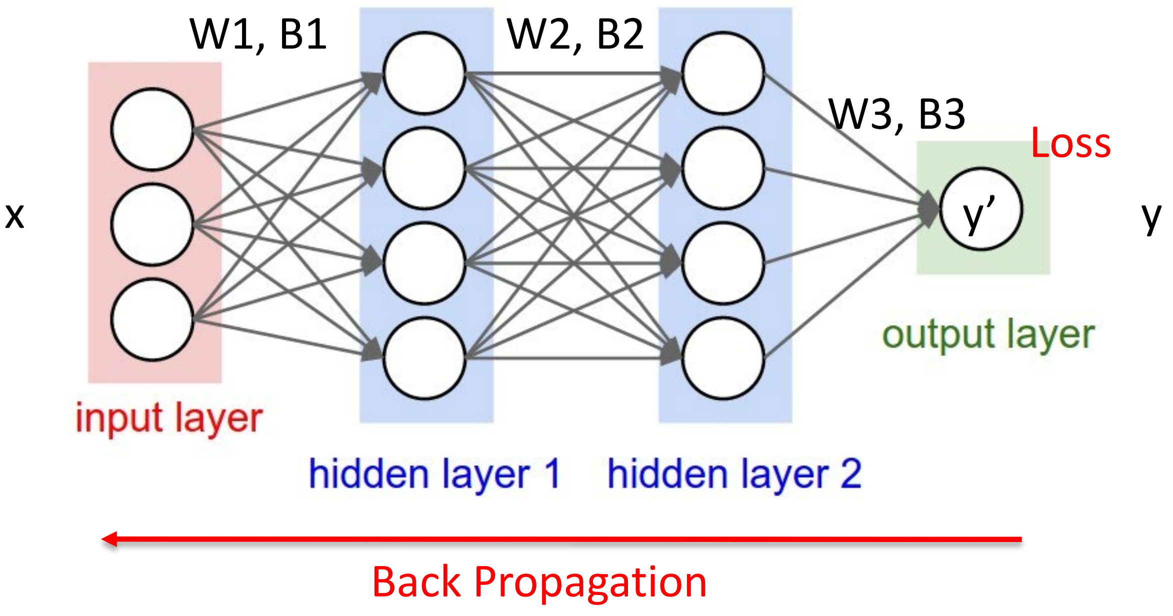
$$y' = W_3 f(W_2 f(W_1 x + b_1) + b_2) + b_3)$$

Loss function: L2 loss

$$l(y, y') = (y - y')^2$$

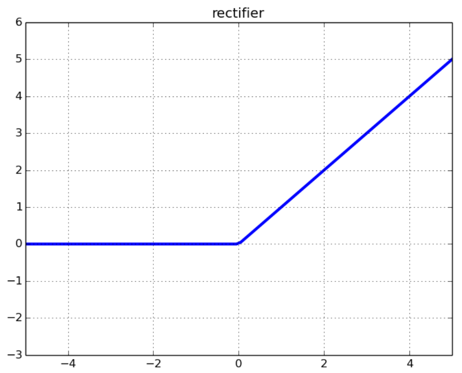
Neural Networks

W = {W1, W2, W3, B1, B2, B3}



f: non-linear activation function

$$\mathrm{ReLU}(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases}$$



Model: Multi-Layer Perceptron (MLP) $y' = W_3 f(W_2 f(W_1 x + b_1) + b_2) + b_3)$

Loss function: L2 loss $l(y, y') = (y - y')^2$

Optimization: Gradient descent $W = W - \eta \frac{\partial L}{\partial W}$

Neural Networks

A three-layer network approximates any continuous function

Let $\varphi(\cdot)$ be a nonconstant, bounded, and monotonically-increasing continuous function. Let I_m denote the *m*-dimensional unit hypercube $[0,1]^m$. The space of continuous functions on I_m is denoted by $C(I_m)$. Then, given any function $f \in C(I_m)$ and $\varepsilon > 0$, there exists an integer N, real constants $v_i, b_i \in \mathbb{R}$ and real vectors $w_i \in \mathbb{R}^m$, where $i = 1, \cdots, N$, such that we may define:

$$F(x) = \sum_{i=1}^{N} v_i \varphi\left(w_i^T x + b_i\right)$$

as an approximate realization of the function f where f is independent of φ; that is,

$$|F(x) - f(x)| < \varepsilon$$

for all $x \in I_m$. In other words, functions of the form $F(x)$ are dense in $C(I_m)$.

Neural Networks

A three-layer network approximates any continuous function

Let $\varphi(\cdot)$ be a nonconstant, bounded, and monotonically-increasing continuous function. Let I_m denote the m-dimensional unit hy

Then, given any

function $f \in C(I$

s $w_i \in \mathbb{R}^m$,

where $i = 1, \cdots,$

$$F(x) = \sum_{i=1}^{N}$$

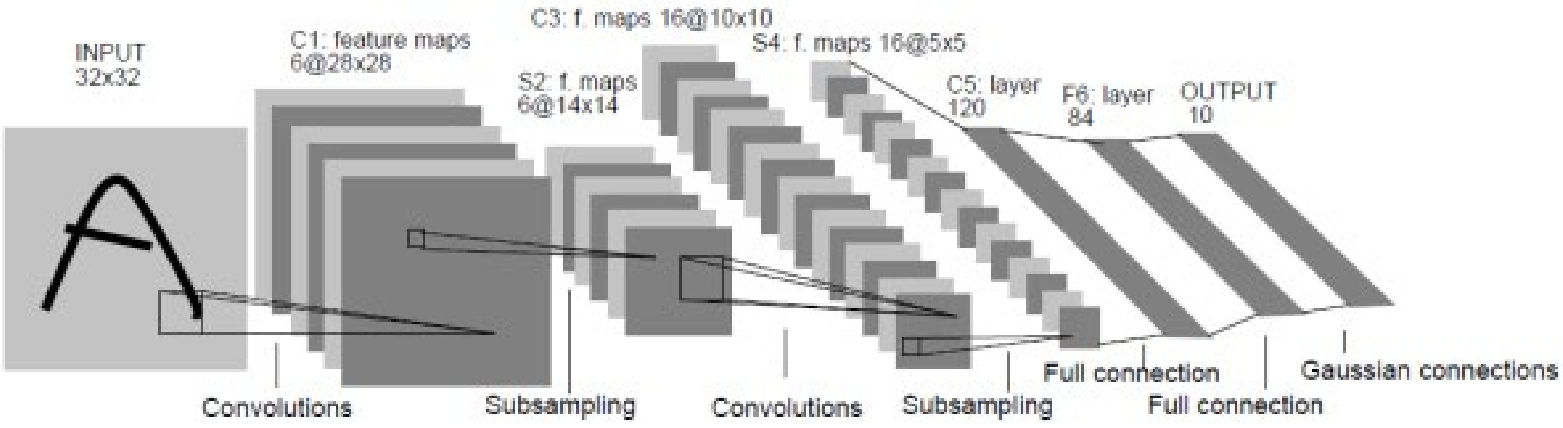
as an approximate

$$|F(x) - f(x)|$$

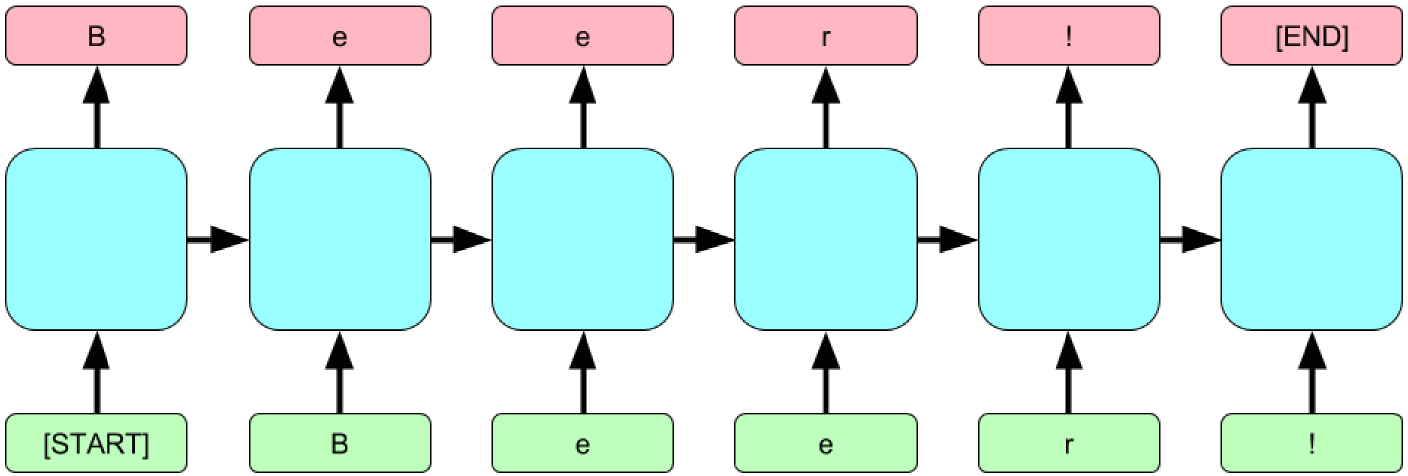
for all $x \in I_m$. In other words, functions of the form $F(x)$ are dense in $C(I_m)$.

At the cost of many parameters and much difficulty to fit

Neural Networks



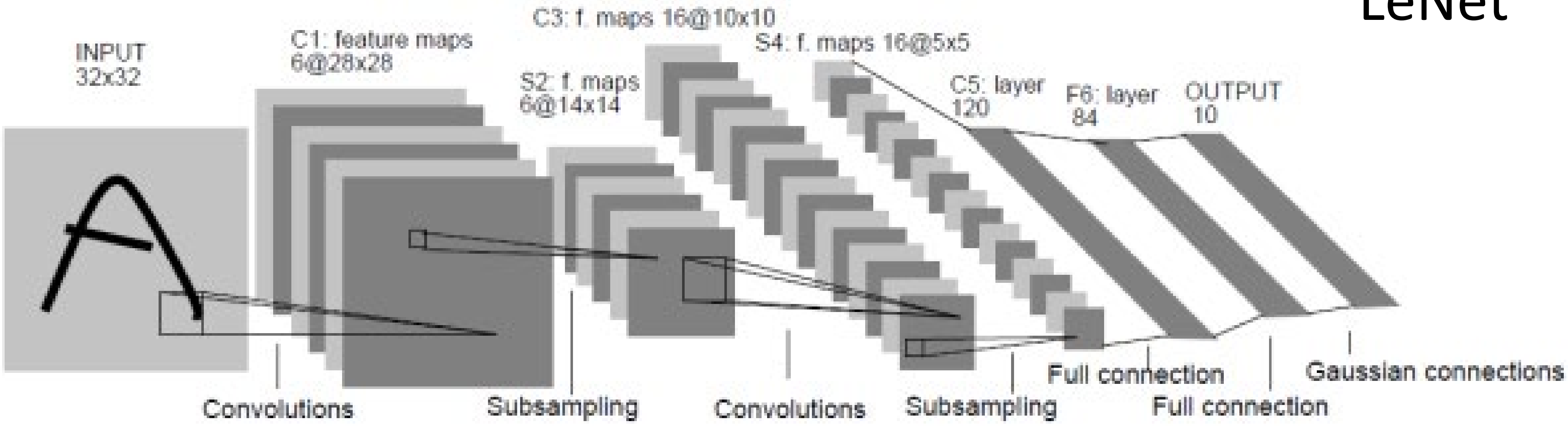
CNN

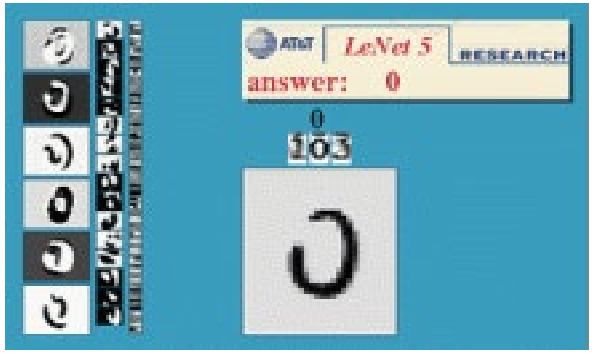


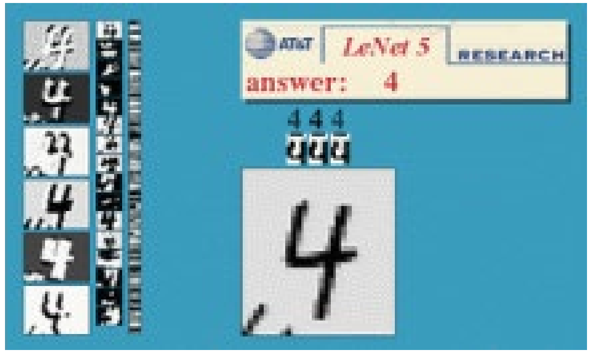
RNN

Convolutional Neural Networks

LeNet







One of the first successful applications of CNN.

Convolutional Neural Networks

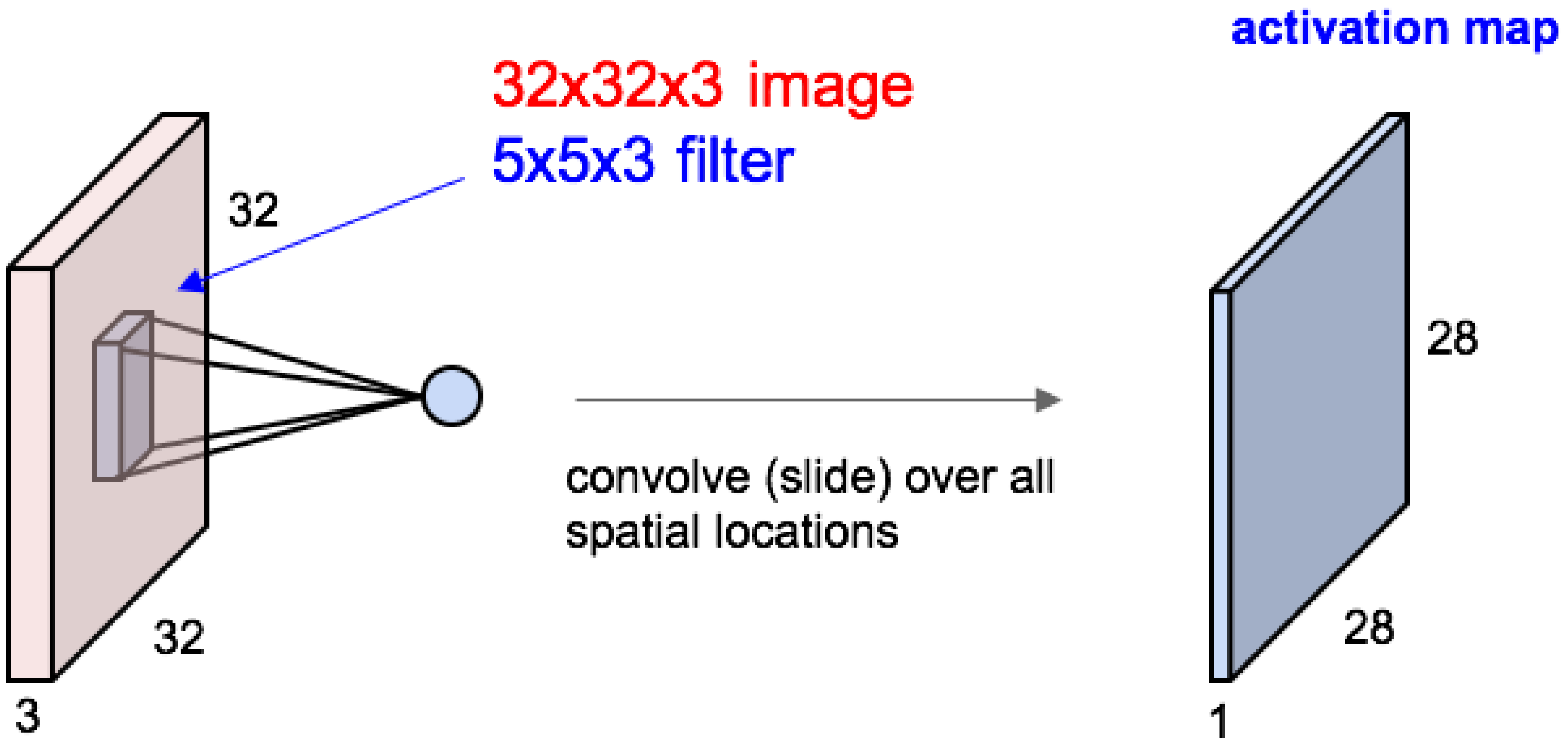


Image Credits: Andrej Karpathy

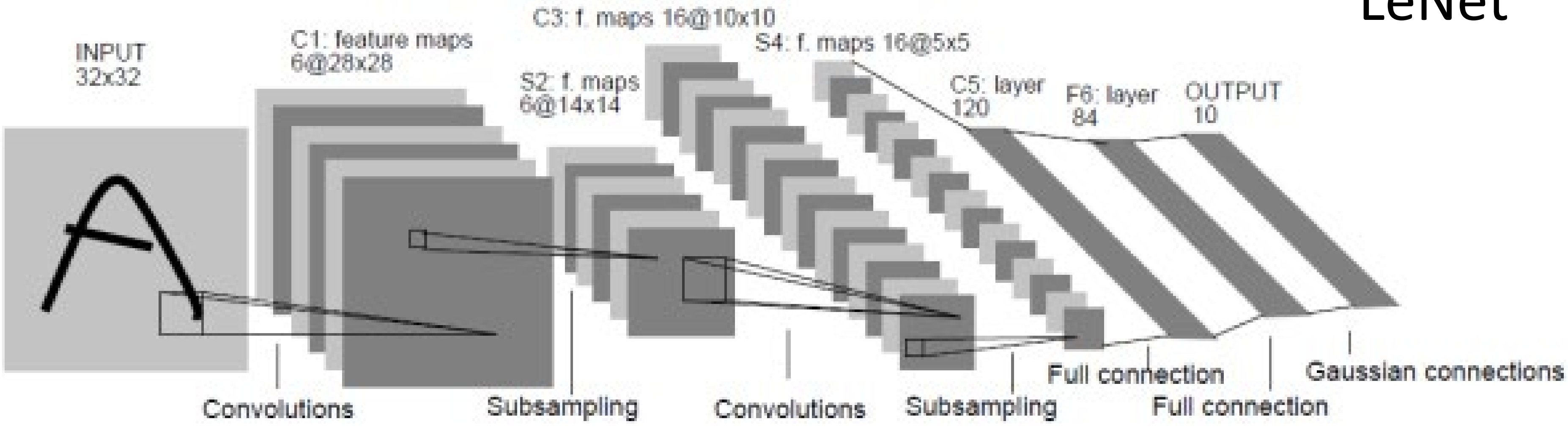
Convolutional Neural Networks

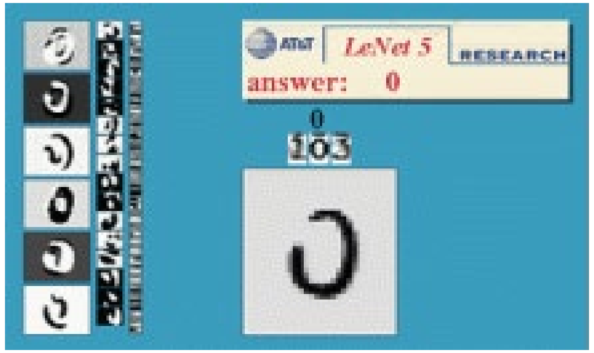
- Filters are doing pattern matching

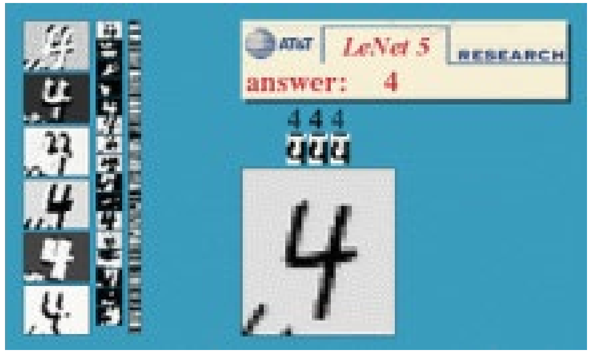
Image Credits: Yan LeCun

Convolutional Neural Networks

LeNet







One of the first successful applications of CNN.

ImageNet Challenge

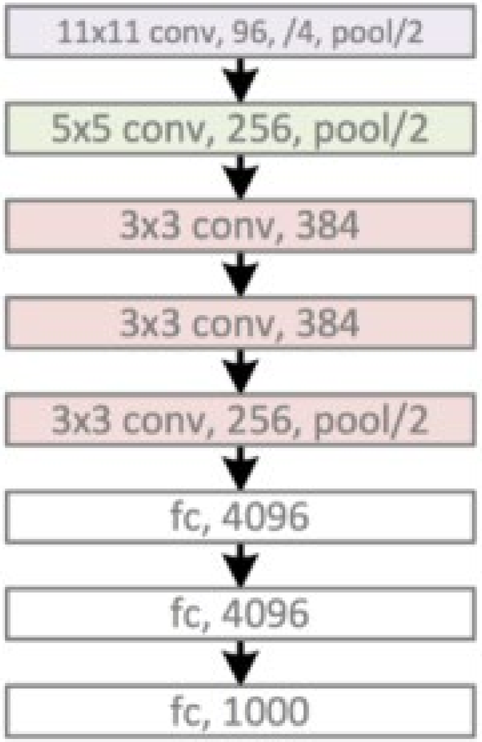
ImageNet Dataset

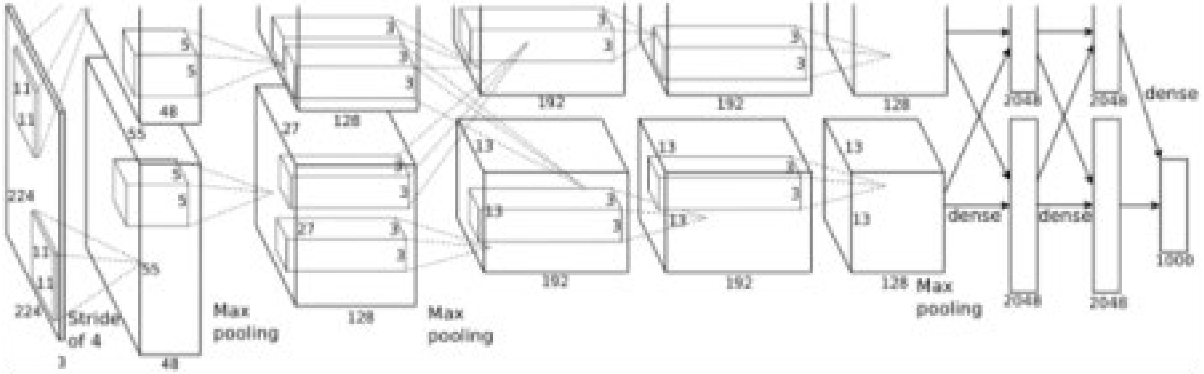
Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang et al. "Imagenet large scale visual recognition challenge." International Journal of Computer Vision 115, no. 3 (2015): 211-252. [web]

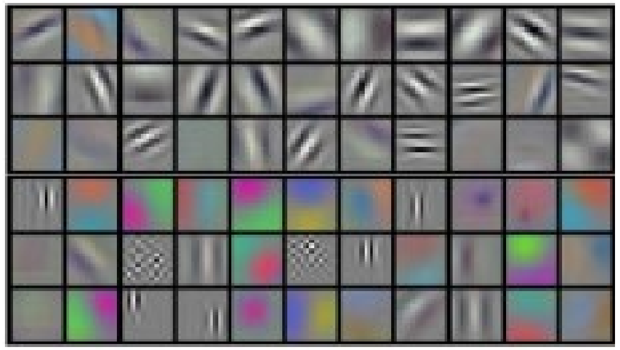
3

Convolutional Neural Networks

AlexNet



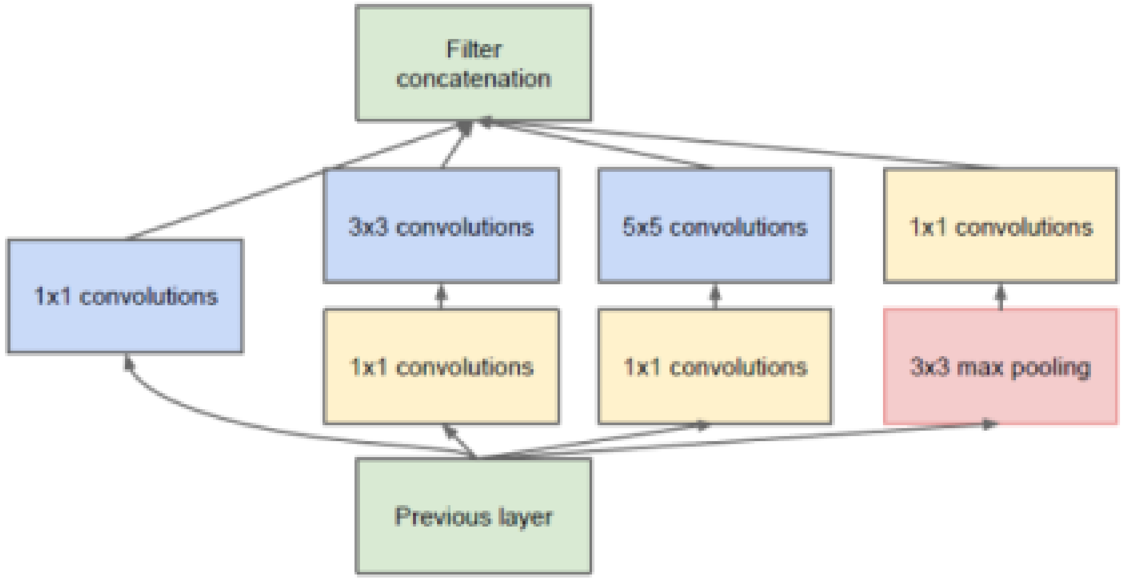




The first work that popularized Convolutional Networks in Computer Vision

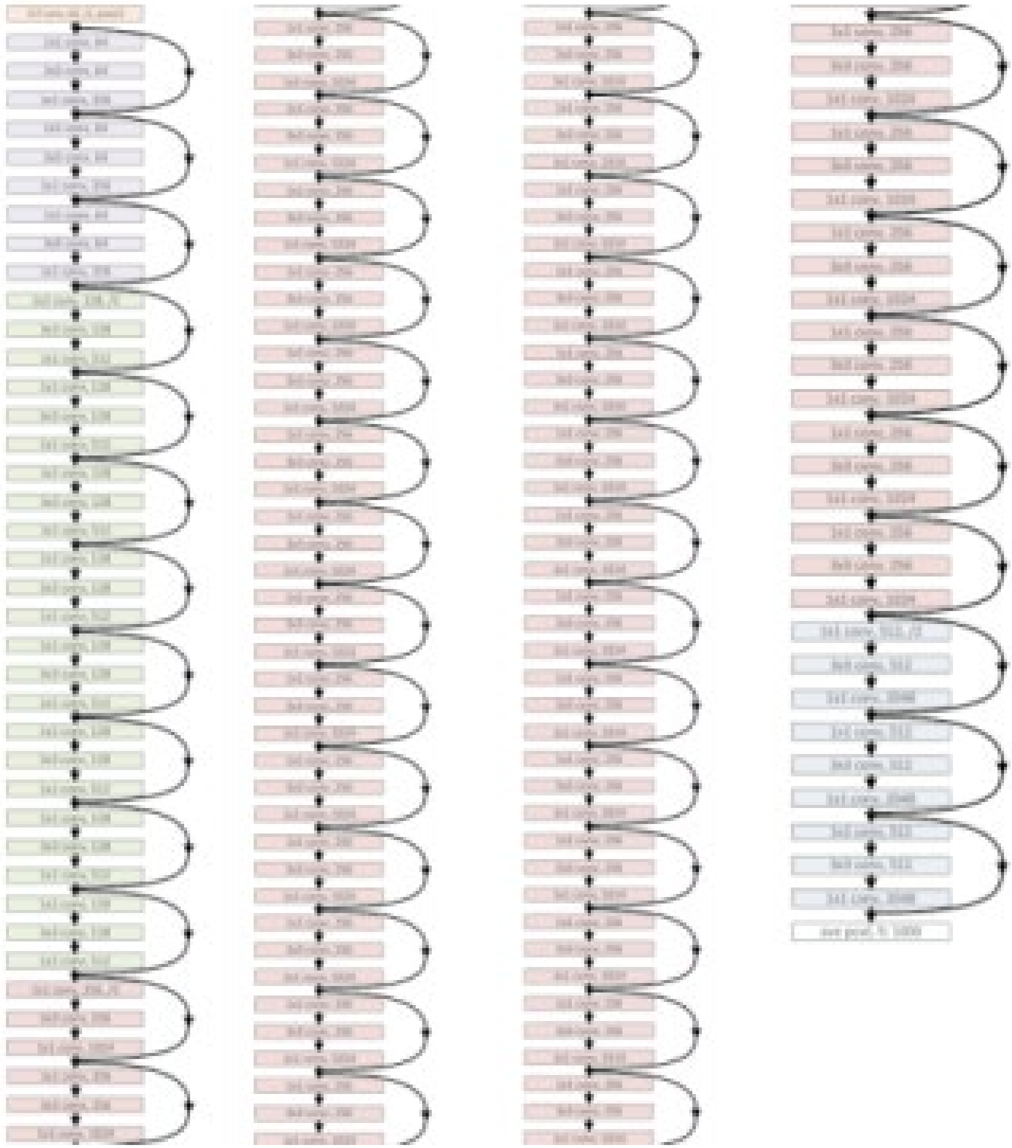
Convolutional Neural Networks

GoogleNet

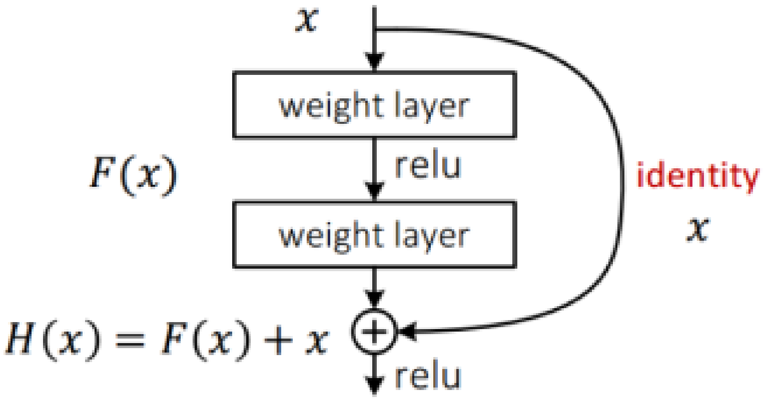


An Inception Module: a new building block..

Convolutional Neural Networks

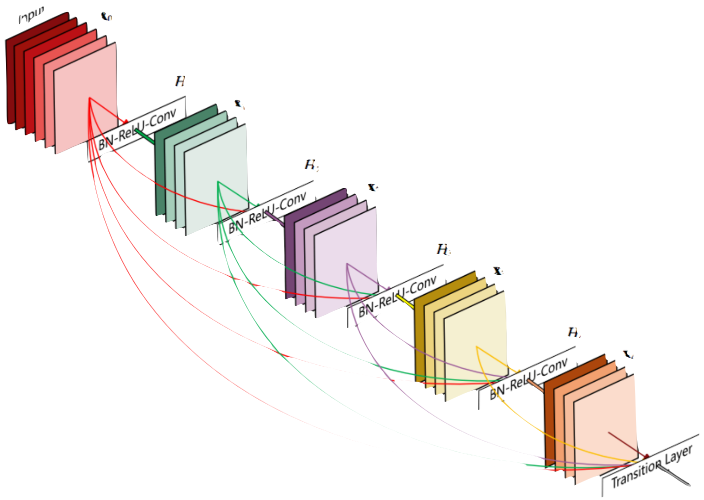


ResNet

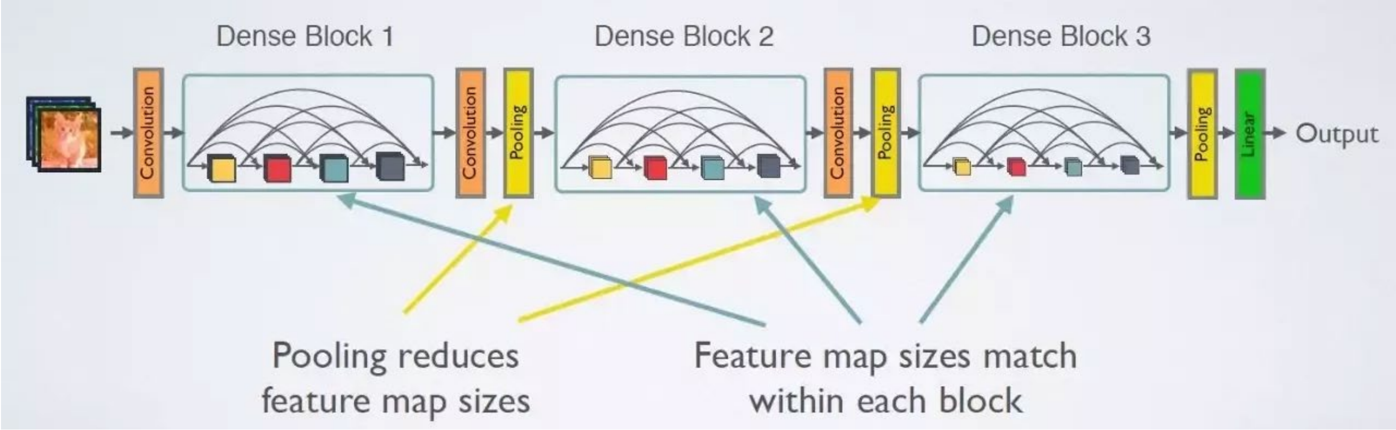


Convolutional Neural Networks

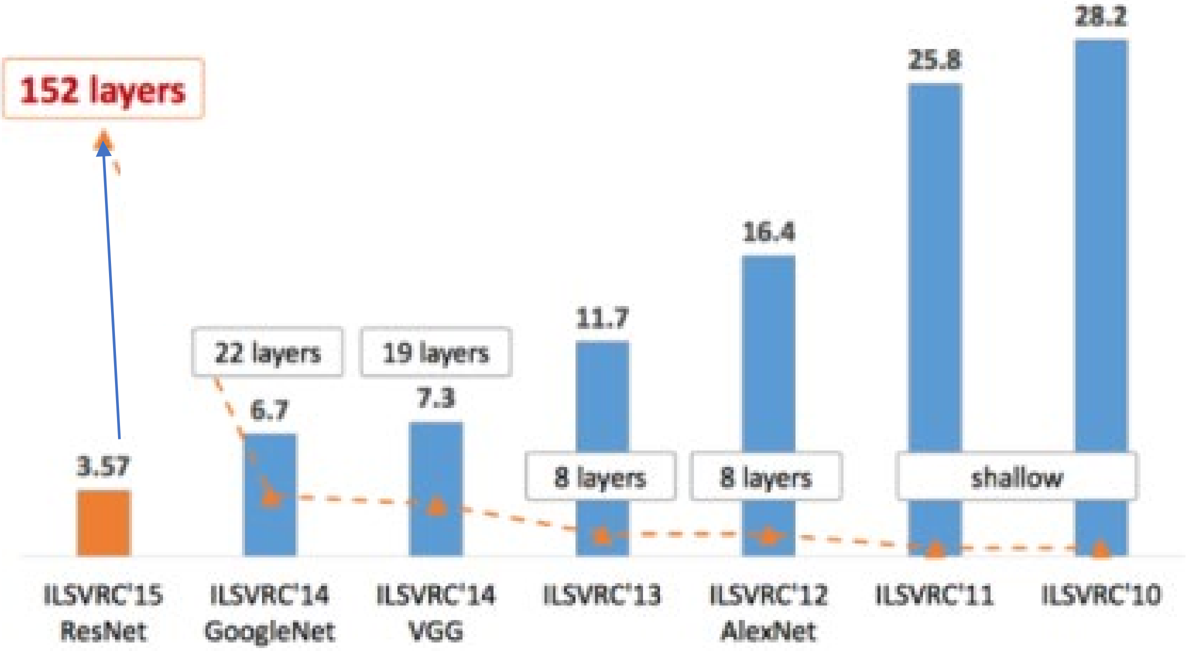
DenseNet



DenseNet-264
$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 64$
$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$

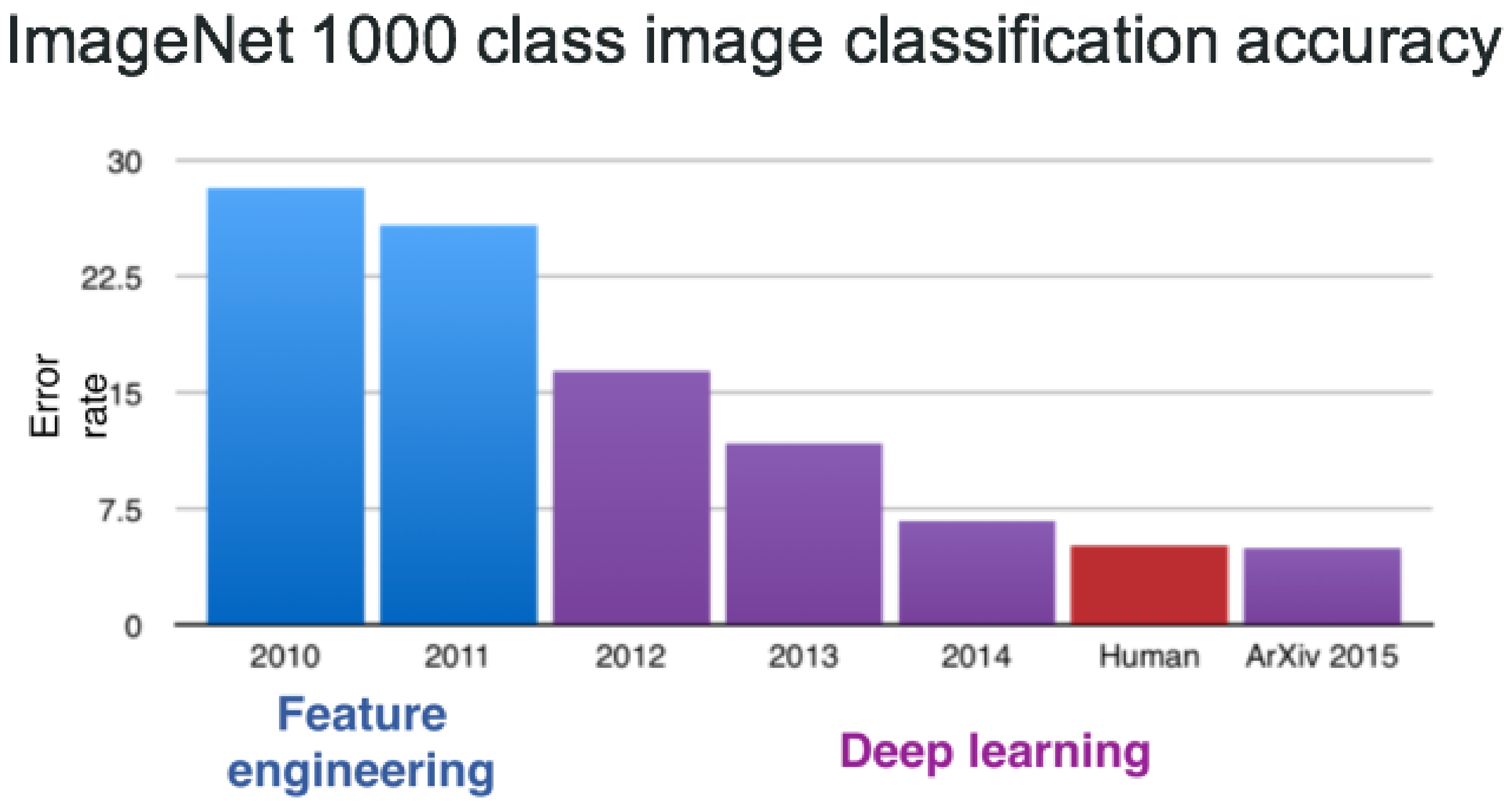


Convolutional Neural Networks



Classification Error: the lower, the better

Convolutional Neural Networks



Transformers

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

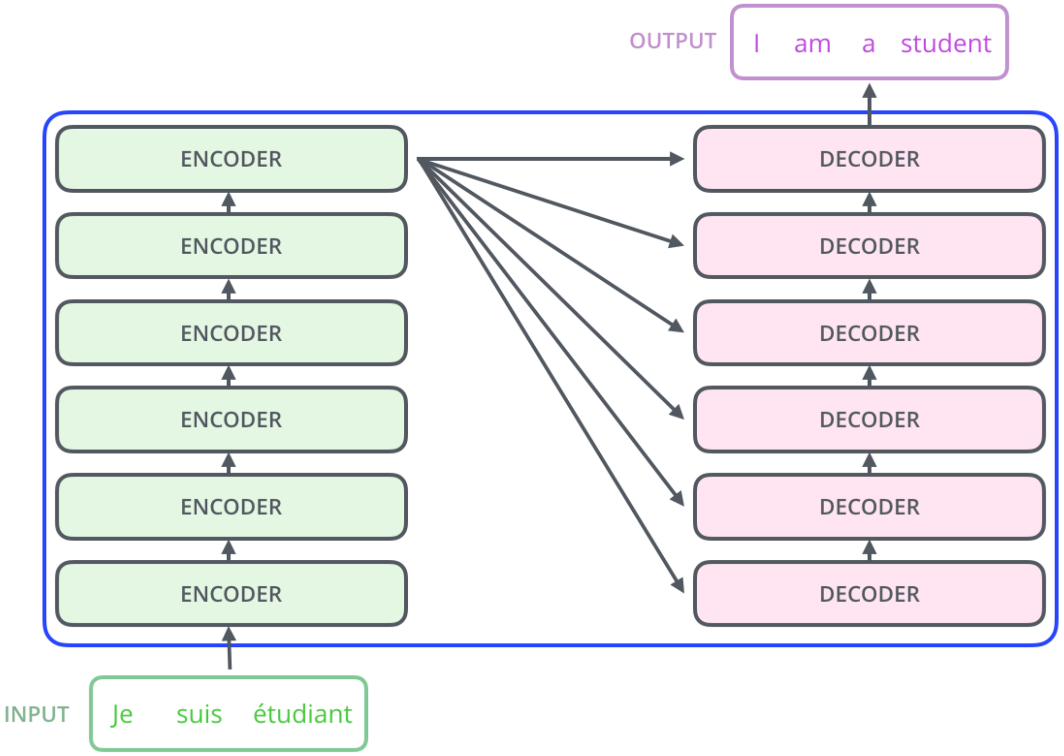
Llion Jones* Google Research llion@google.com

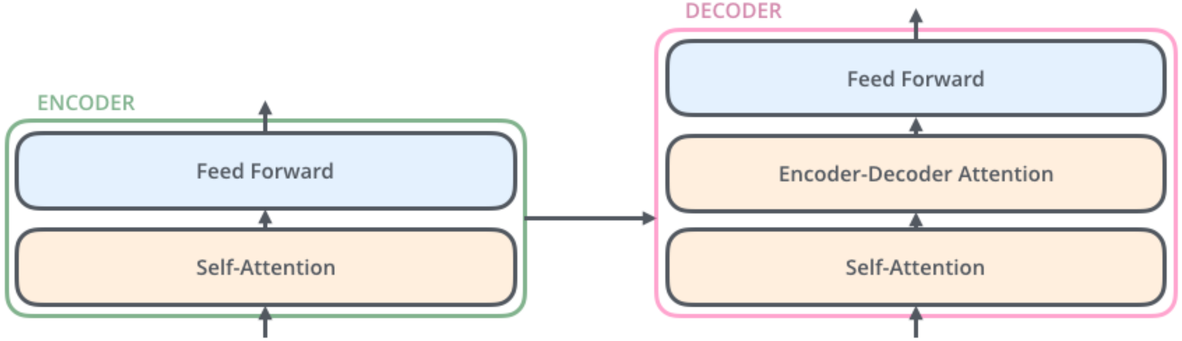
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

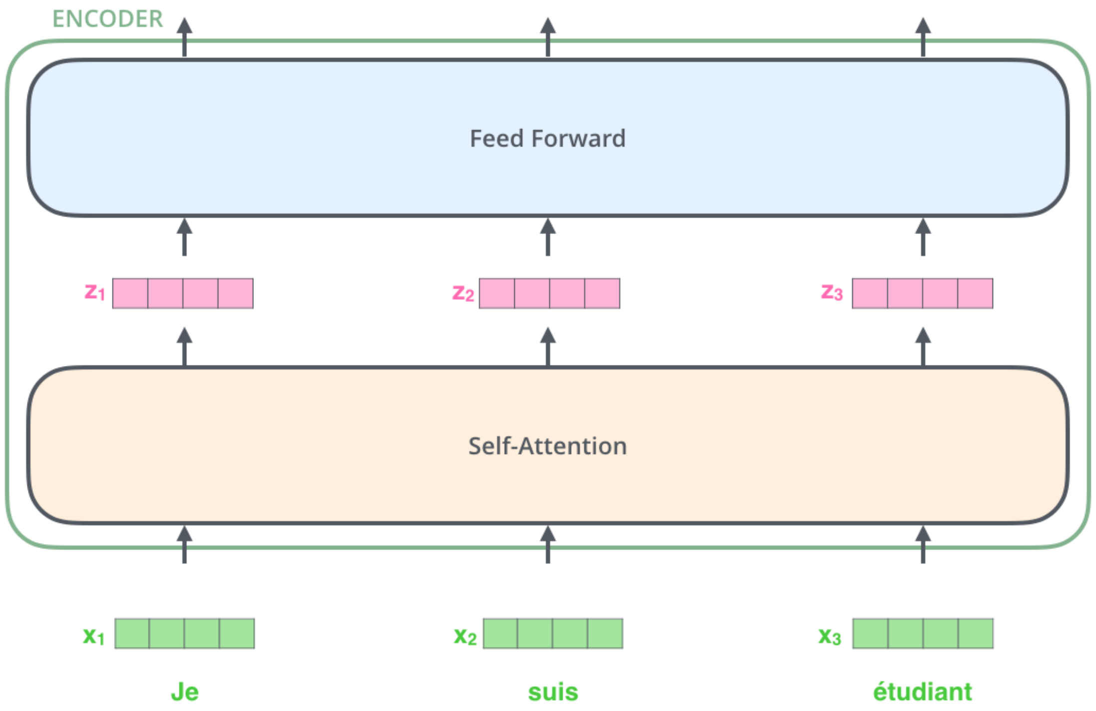
Transformers

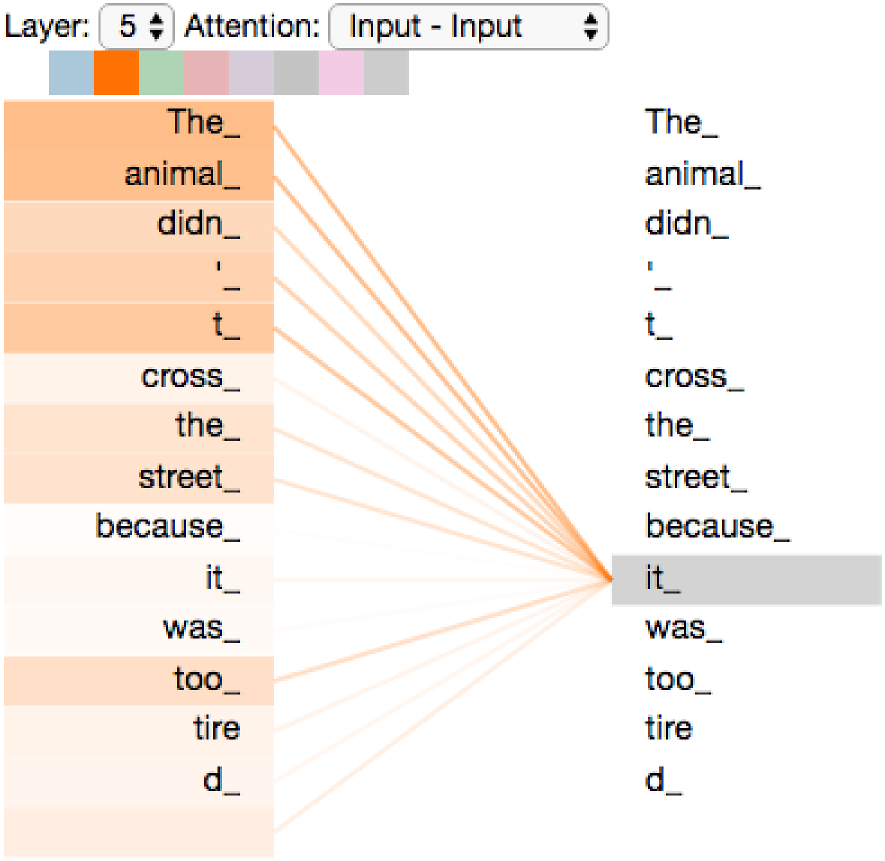




Credits: Jay Alammar's blog
(http://jalammar.github.io/illustrated-transformer/)

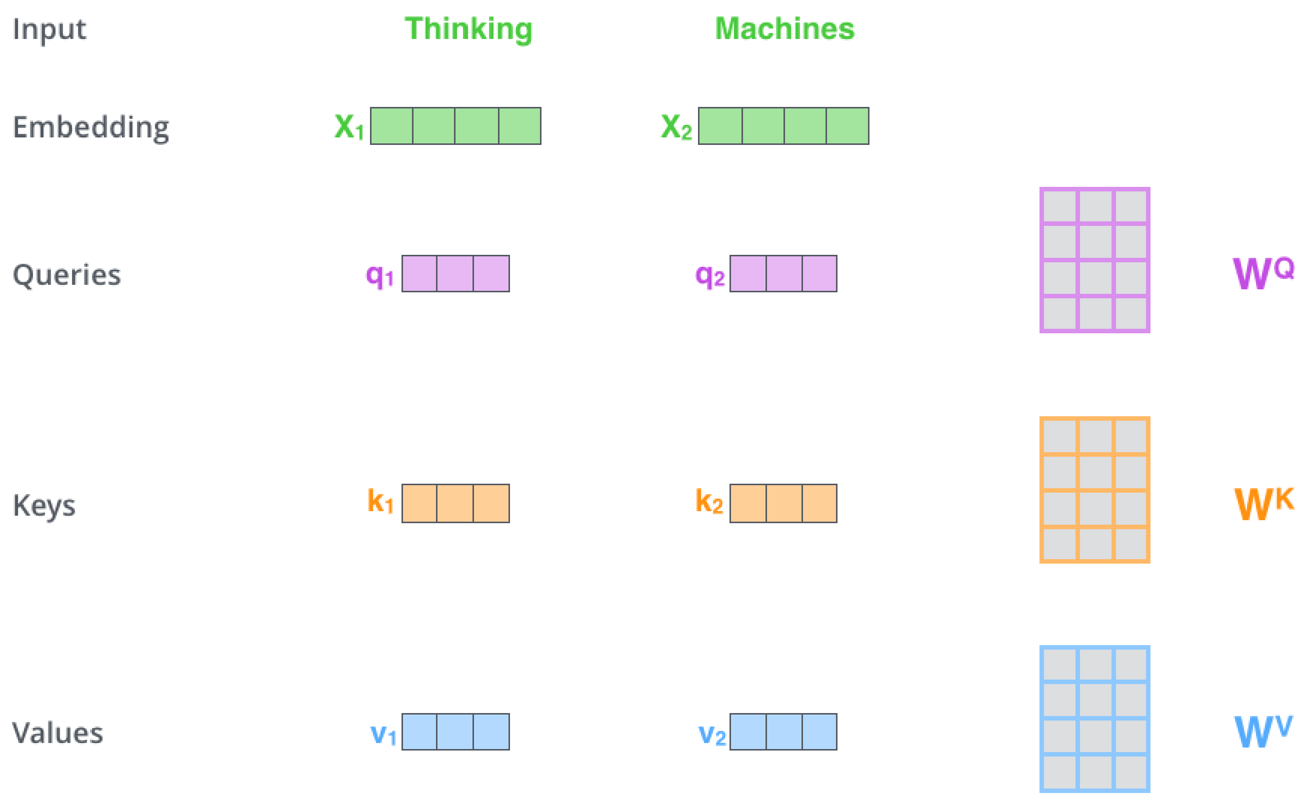
Transformers for NLP: Attention Mechanism





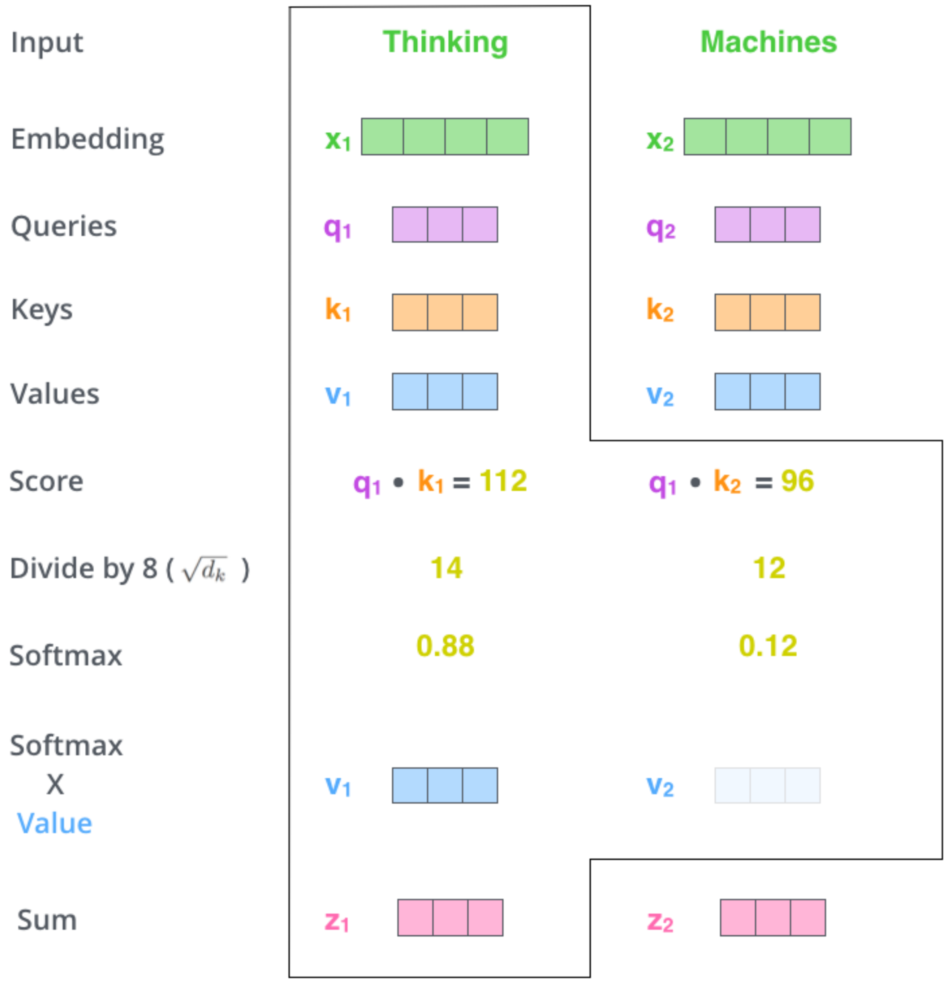
Credits: Jay Alammar's blog (http://jalammar.github.io/illustrated-transformer/)

A Closer Look at Self-Attention



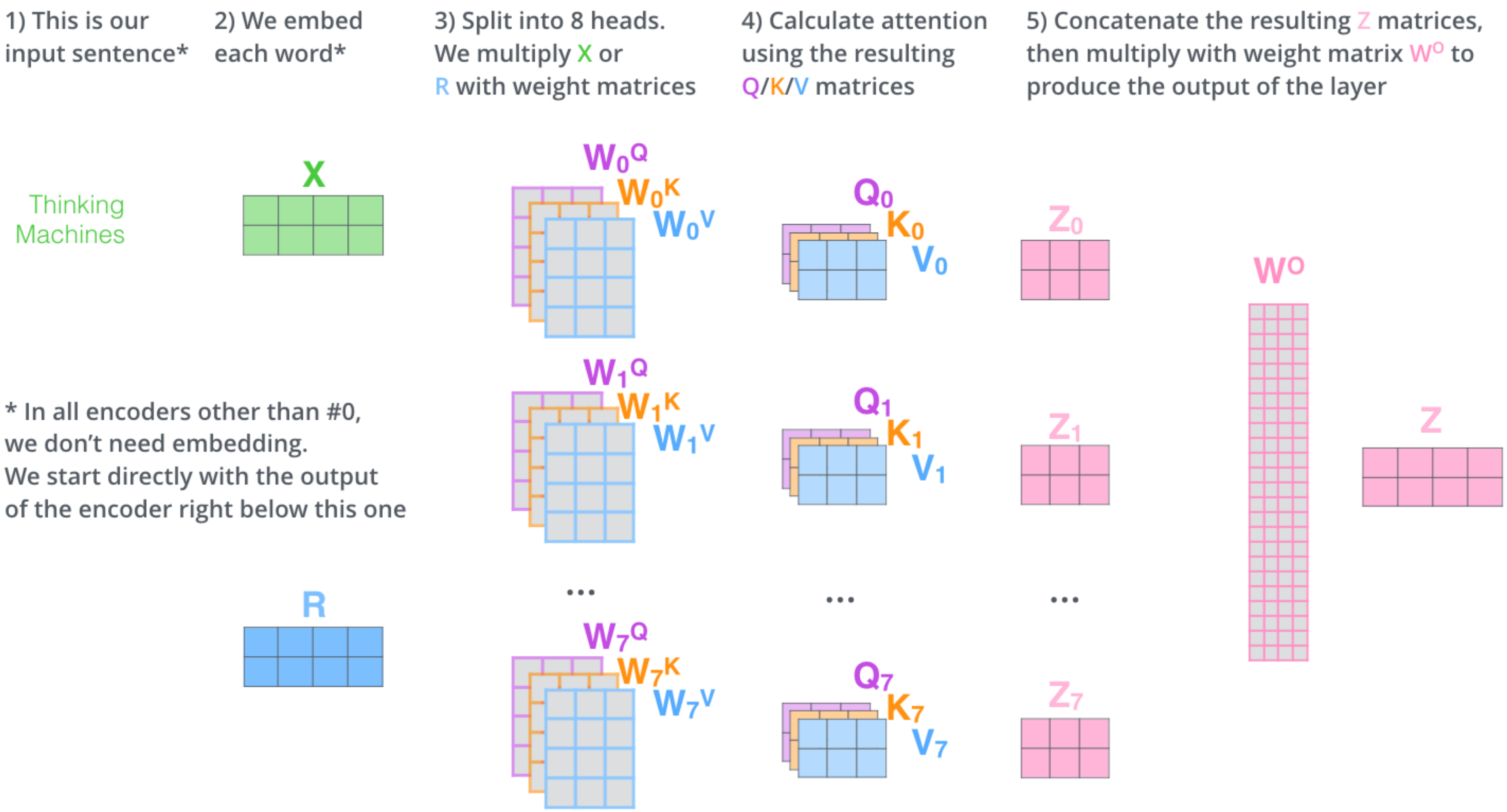
$Q = W^Q \times X; \quad K = W^K \times X; \quad V = X^V \times X$

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



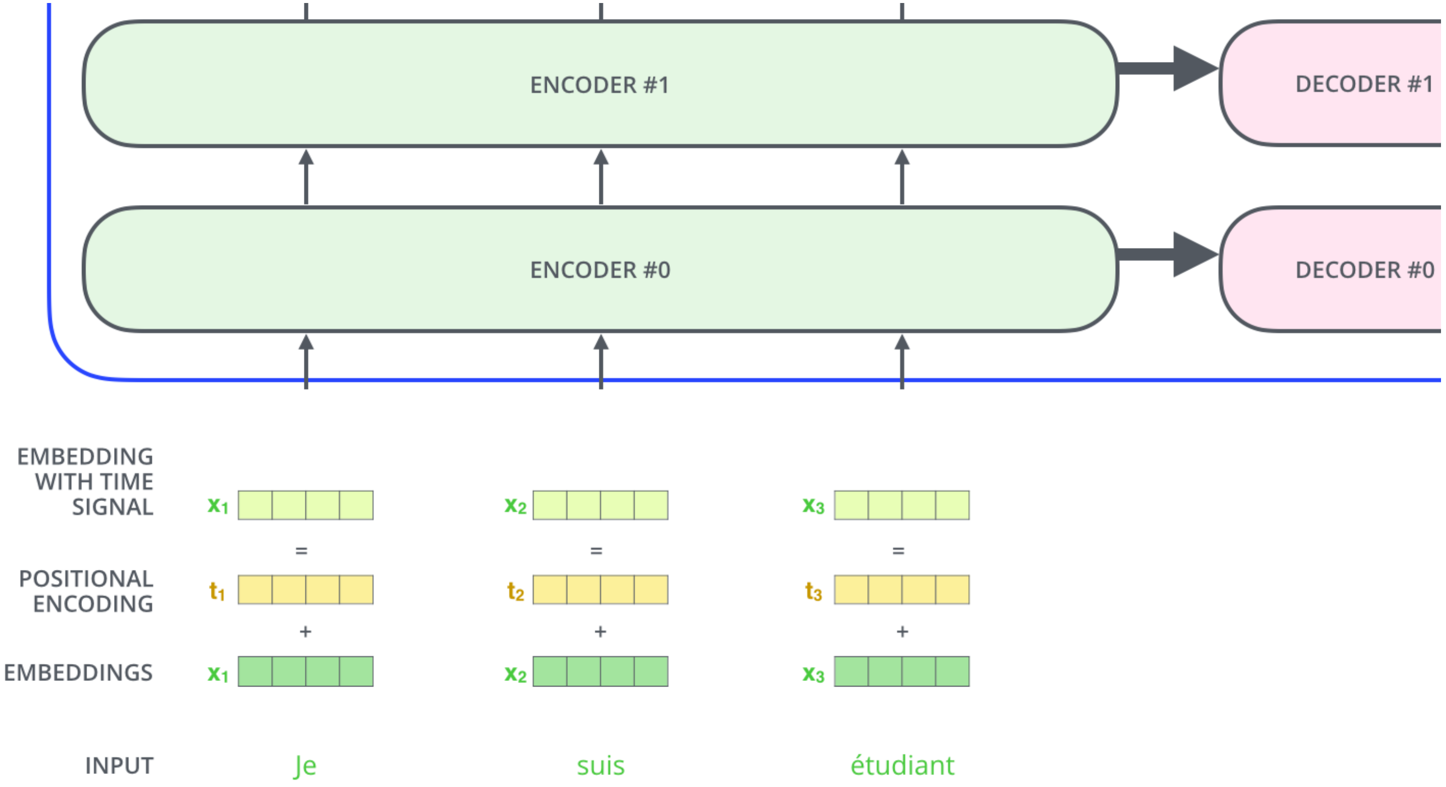
Credits: Jay Alammar's blog
(http://jalammar.github.io/illustrated-transformer/)

Multi-head Attentions

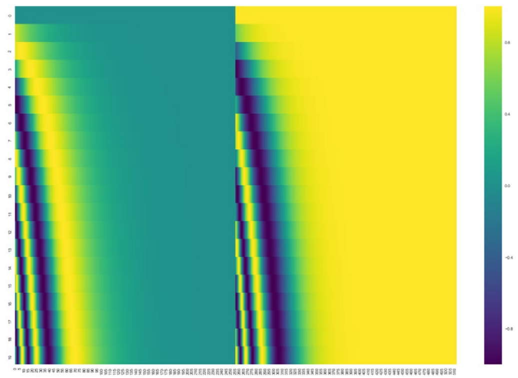


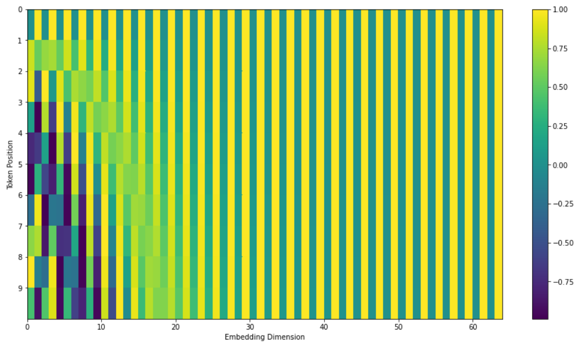
Credits: Jay Alammar's blog
(http://jalammar.github.io/illustrated-transformer/)

Positional Encoding



0, 1, 2, 3, ….



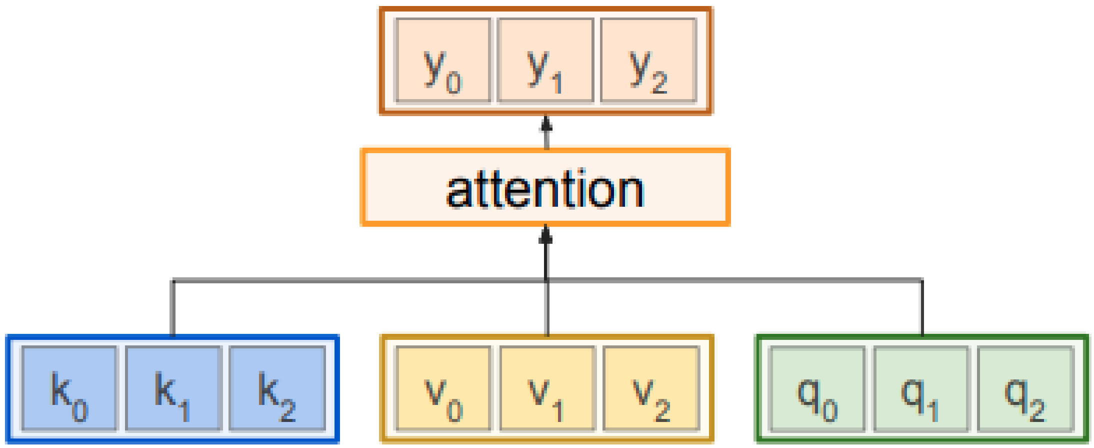


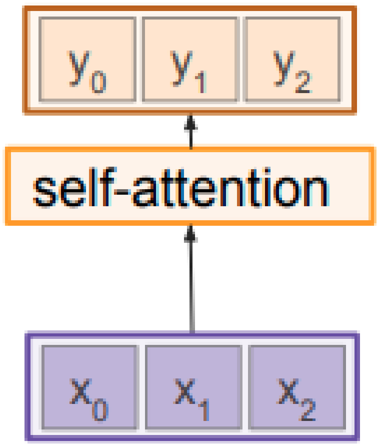
etc.

Read this blog for more details -->

Credits: Jay Alammar's blog (http://jalammar.github.io/illustrated-transformer/)

General Attention Layers v.s. Self-Attention



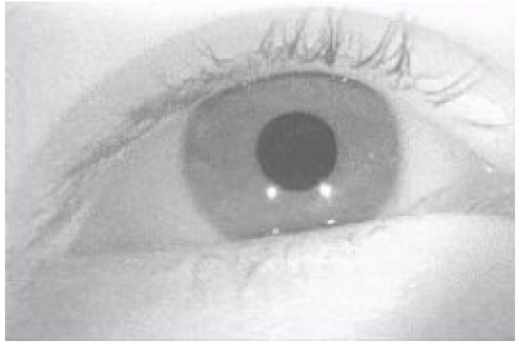


For example, this can be useful in decoding time, by feeding in the decoding features as queries.

Image Credits: CS 231n

Attentions for Vision (over 2D Images)

gif source



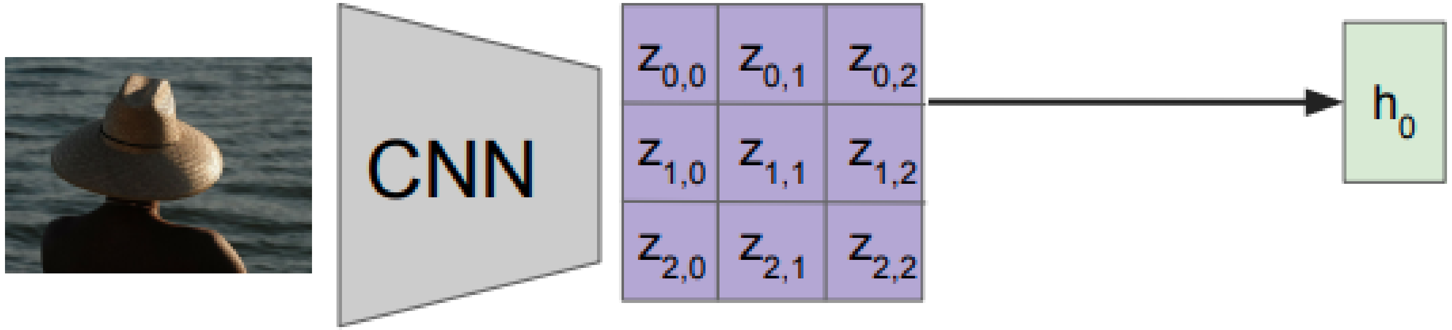
Attention Saccades in humans

Attention idea: New context vector at every time step.

Each context vector will attend to different image regions

Key & Value Matrix

A Query Vector



Extract spatial features from a pretrained CNN

Features: H x W x D

Global Image Attention

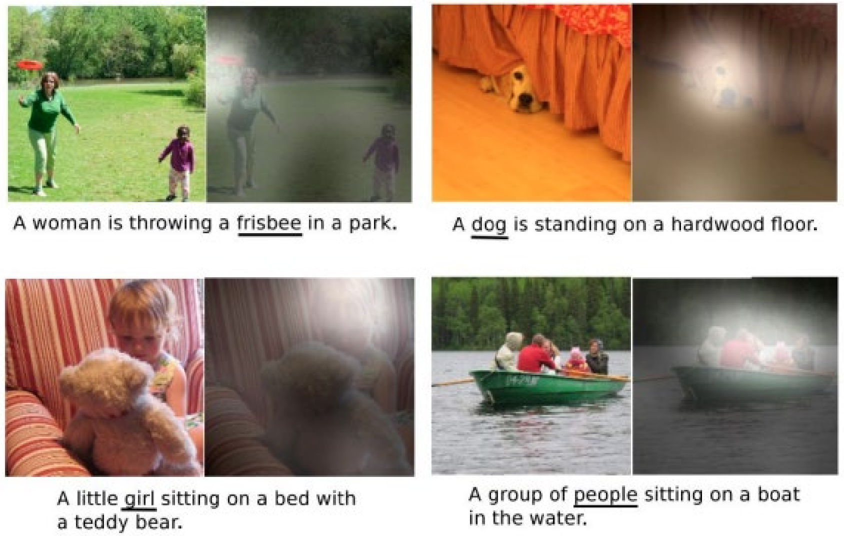
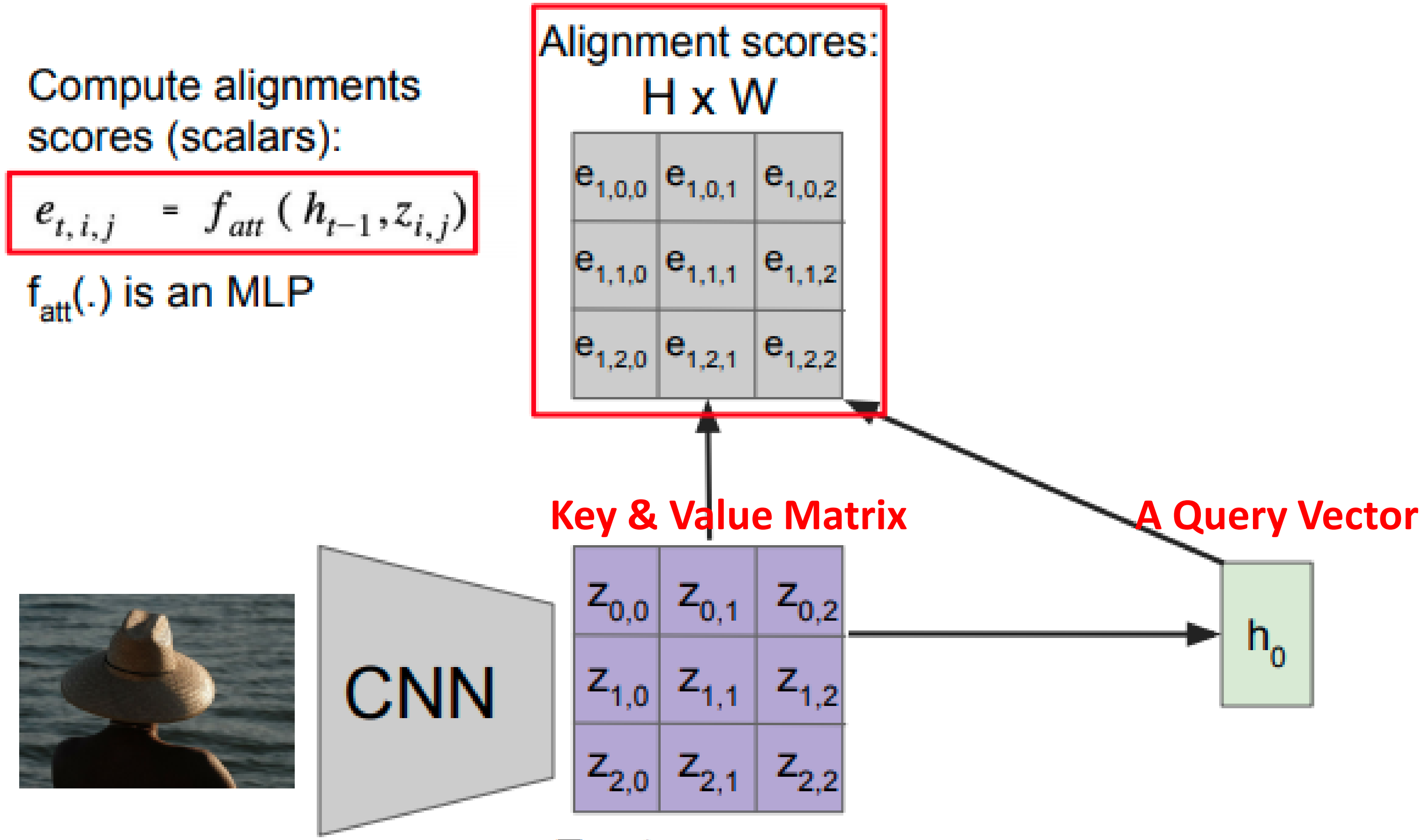


Image Credits: CS 231n

Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Attentions for Vision (over 2D Images)



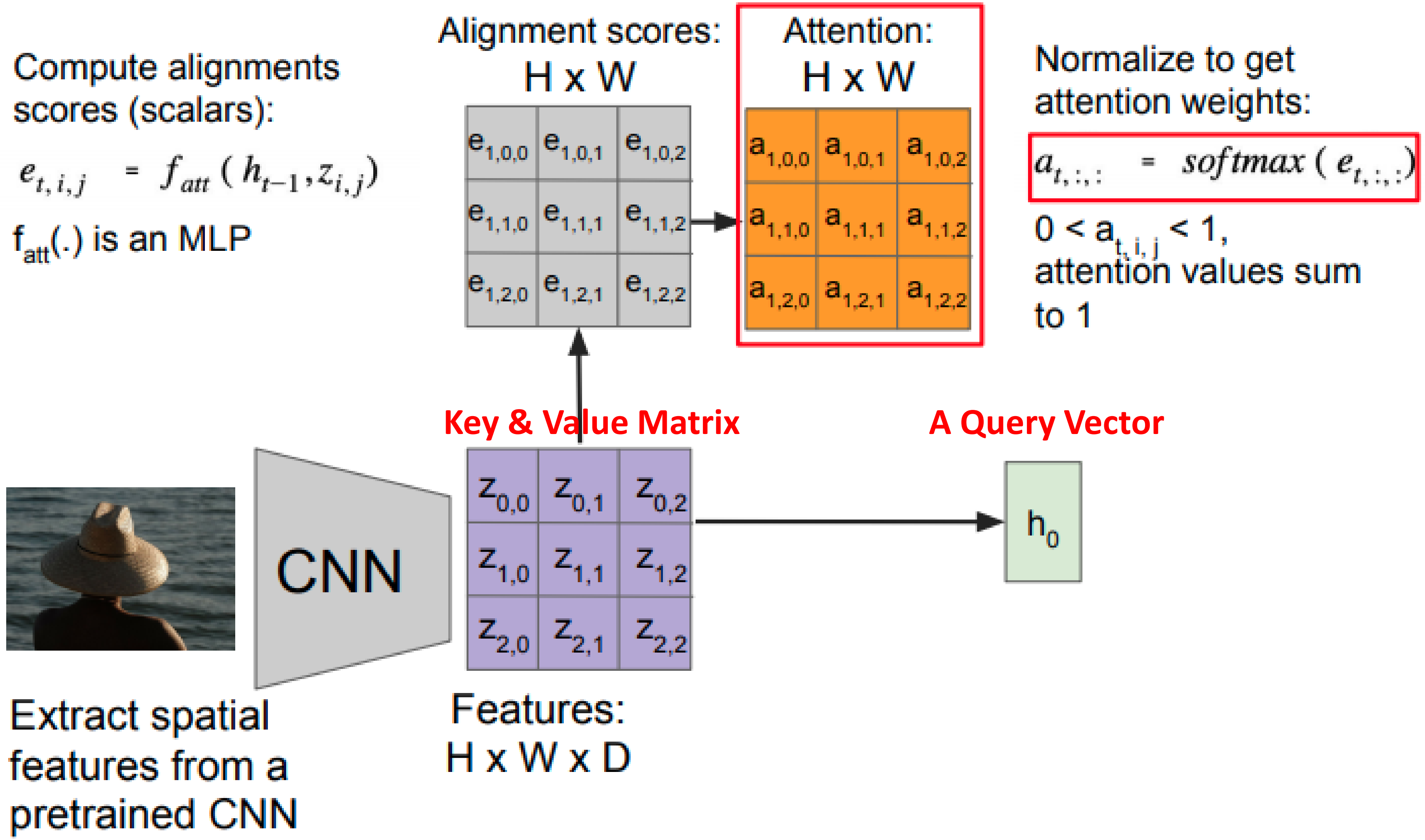
Extract spatial features from a pretrained CNN

Features: H x W x D

Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Image Credits: CS 231n

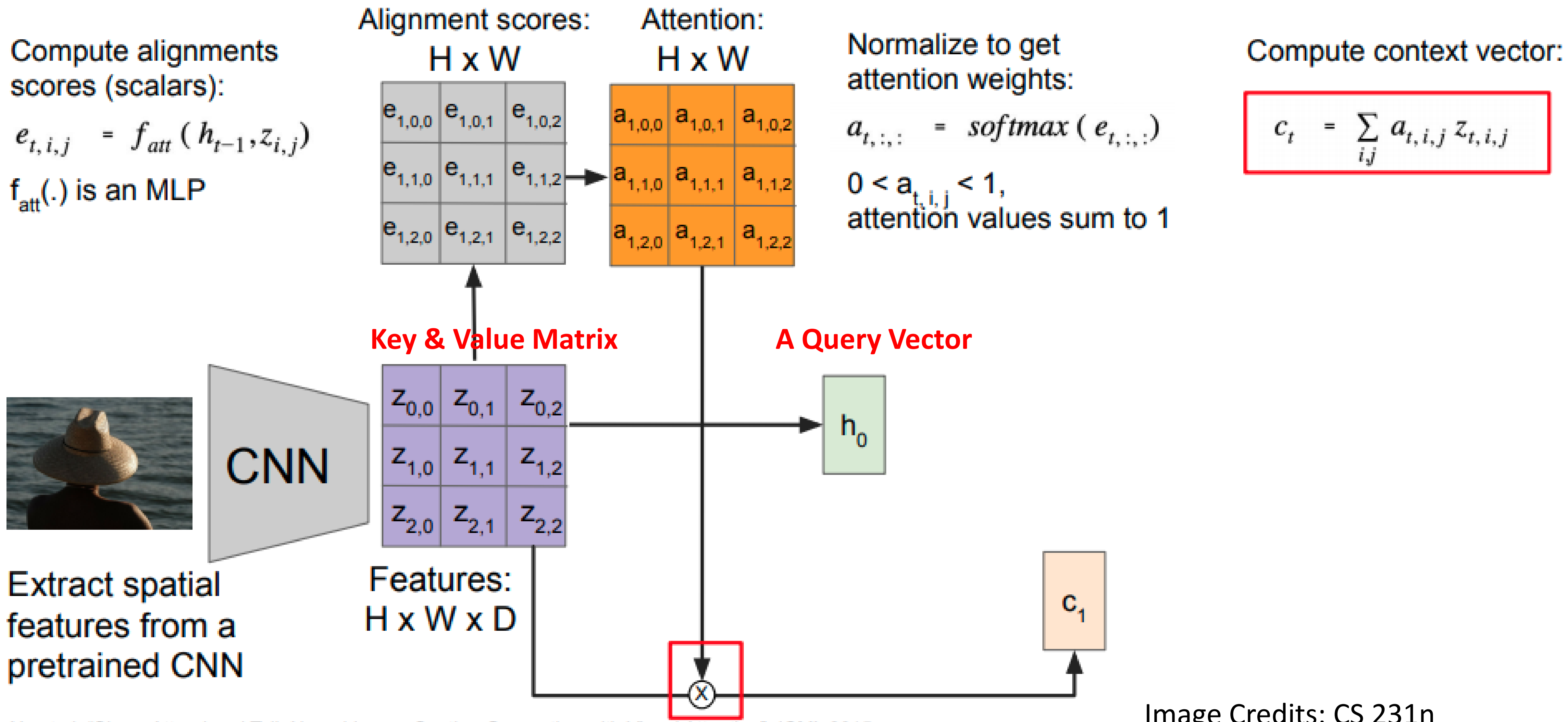
Attentions for Vision (over 2D Images)



Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Image Credits: CS 231n

Attentions for Vision (over 2D Images)



Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Image Credits: CS 231n

Visual Transformer (ViT)

Published as a conference paper at ICLR 2021

An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale

Alexey Dosovitskiy[∗,†], **Lucas Beyer**[∗], **Alexander Kolesnikov**[∗], **Dirk Weissenborn**[∗],
Xiaohua Zhai[∗], **Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer,**
Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby[∗,†]
[∗]equal technical contribution, [†]equal advising
Google Research, Brain Team
{adosovitskiy, neilhoulsby}@google.com

End-to-End Object Detection with Transformers

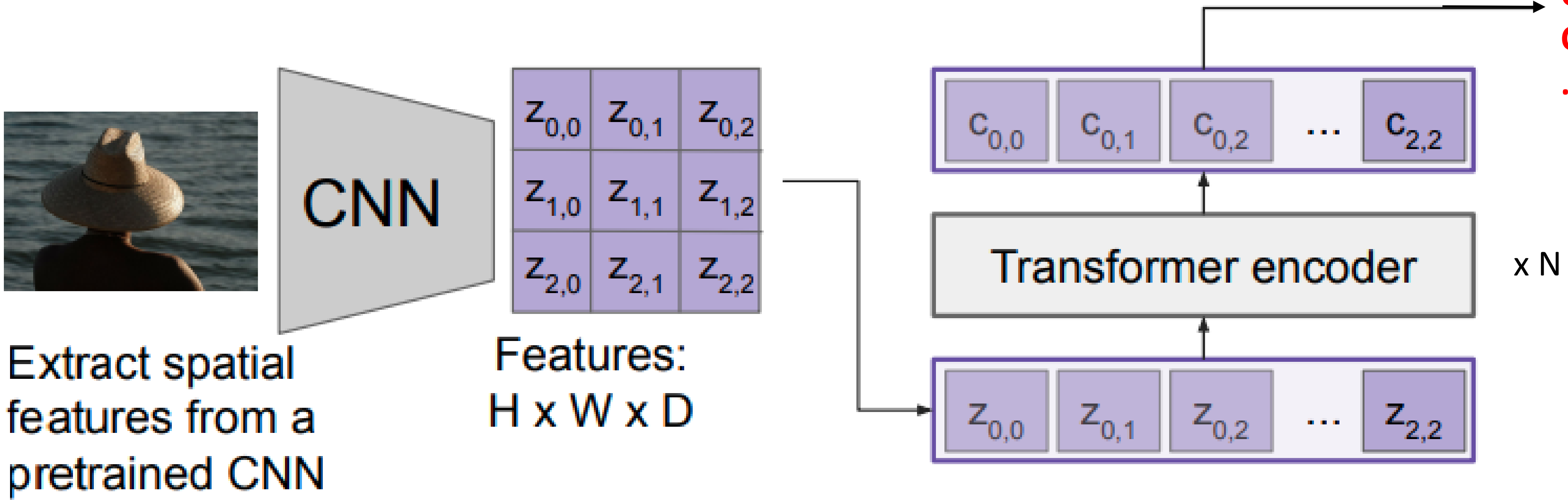
Nicolas Carion[⋆], Francisco Massa[⋆], Gabriel Synnaeve, Nicolas Usunier,
Alexander Kirillov, and Sergey Zagoruyko

Facebook AI

Visual Transformer (ViT)

Encoder: $\mathbf{c} = T_W(\mathbf{z})$
where $\mathbf{z}$ is spatial CNN features
$T_W(.)$ is the transformer encoder

<u>The general attention layer is a new type of layer (like Convolutional layer) that can be used to design new neural network architectures.</u>



Classification
Object Detection
Caption Generation
……

Flattened into tokens, but with positional encodings

Image Credits: CS 231n

Useful Courses

- CS 231n: http://cs231n.stanford.edu/
  - Deep Learning for Visual Data
- CS 224n: http://cs224n.stanford.edu/
  - Deep Learning for Language / Sequential Data
- CS 234: http://cs234.stanford.edu/
  - Reinforcement Learning

Programming Neural Networks

- TensorFlow, Python, Google
  - https://www.tensorflow.org/
- **PyTorch, Python, Facebook**
  - https://pytorch.org/
- Caffe, Berkeley
  - http://caffe.berkeleyvision.org/

Programming Neural Networks

Keras

```
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
```

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
```

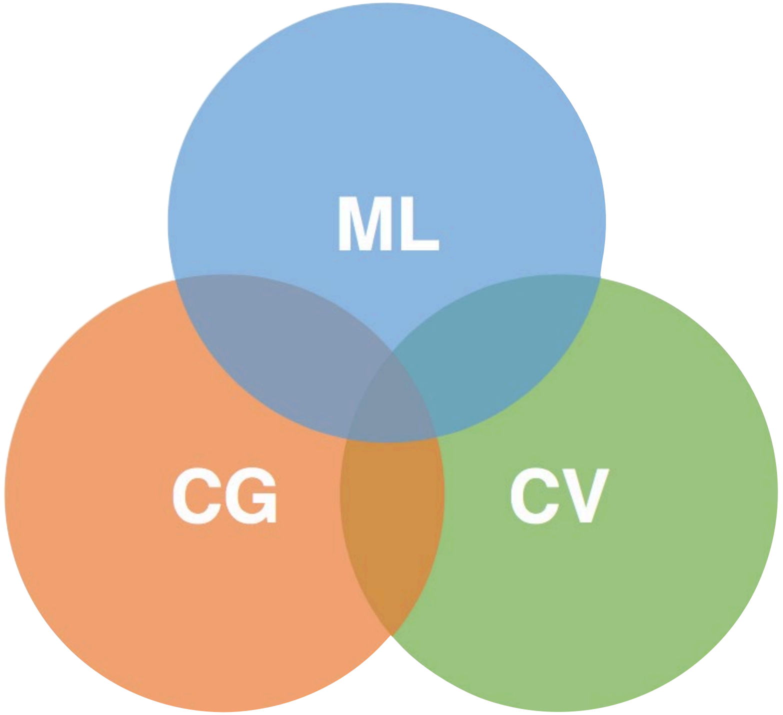
```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])

model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

3D Deep Learning

(let's do DL / train NNs over 3D data)

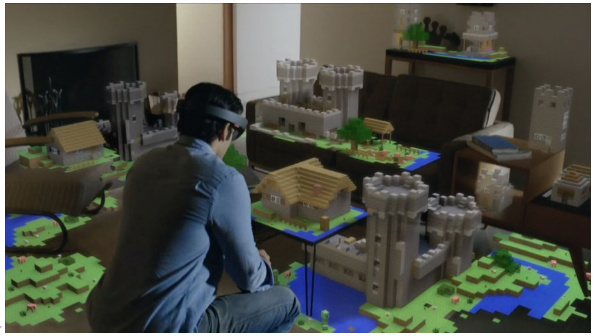
3D Deep Learning

- A field with very short history — starting from 2015 (approx.)
- But very active due to huge industry interests!

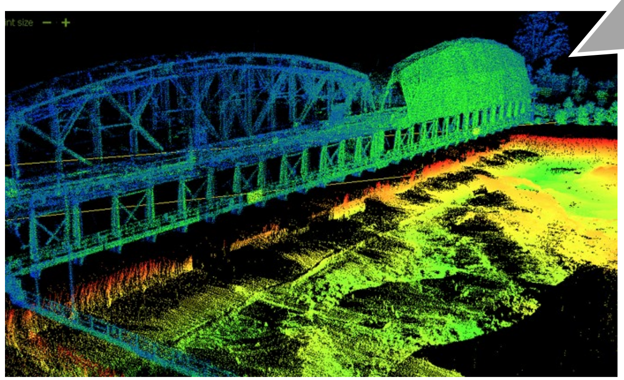


3D Applications

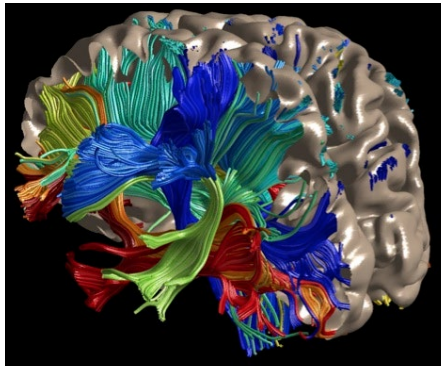
Robotics



Augmented Reality



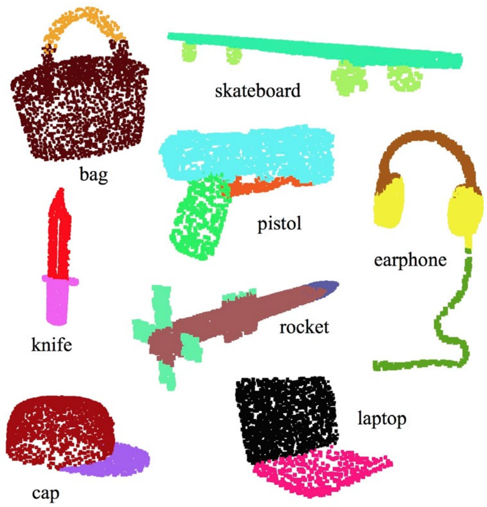
Autonomous driving



Medical Image Processing

3D Shape Understanding and Analysis

Classification

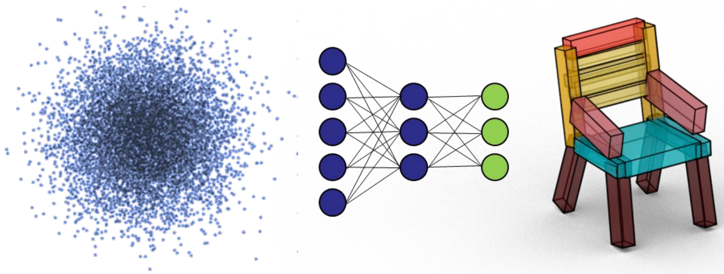


Parsing
(segmentation)

Correspondence

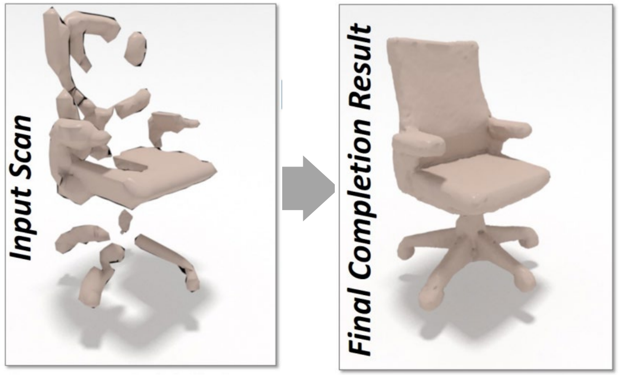
3D Deep Learning Tasks

3D Shape Generation and Synthesis



Shape generation

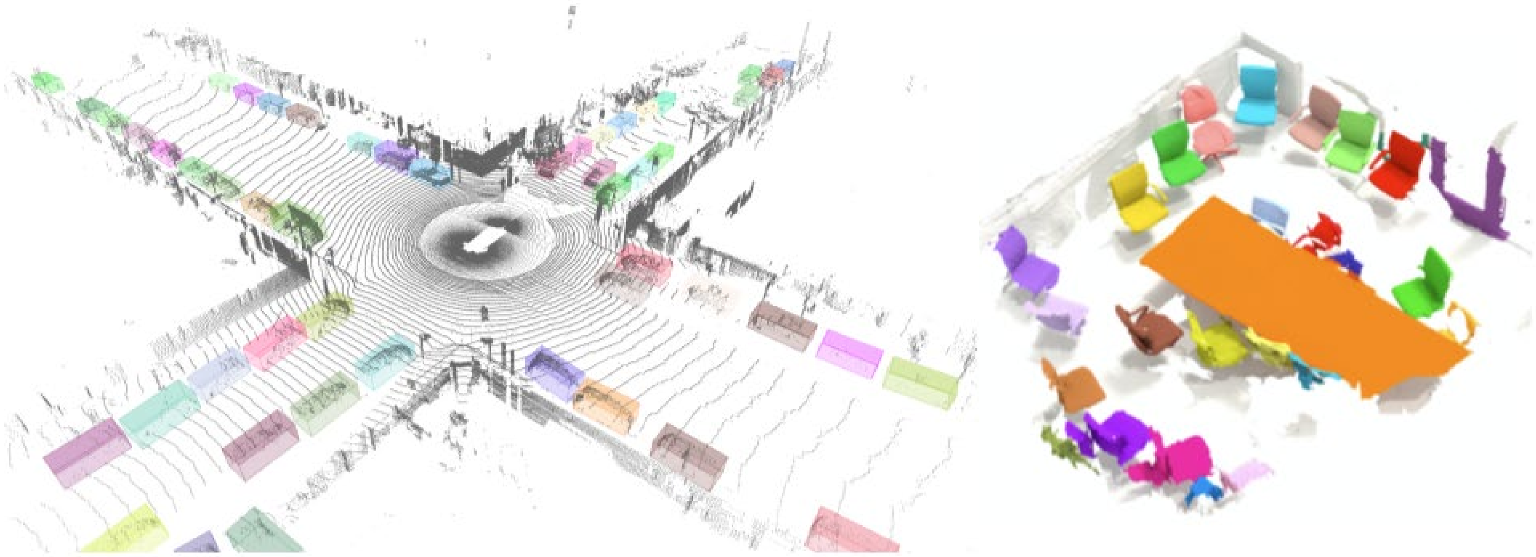
Monocular
3D reconstruction



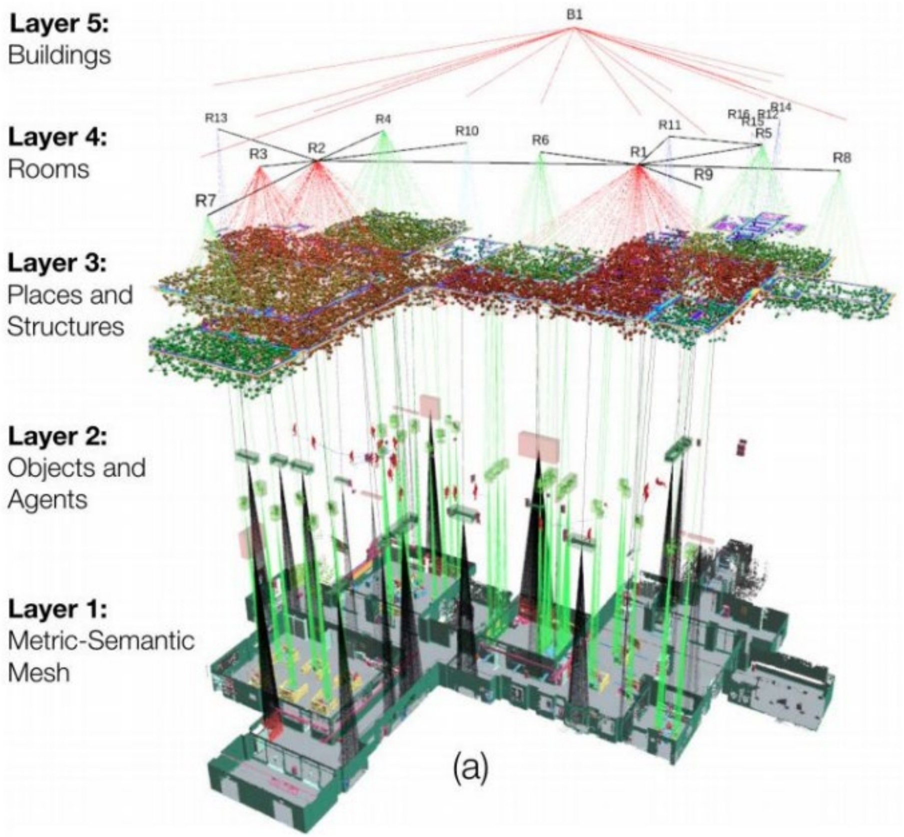
Shape completion

3D Deep Learning Tasks

3D Scene Understanding and Analysis



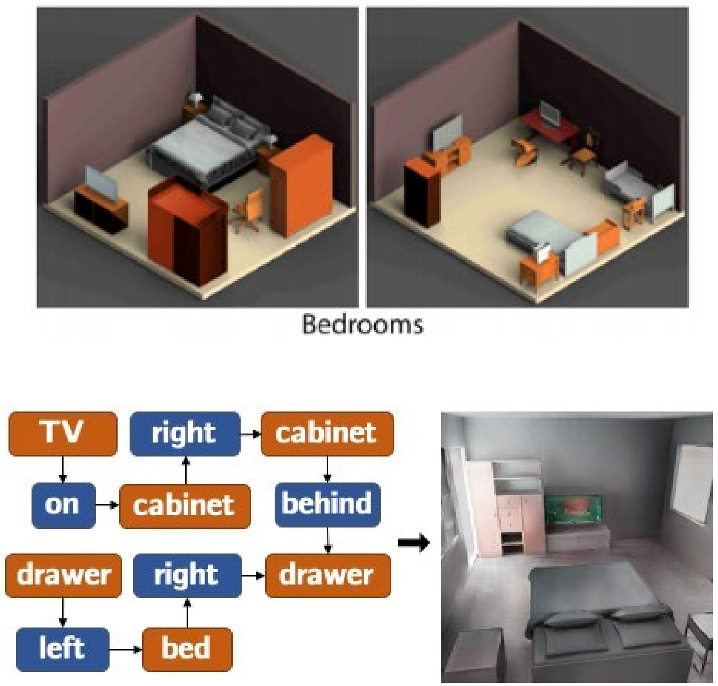
Object Detection / Scene Segmentation



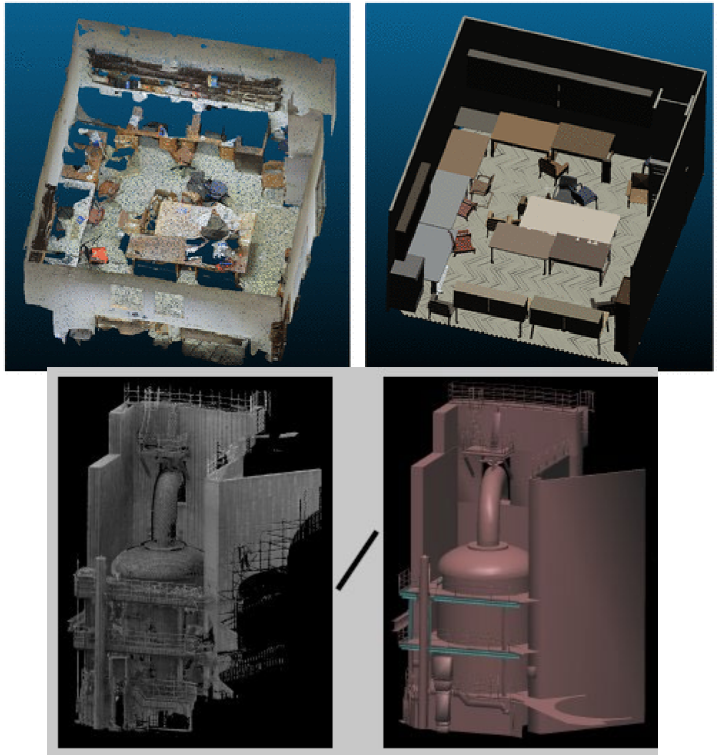
Scene Structure Parsing

3D Deep Learning Tasks

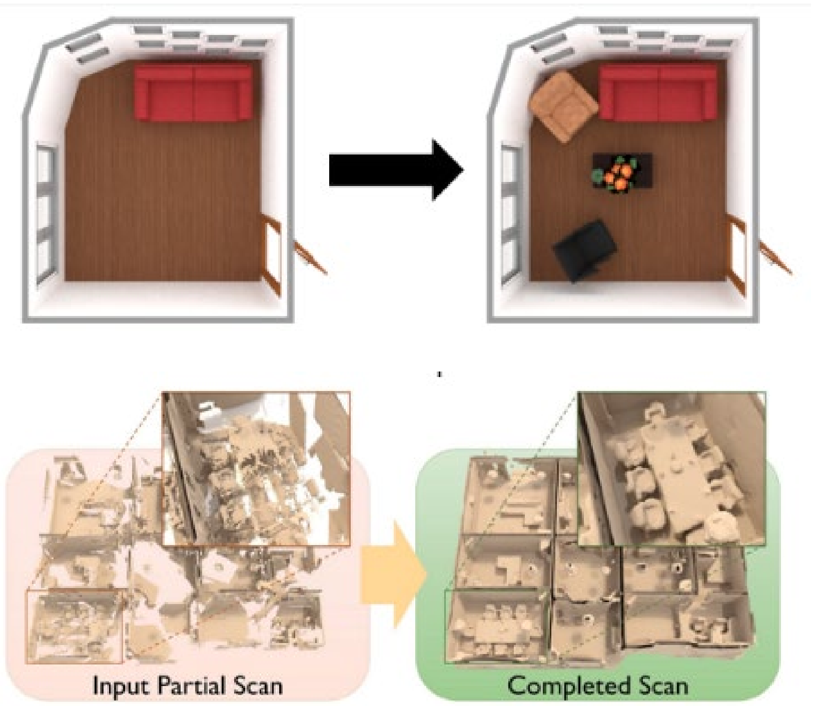
3D Scene Generation and Synthesis



Scene Generation



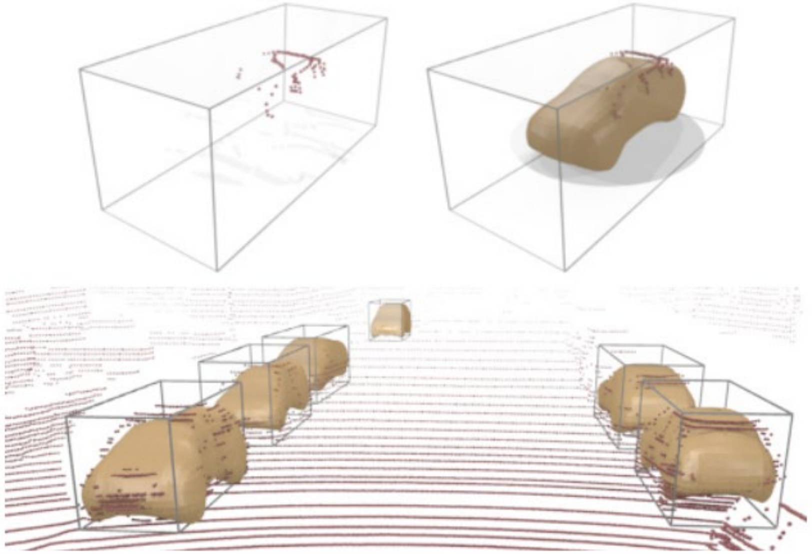
Scene Reconstruction



Scene Completion

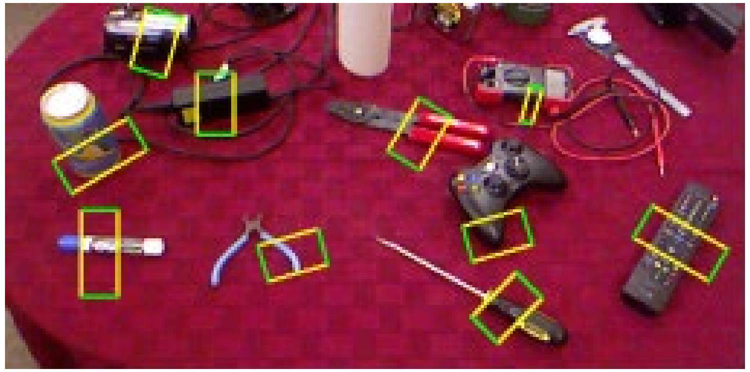
3D Deep Learning Tasks

Autonomous Driving Applications



3D Deep Learning Tasks

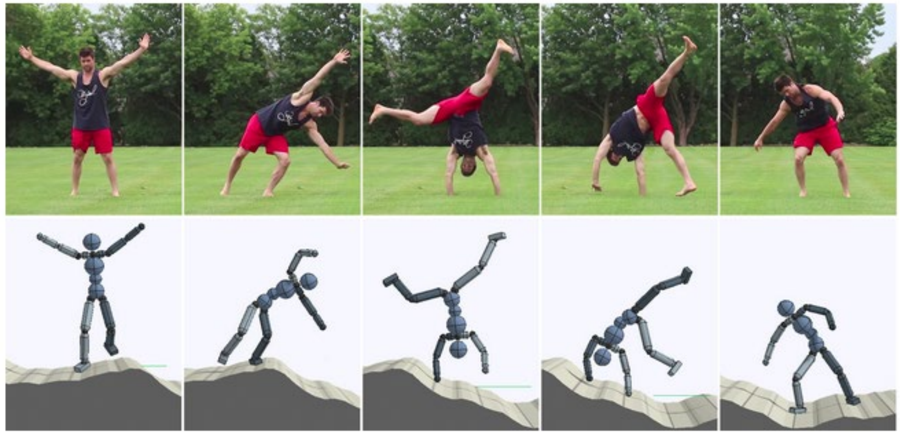
Robotics Applications

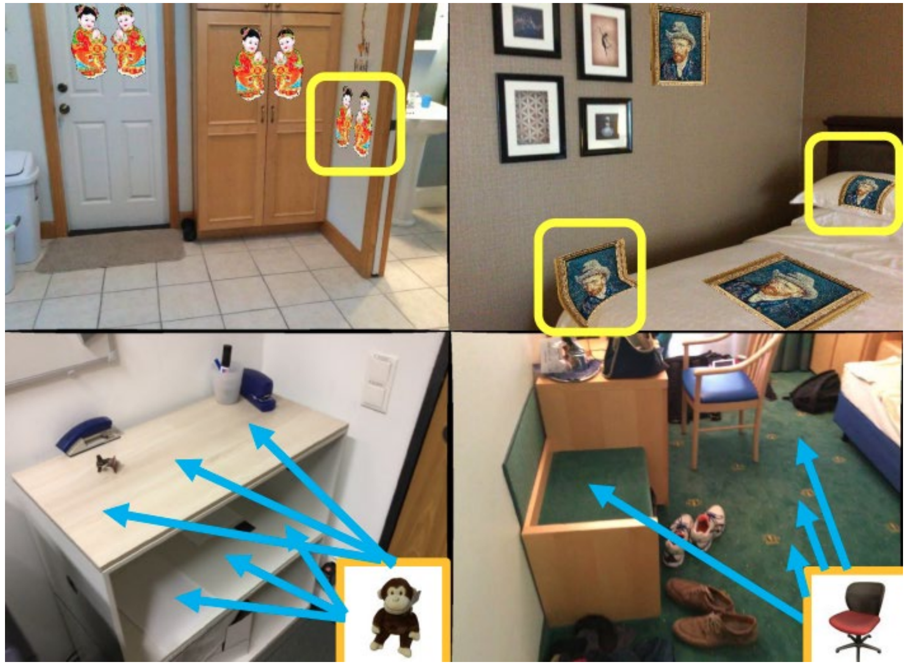


Robot Grasping Affordance Map

Human Interaction Affordance Map

Metaverse Applications





3D Voxel CNNs

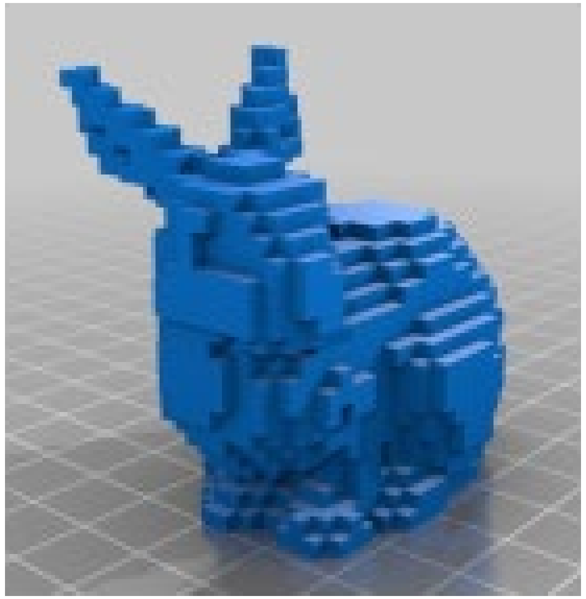
(DL / NNs over the simplest / most regular 3D representation)

Volumetric Representation: 2D Images

Images: canonical representation with regular data structure

1	44	33	12	20	23	35	14
51	16	40	32	46	48	28	17
29	60	3	63	49	55	36	7
52	22	26	41	38	10	61	53
2	24	19	11	34	43	5	8
57	9	37	42	25	21	27	18
30	56	50	64	4	59	6	13
58	47	45	31	39	15	62	54

Volumetric Representation: 3D Geometry



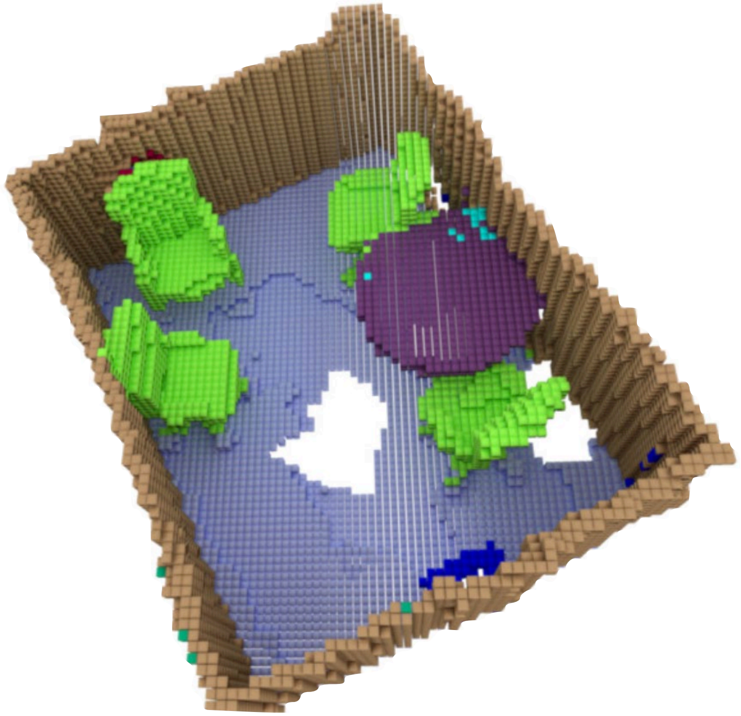
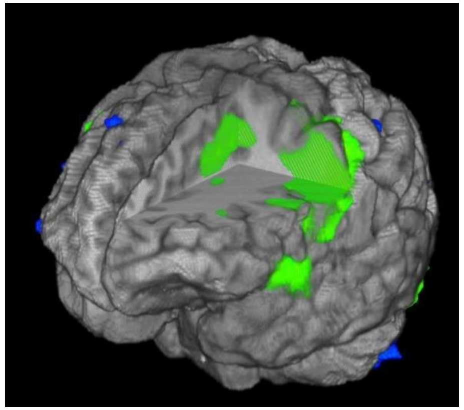
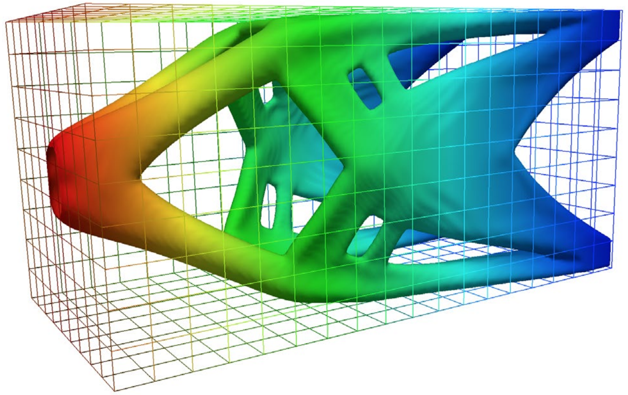


Image Credits: Scannet

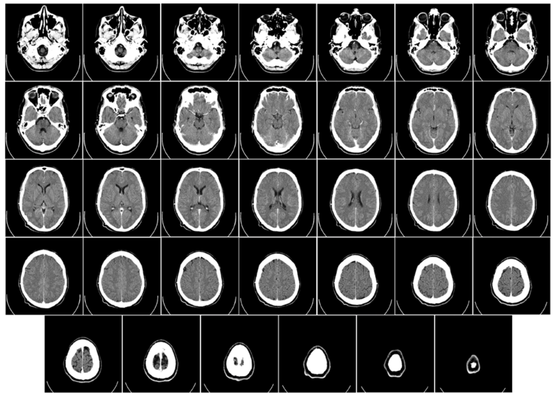
Volumetric Representation: Applications



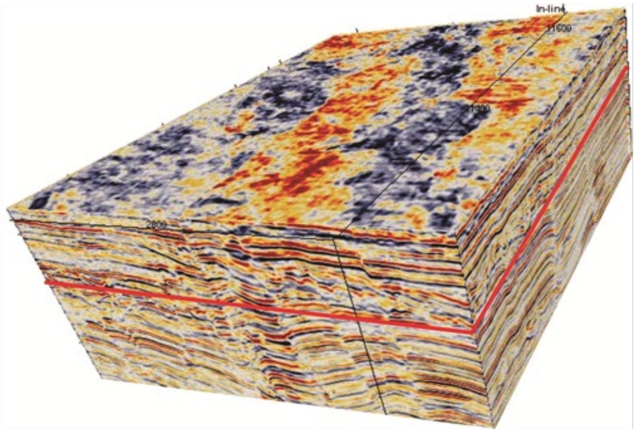
fMRI



Manufacturing
(finite-element analysis)

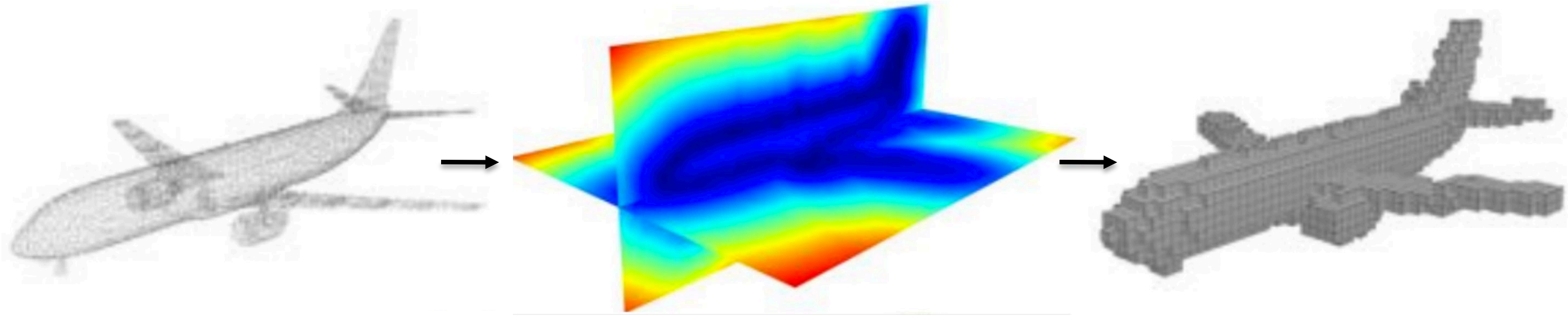


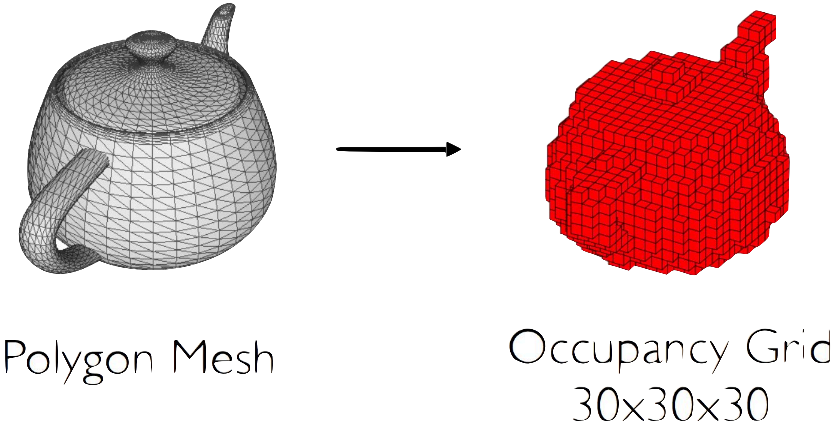
CT



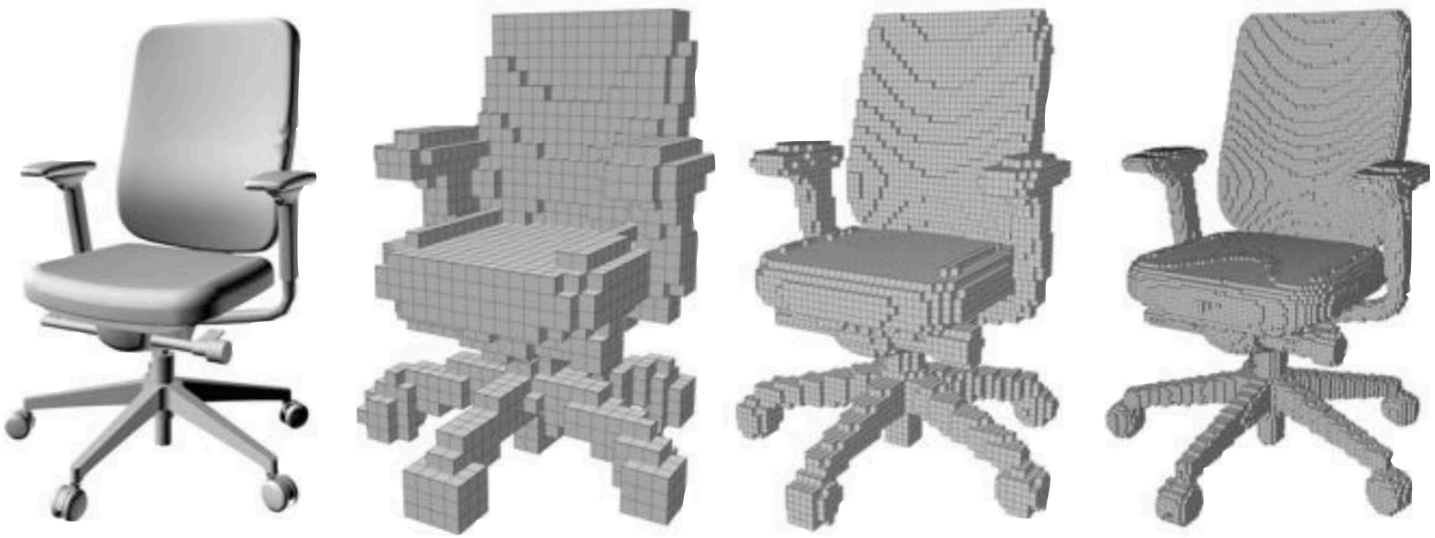
Geology

Volumetric Representation: Conversions





Volumetric Representation

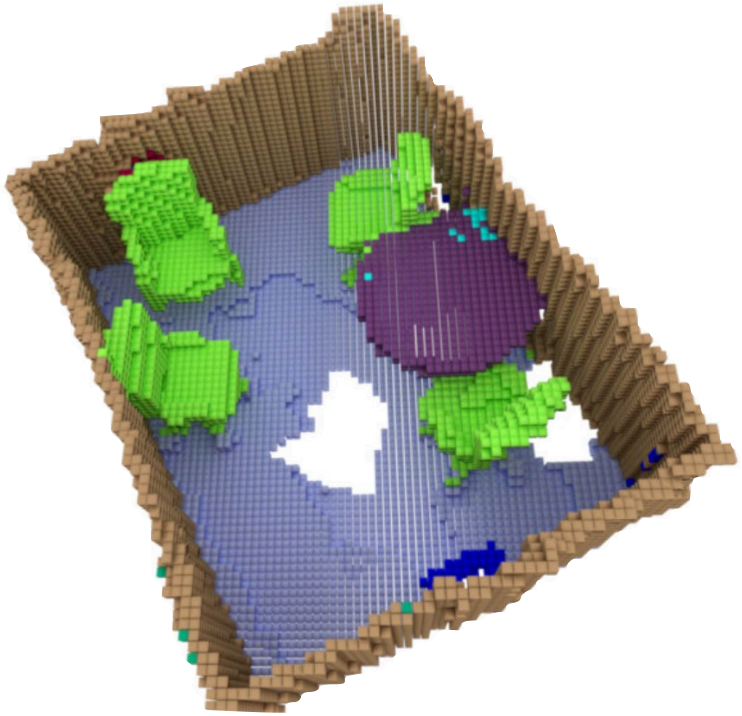


$\frac{\#occupied\ grid}{\#total\ grid}$ Occupancy: 10.41% 5.09% 2.41%

Resolution: 32 64 128

Resolution N O(N^3)

Volumetric Representation

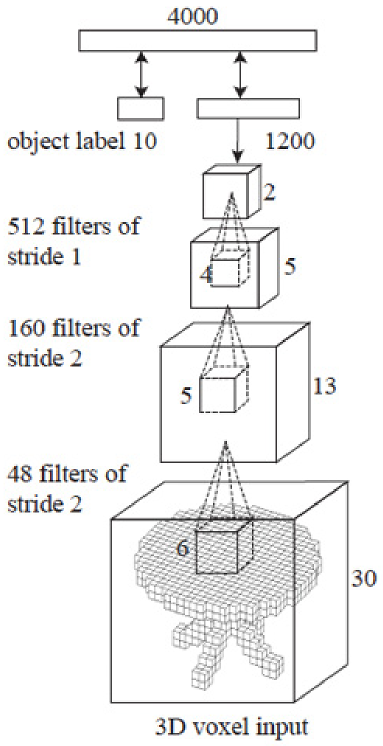


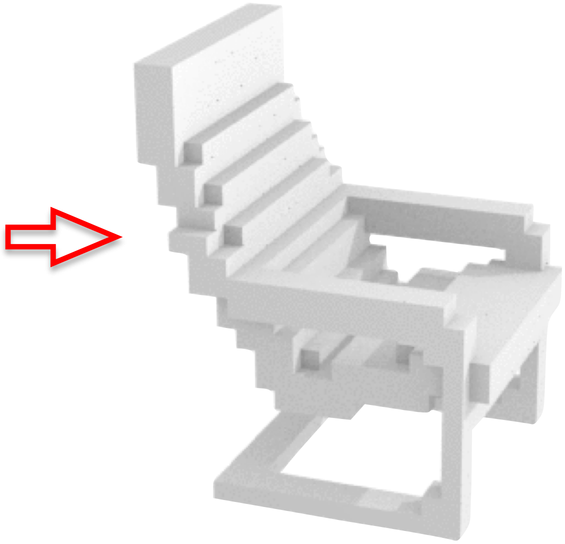
< 1%

Image Credits: Scannet

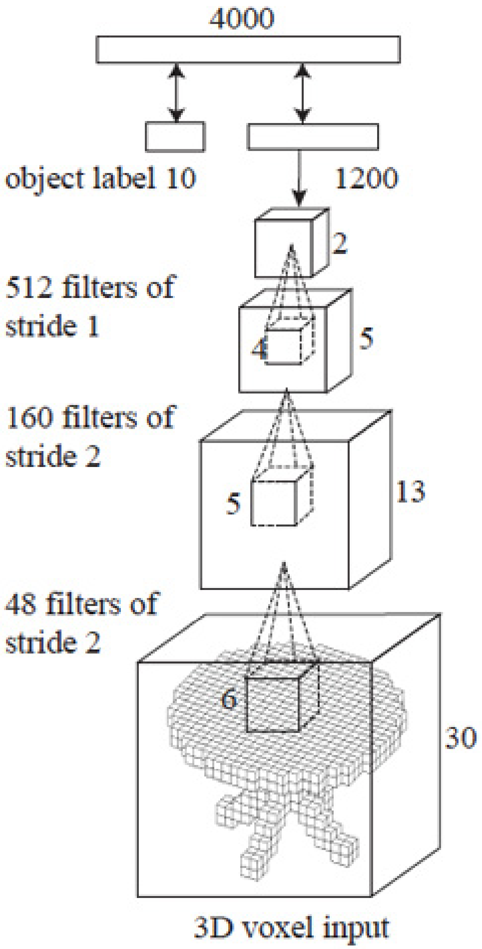
Voxel CNNs for 3D Data Understanding

- Classification (Recognition)
- Reconstruction (Generation)



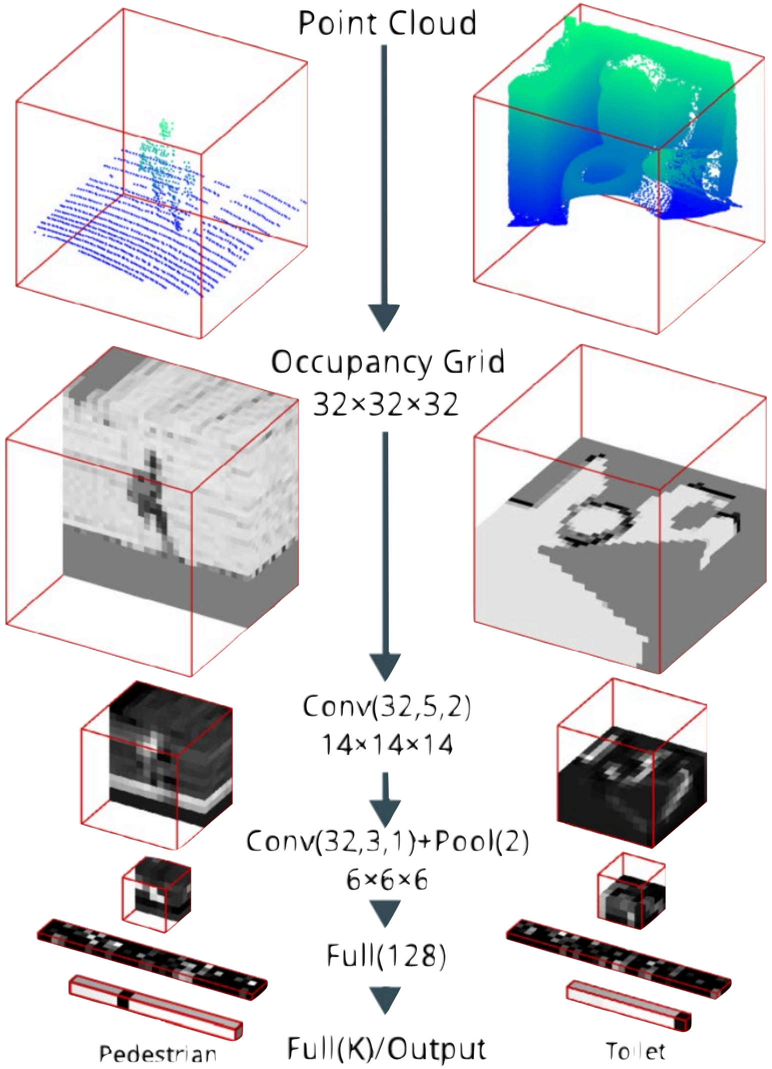


3D Voxel CNNs: Classification



Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang and J. Xiao, **"3D ShapeNets: A Deep Representation for Volumetric Shape Modeling",** *CVPR2015*

3D Voxel CNNs: Classification

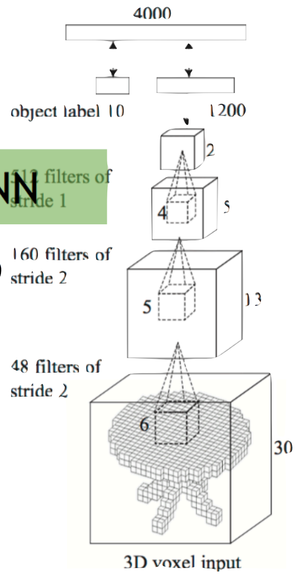


Daniel Maturana and Sebastian Scherer, **"VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition",** *IROS2015*

3D Voxel CNNs: Classification

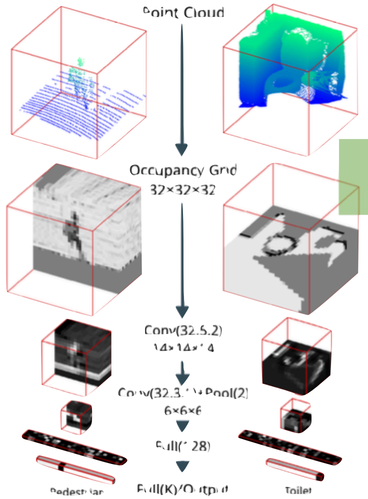
3DShapeNets from Princeton
CVPR 2015 Oral

VoxNet from CMU Robotics
IEEE/RSJ 2015



3D CNN

77.3%



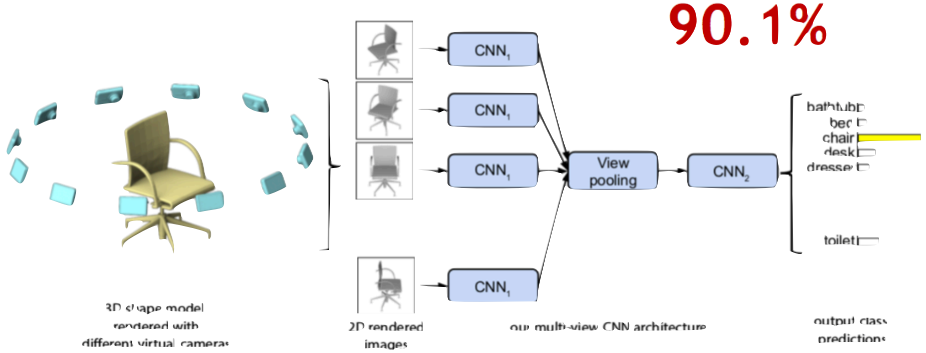
3D CNN

83.0%

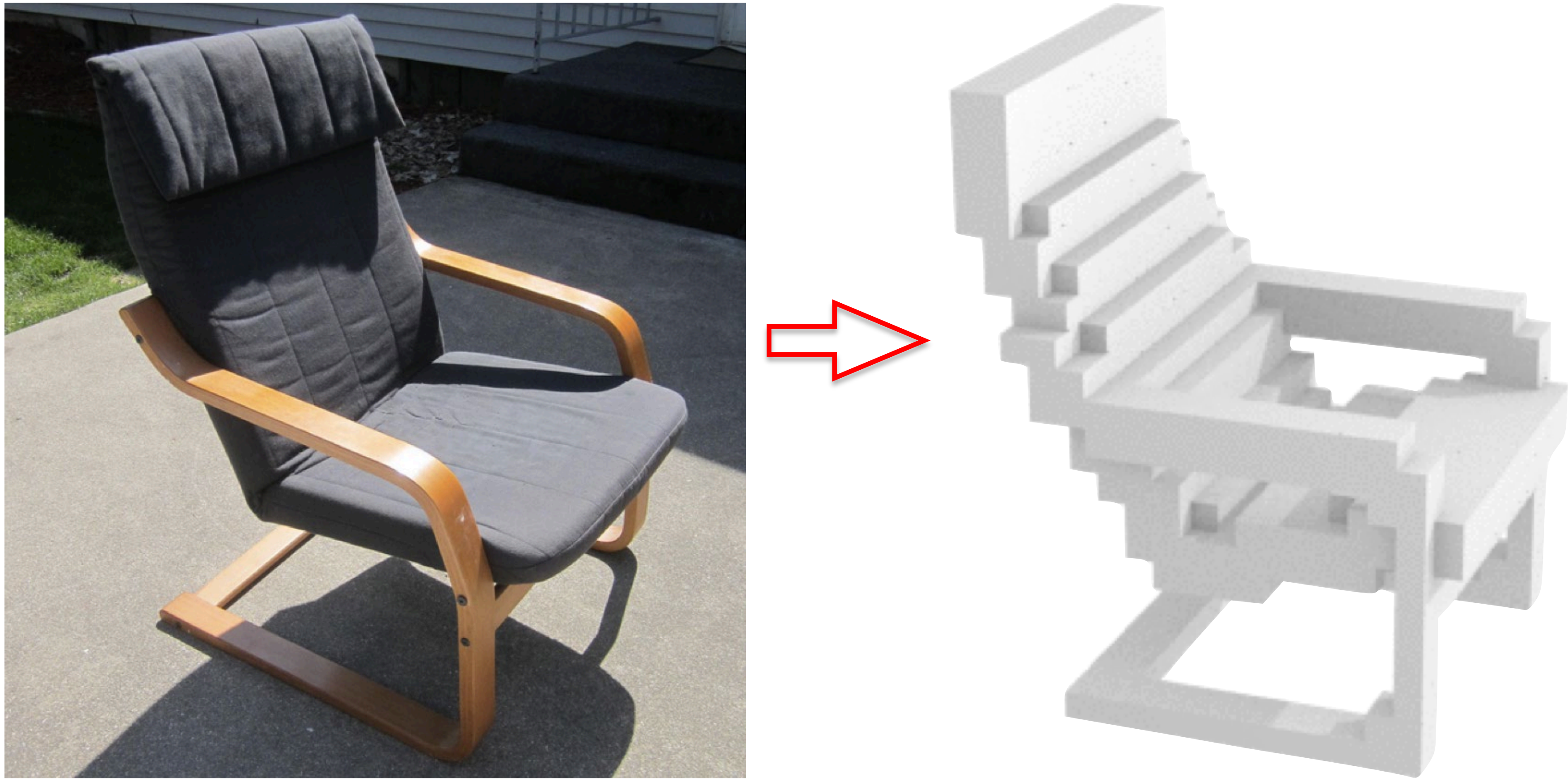
Rendering + 2D CNN

90.1%

MVCNN from UMass
ICCV 2015



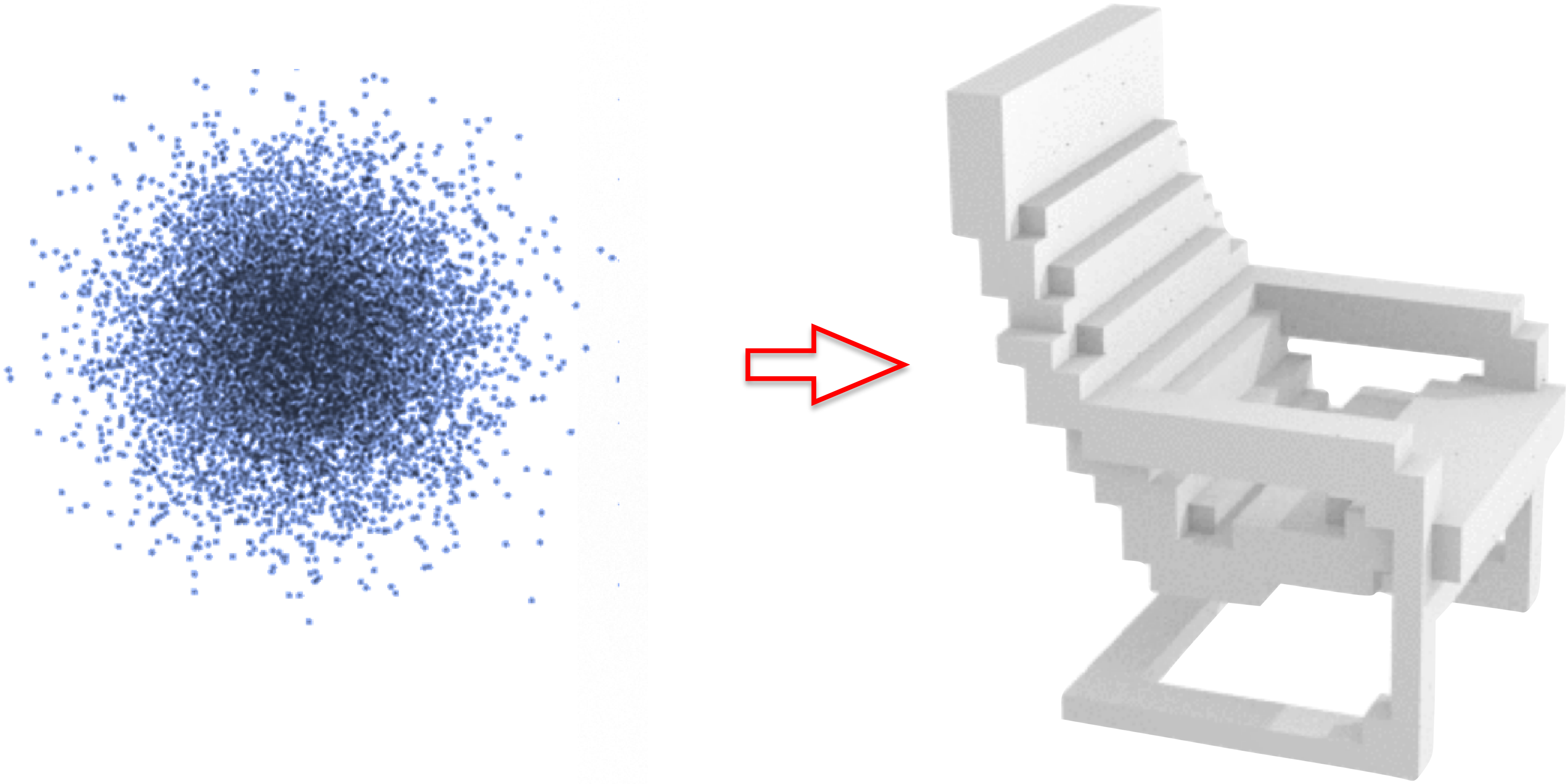
3D Voxel CNNs: Reconstruction



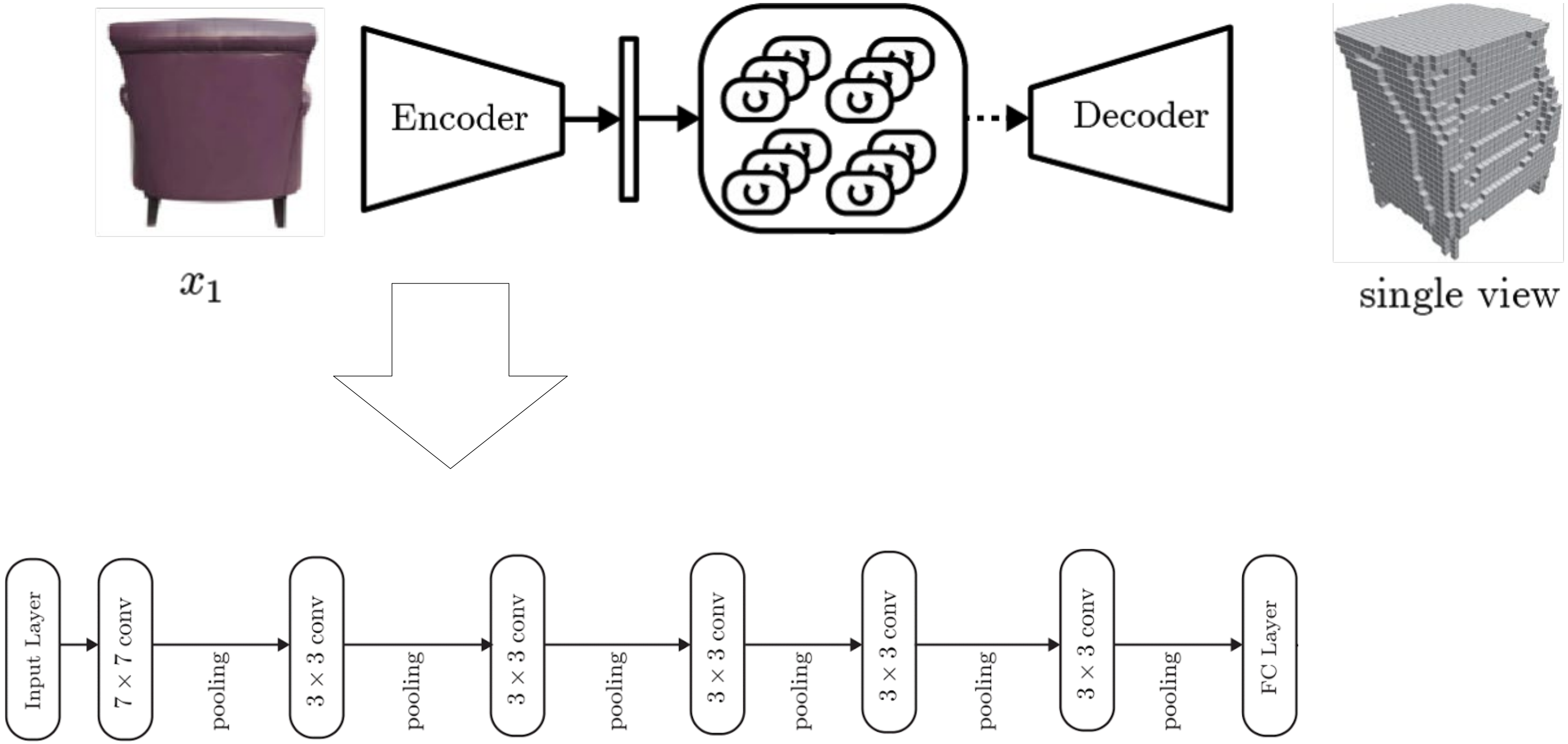
3D Voxel CNNs: Completion

Image Credits: Angela Dai et. al.

3D Voxel CNNs: Generation

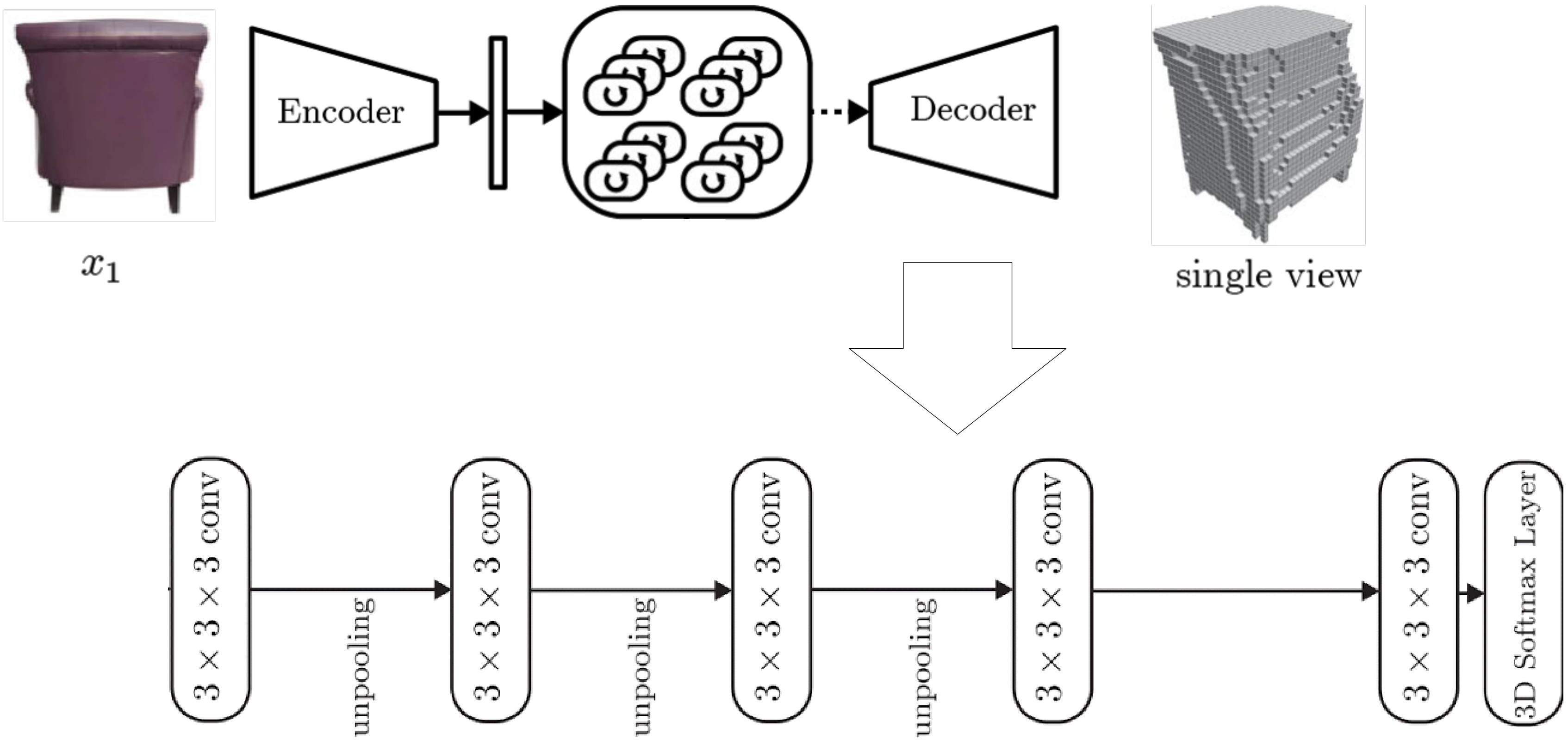


3D Voxel CNNs: Reconstruction



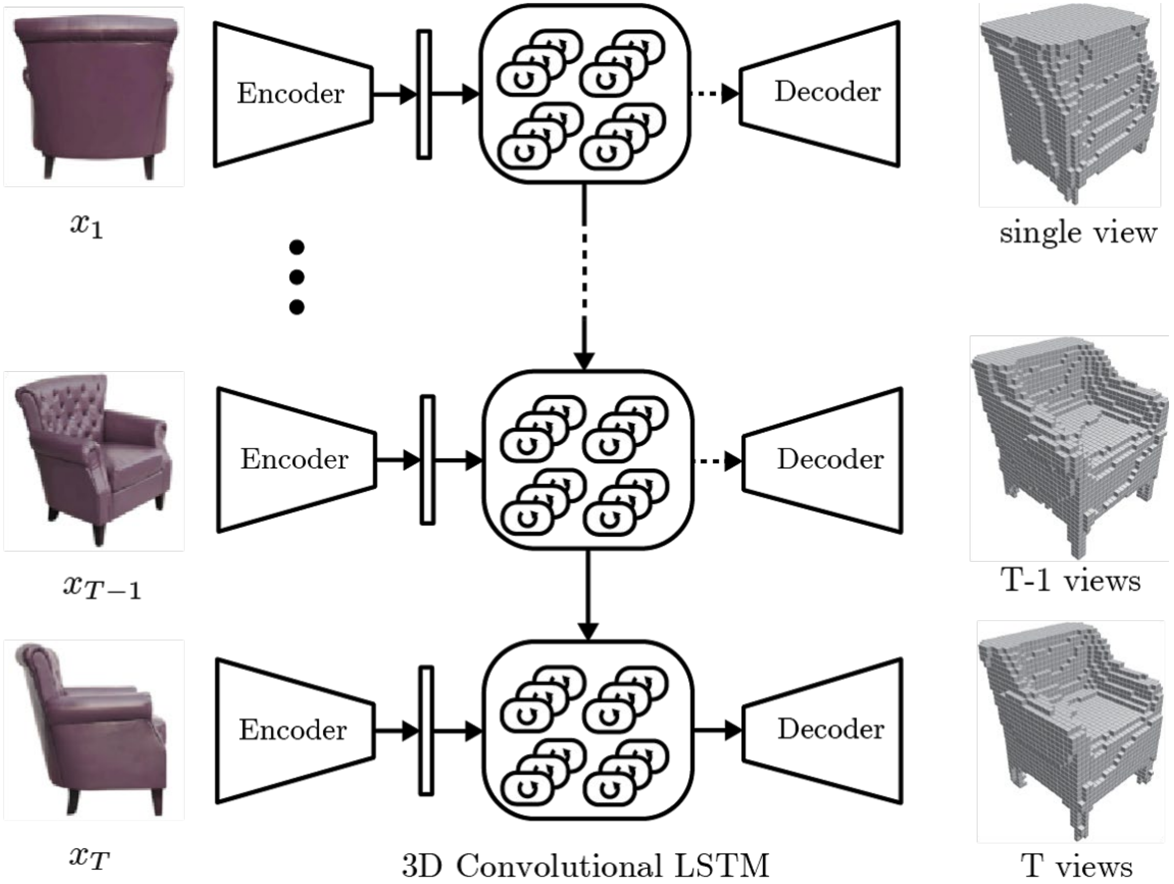
3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction
Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, Silvio Savarese
ECCV 2016

3D Voxel CNNs: Reconstruction



3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction
Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, Silvio Savarese
ECCV 2016

3D Voxel CNNs: Reconstruction



3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction
Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, Silvio Savarese
ECCV 2016

3D Voxel CNNs: Reconstruction

3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction
Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, Silvio Savarese
ECCV 2016

3D Voxel CNNs: Completion

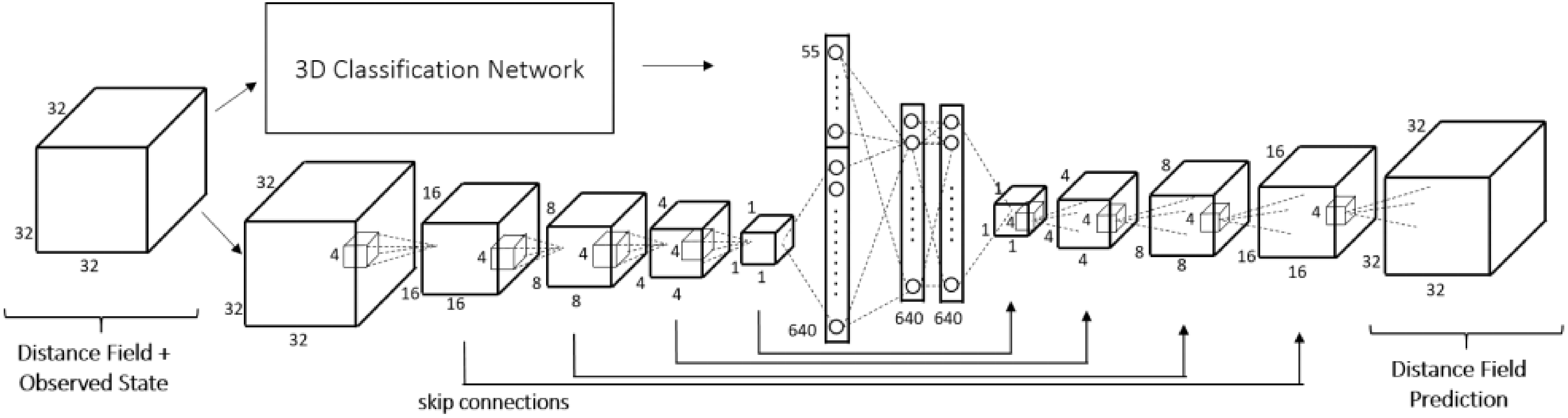
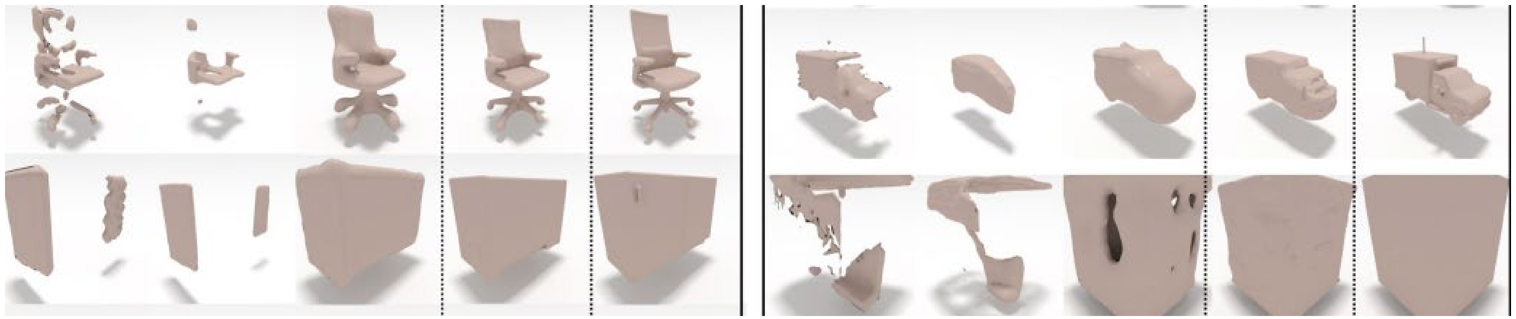
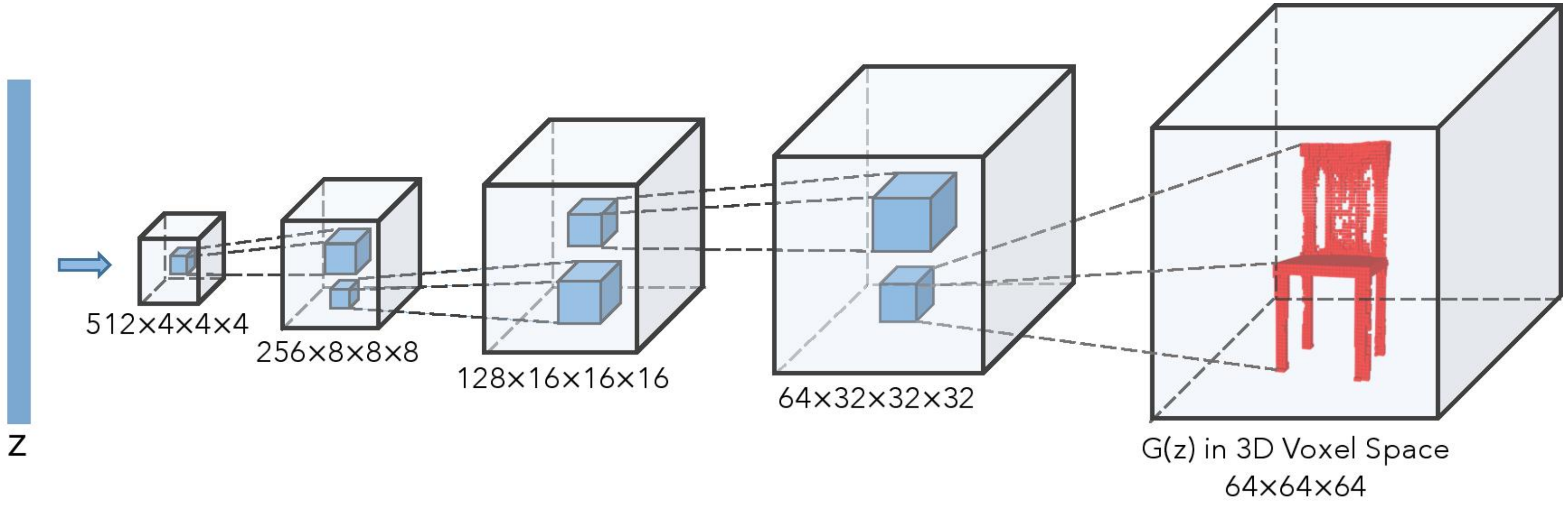


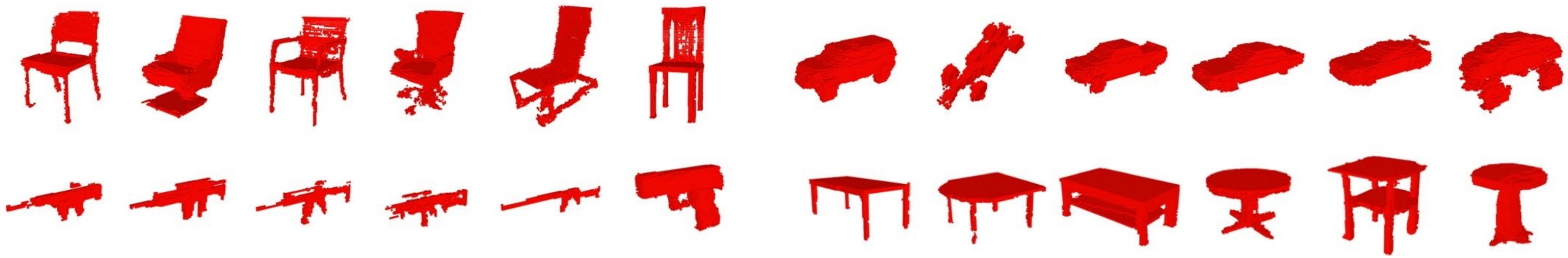
Figure 1: Network architecture of our 3D Encoder-Predictor Network.



Shape Completion using 3D-Encoder-Predictor CNNs and Shape Synthesis
Angela Dai, Charles R. Qi, Matthias Niessner
CVPR2017

3D Voxel CNNs: Generation





Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling
Jiajun Wu*, Chengkai Zhang*, Tianfan Xue, William T. Freeman, and Joshua B. Tenenbaum
NeurIPS2016

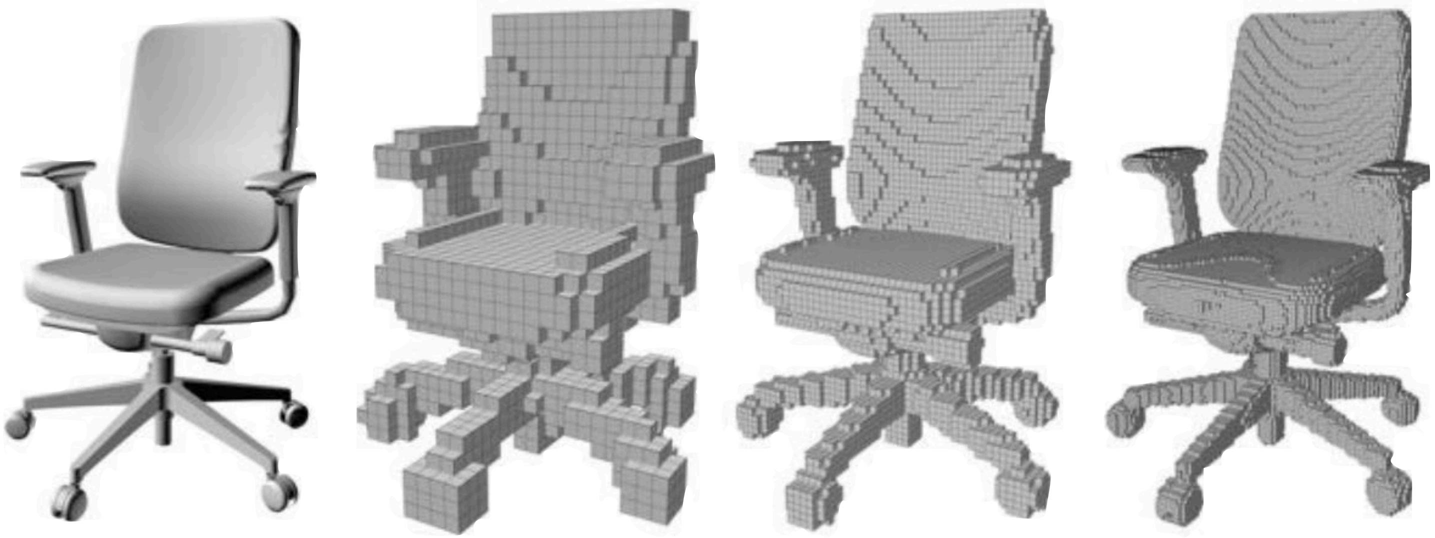
Volumetric Representation

- Pros
  - Regular grids representation
  - Intuitive extension of images
  - Easy to input to Neural Nets
  - Each grid can have many feature inputs
- Cons
  - Need to convert from point clouds scans
  - Surface voxels? / solid voxels?
  - Space / Time Complexities

Hierarchical 3D Voxel CNNs

(Use Hierarchical Architectures to Alleviate the Curse of Dimensionality)

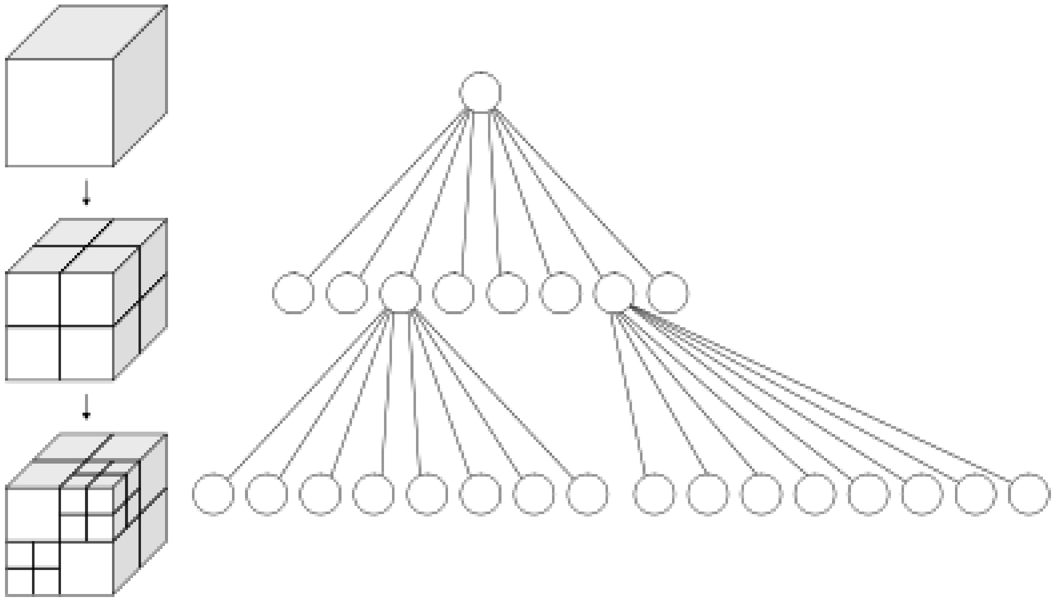
3D Voxel Data: Curse of Dimensionality



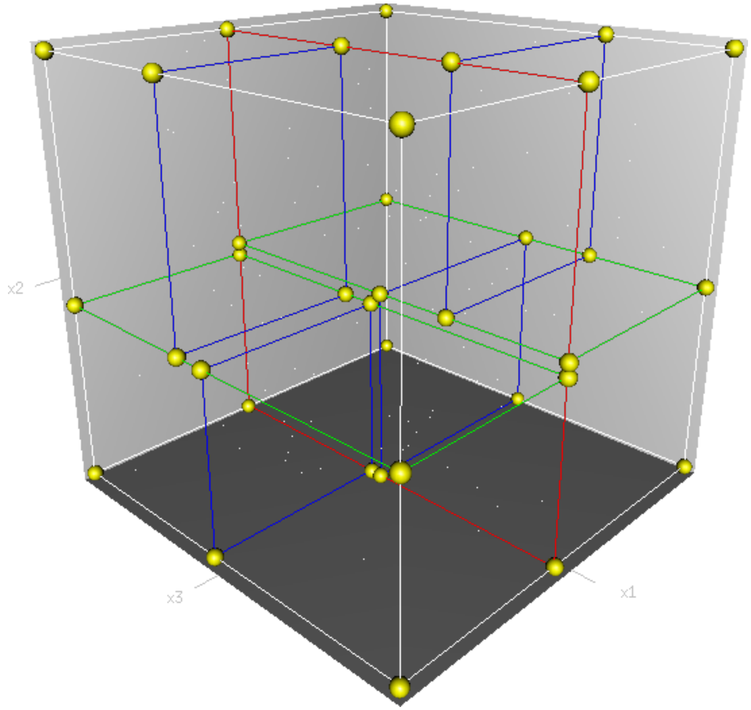
		32	64	128
$\frac{\#occupied\ grid}{\#total\ grid}$	Occupancy:	10.41%	5.09%	2.41%
	Resolution:	32	64	128

Resolution N O(N^3)

Hierarchical Volumetric Representation



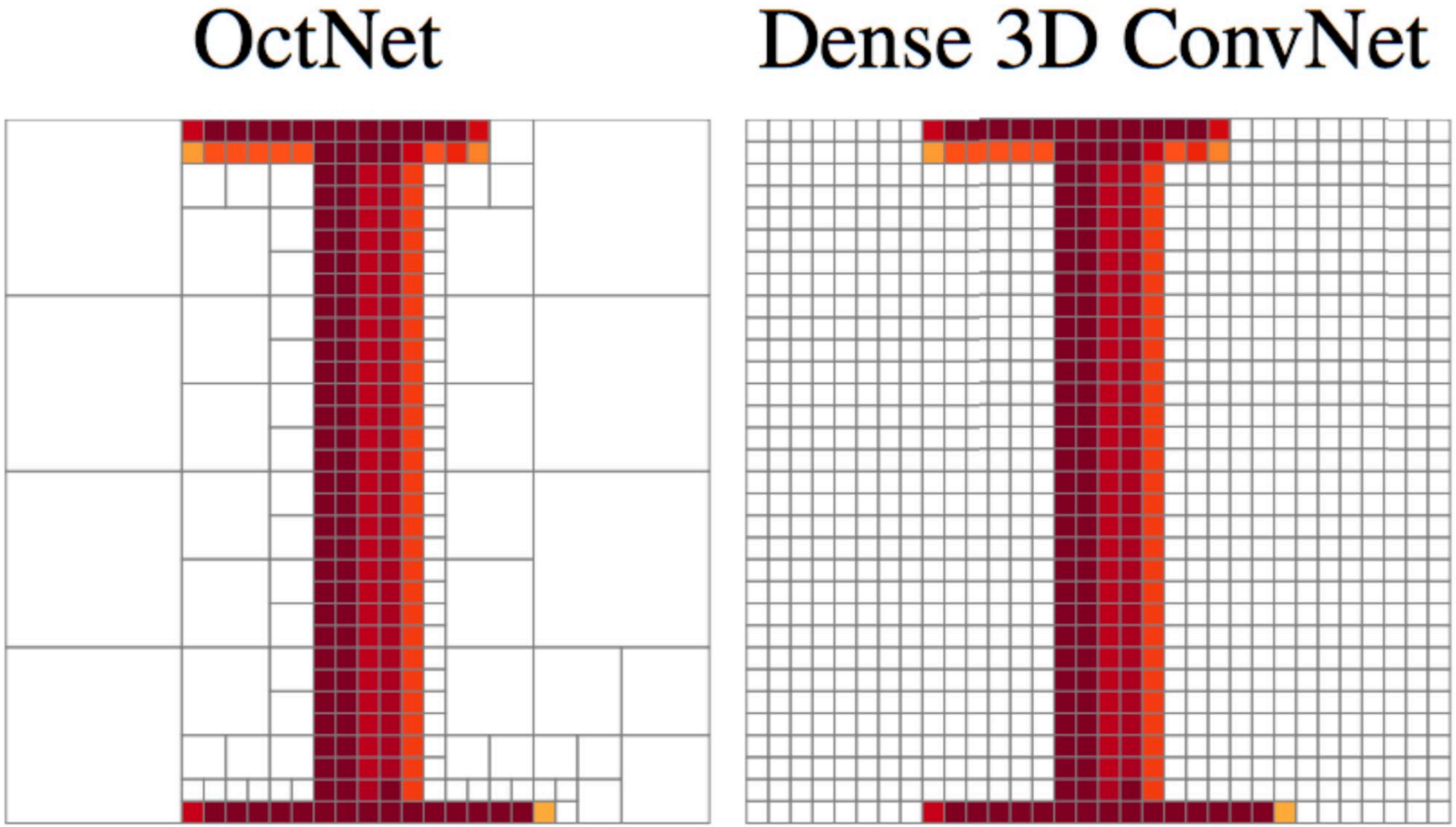
Octree



3D KD-Tree

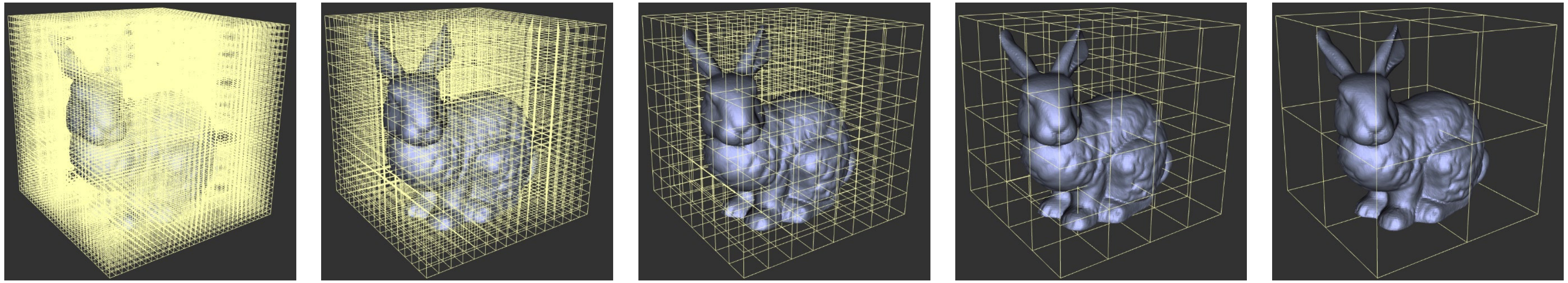
Image Credits: Wikipedia

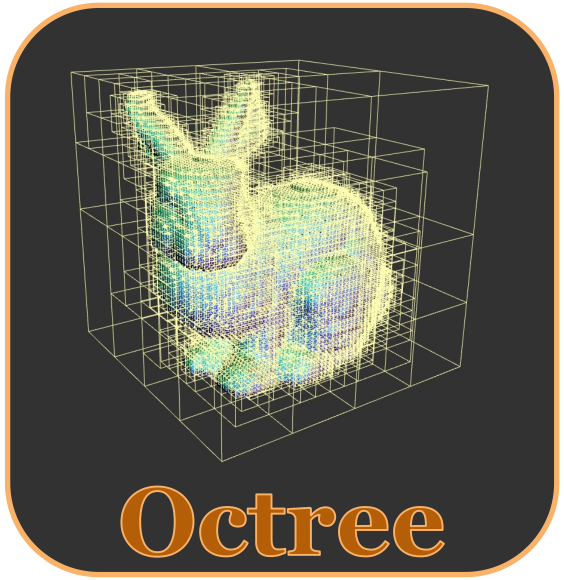
Hierarchical Volumetric Representation



OctNet: Learning Deep 3D Representations at High Resolutions
Gernot Riegler, Ali Osman Ulusoy, Andreas Geiger, CVPR 2017

Hierarchical Volumetric Representation



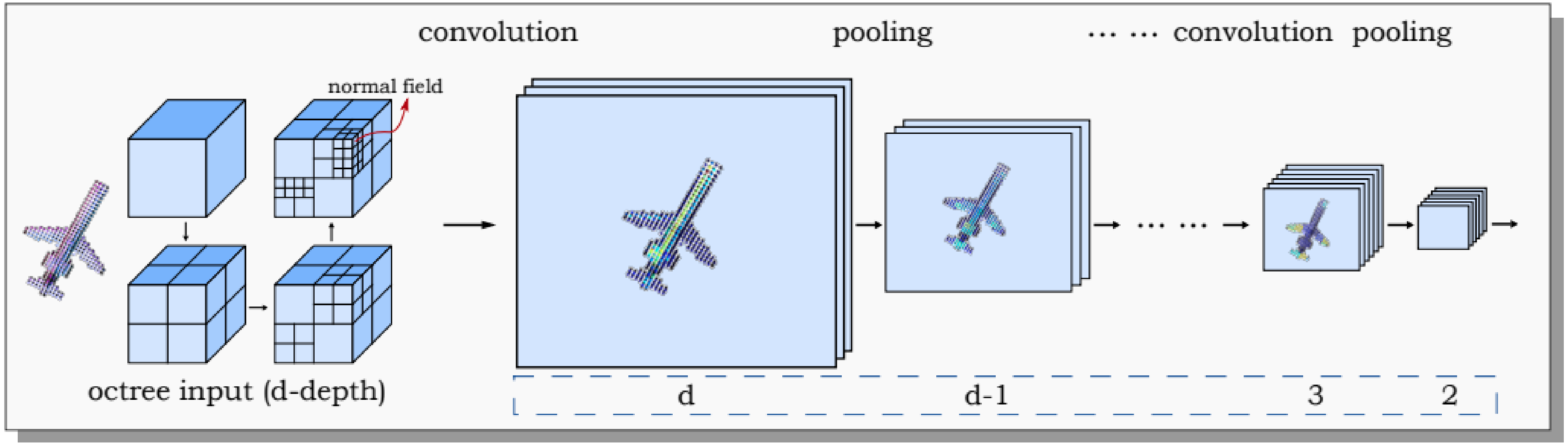


Hier 3D Voxel NN: Recognition

O-CNN: Octree-based Convolutional Neural Networks for 3D Shape Analysis

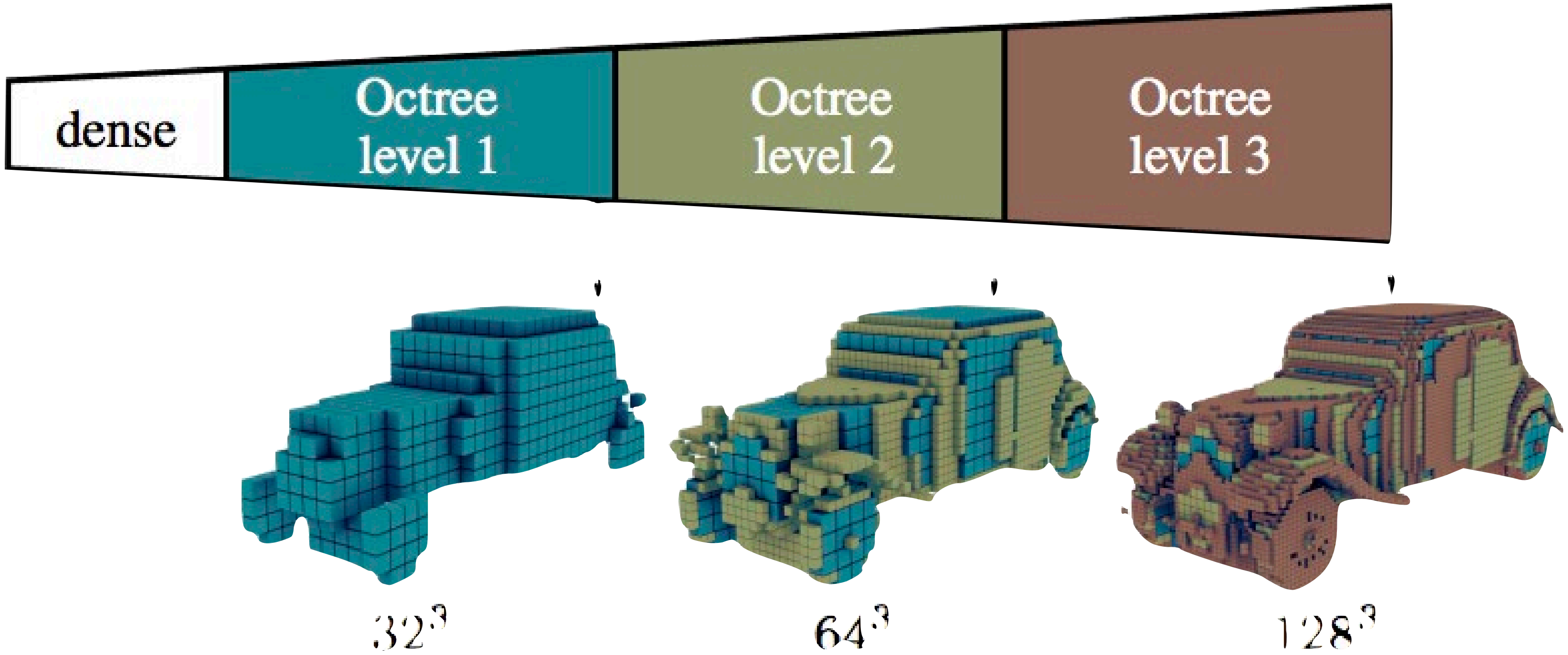
SIGGRAPH 2017

PENG-SHUAI WANG, Tsinghua University and Microsoft Research Asia
YANG LIU, Microsoft Research Asia
YU-XIAO GUO, University of Electronic Science and Technology of China and Microsoft Research Asia
CHUN-YU SUN, Tsinghua University and Microsoft Research Asia
XIN TONG, Microsoft Research Asia



Classification Accuracy: 89.9%

Hier 3D Voxel NN: Generation

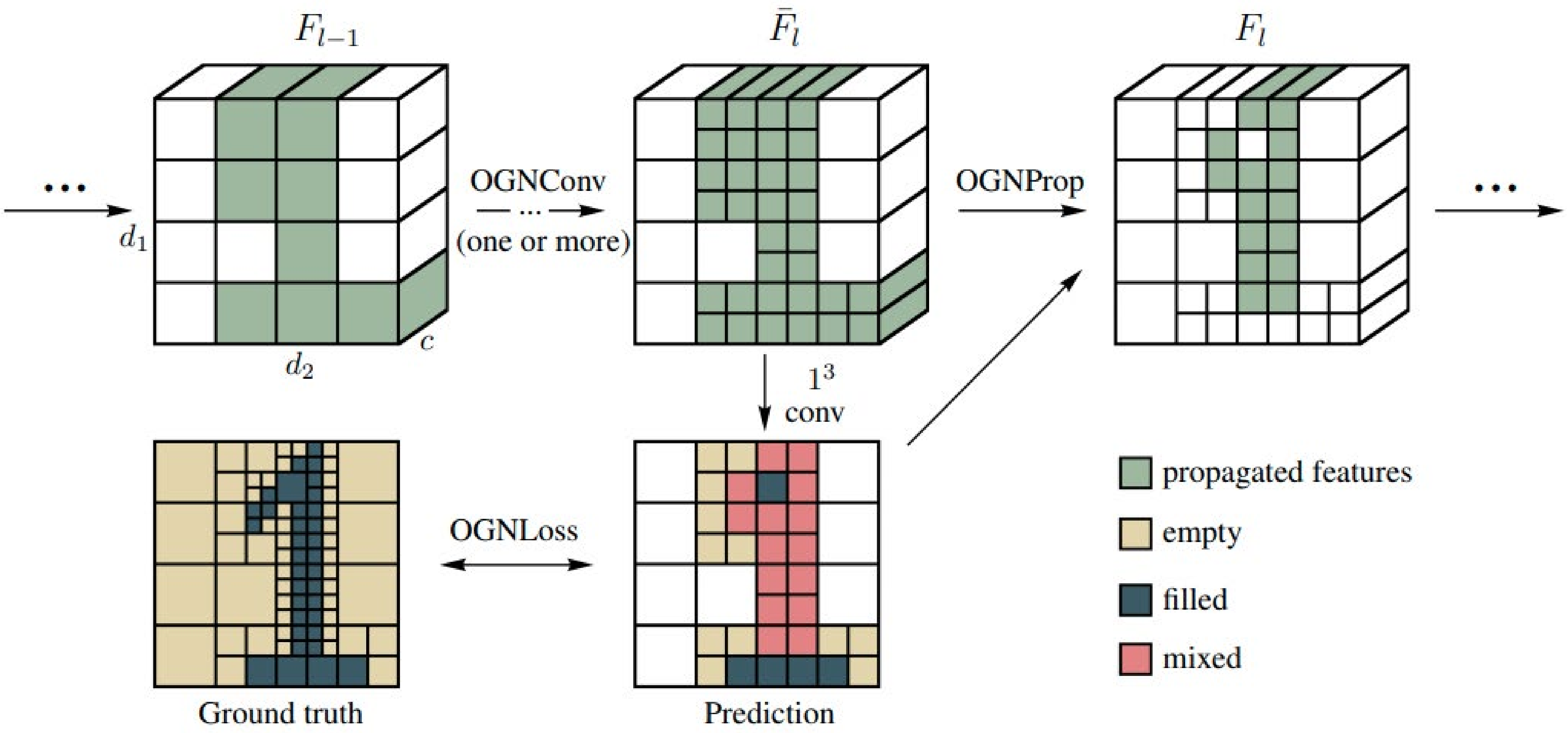


Maxim Tatarchenko, Alexey Dosovitskiy, Thomas Brox

"Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs"

ICCV, 2017

Hier 3D Voxel NN: Generation

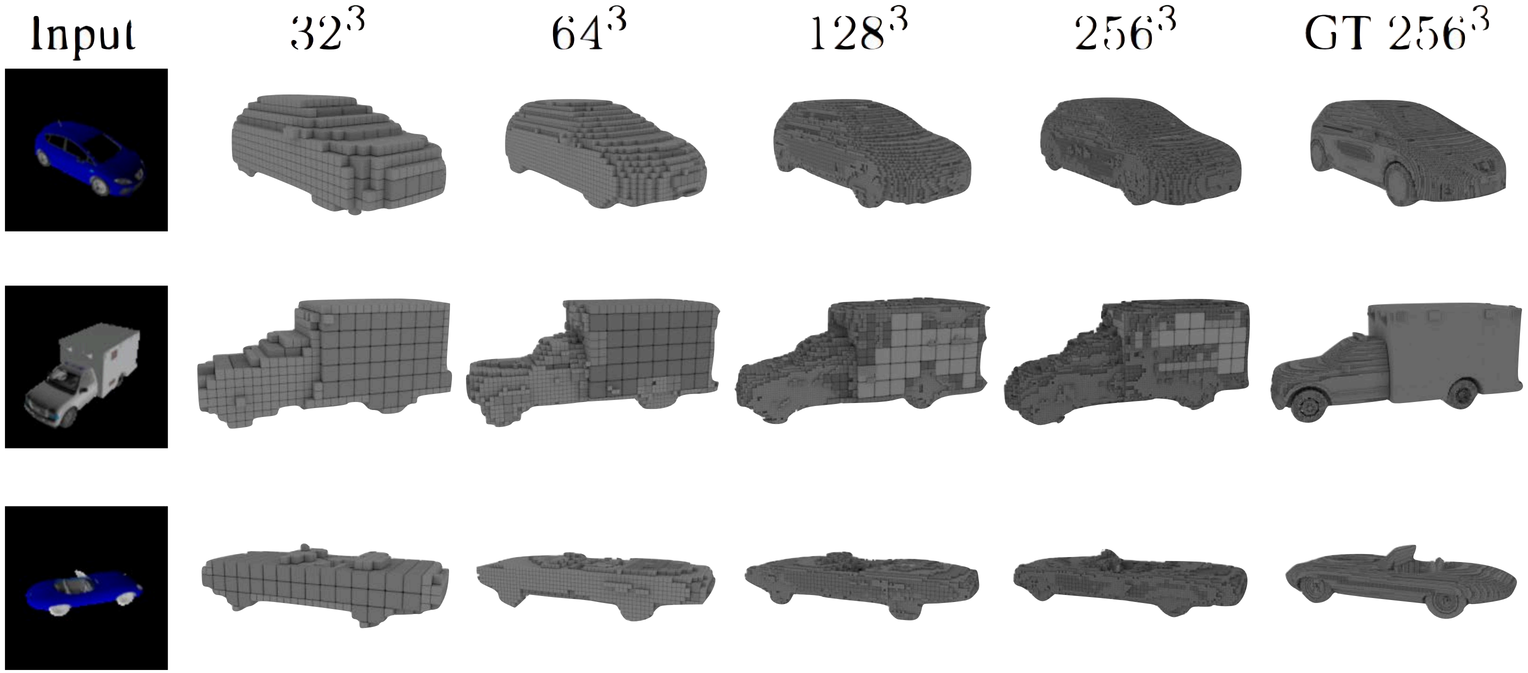


Maxim Tatarchenko, Alexey Dosovitskiy, Thomas Brox

"Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs"

ICCV, 2017

Hier 3D Voxel NN: Generation



Maxim Tatarchenko, Alexey Dosovitskiy, Thomas Brox

"Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs"

ICCV, 2017

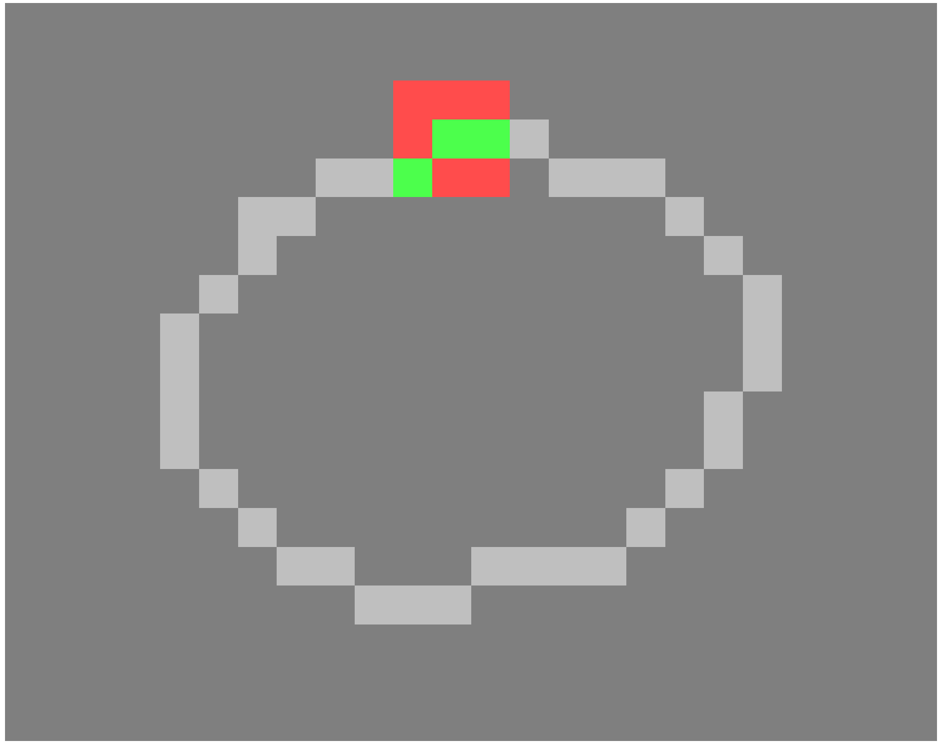
Hierarchical 3D Voxel NNs

- Pros
  - Address the O(N^3) Storage / Computation Explosion
  - Allow 3D CNNs to work with Larger N --> better performance
- Cons
  - Still have computations over empty voxels
  - Not very flexible given the specific data structure (e.g. the octree)

3D Sparse ConvNets

(Only Perform Convolution over Existing Voxels)

3D Sparse ConvNets

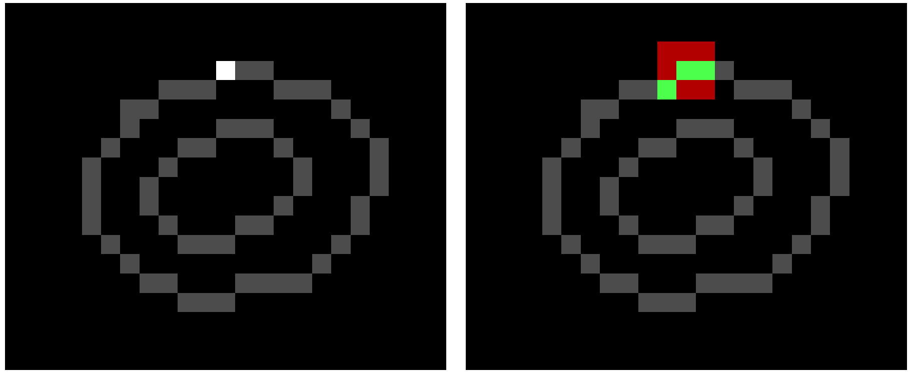


Submanifold Sparse Convolution (SSC)

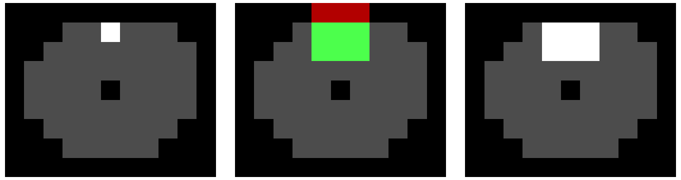
- carefully engineered that no computational overhead at the empty cells, using a ***hash-table*** and a ***rule-book***
- only computed when the kernel ***center*** is over an ***occupied*** cell

3D Semantic Segmentation with Submanifold Sparse Convolutional Networks
Benjamin **Graham, Martin Engelcke, Laurens van der Maaten,** CVPR, 2018

3D Sparse ConvNets



SSC with Stride 2
(Size 3)

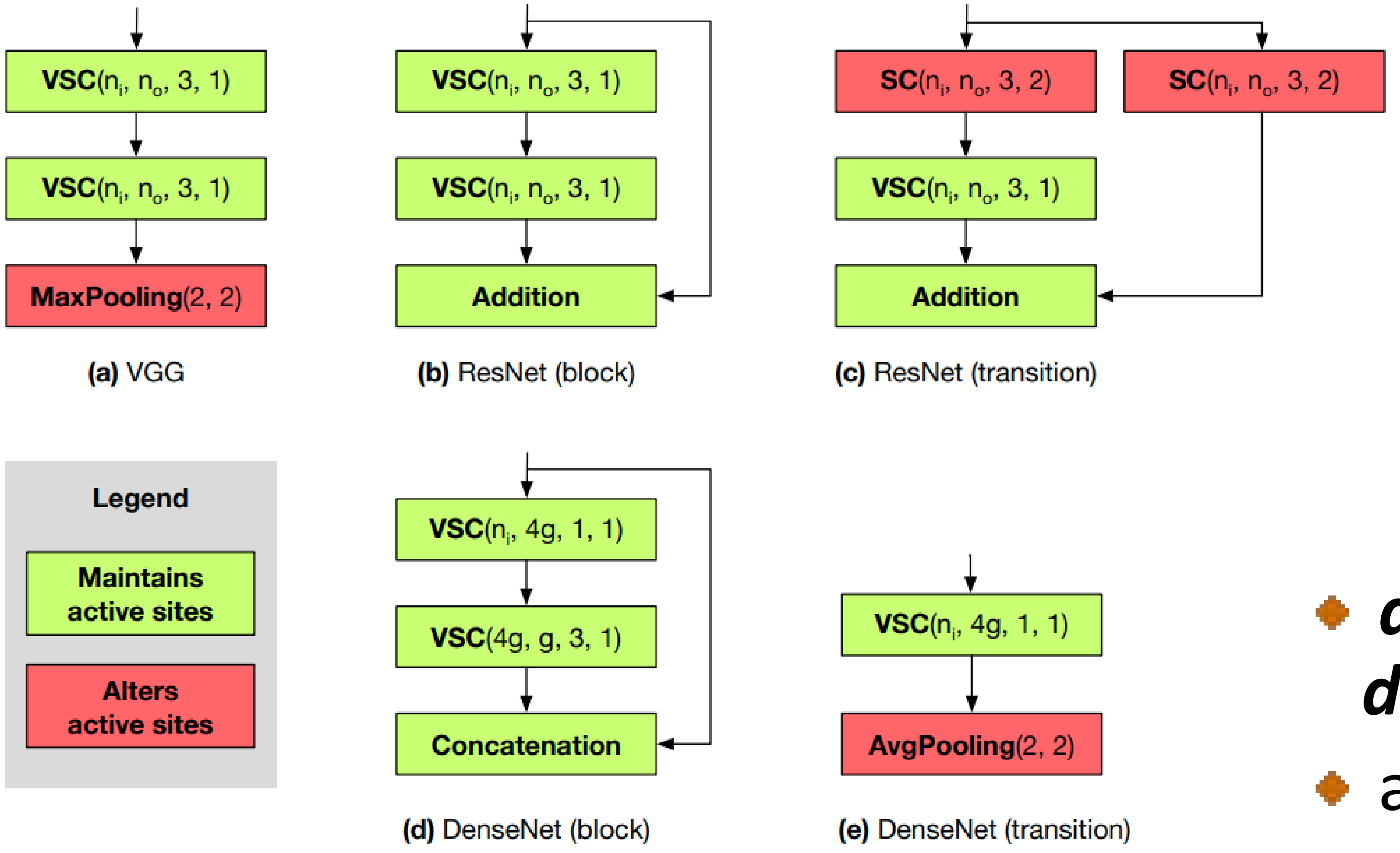


SSC with Stride 1
(Size 3)

- perform ***strided convolutions***, ***pooling operations***, or ***regular sparse convolutions*** (i.e. perform convolution over regions containing at least one occupied cell) to correlate disconnected components

3D Semantic Segmentation with Submanifold Sparse Convolutional Networks
***Benjamin* Graham, Martin Engelcke, Laurens van der Maaten,** CVPR, 2018

3D Sparse ConvNets

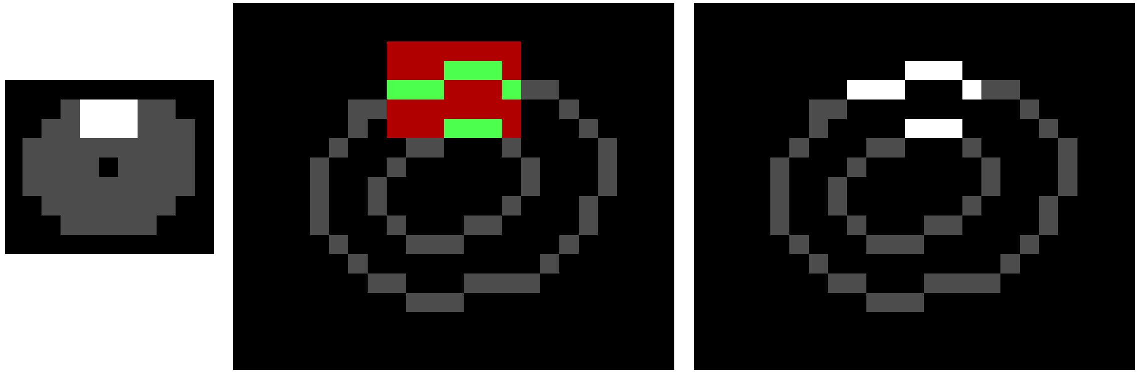


VSC: Submanifold Sparse Conv
SC: Regular Sparse Conv

- ***allow training 3D ConvNets as deep as the 3D counterparts!!!***
- and, as we expect, deeper networks often work better

3D Semantic Segmentation with Submanifold Sparse Convolutional Networks
***Benjamin* Graham, Martin Engelcke, Laurens van der Maaten,** CVPR, 2018

3D Sparse ConvNets



Deconvolution

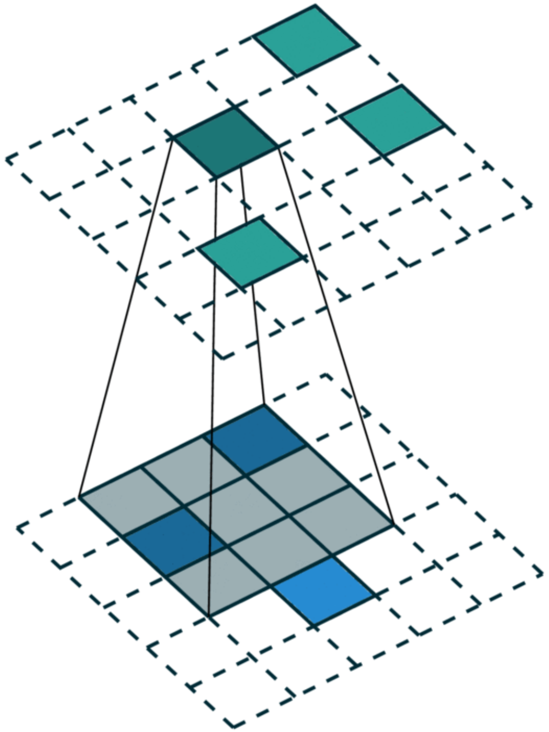
- ***deconvolution layers*** can also be implemented for Sparse ConvNet Decoders, which allows us to train 3D-Sparse-UNet for ***per-voxel labeling tasks*** (e.g. semantic segmentation)

3D Semantic Segmentation with Submanifold Sparse Convolutional Networks
***Benjamin* Graham, Martin Engelcke, Laurens van der Maaten,** CVPR, 2018

Minkowski Engine: A GenSparseConv System

- Implement generalized sparse convolution
- Generalize to 4D and higher dimensional data
- Cover the previous work Submanifold Sparse Conv as a special case

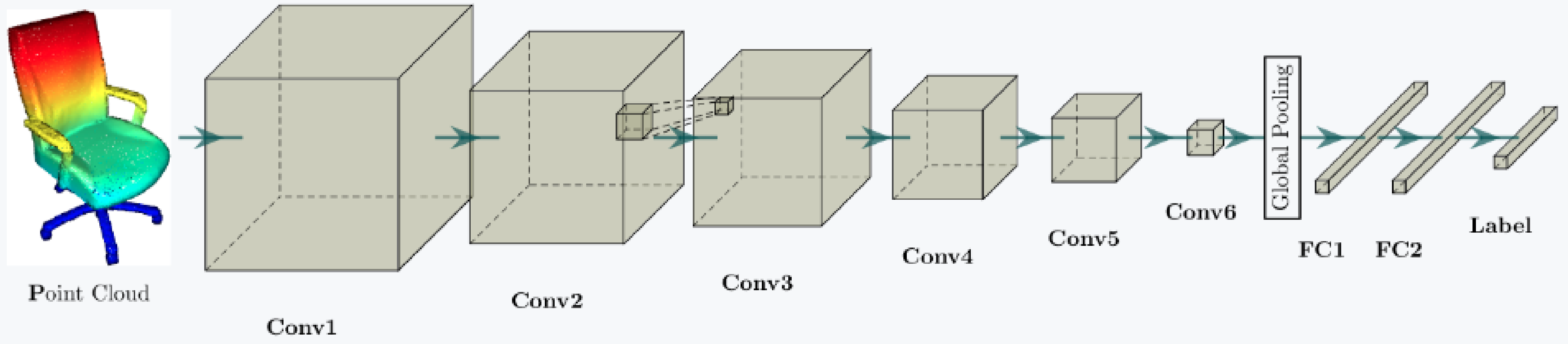
- An open-source, well-engineered, well-documented, auto-differentiation library



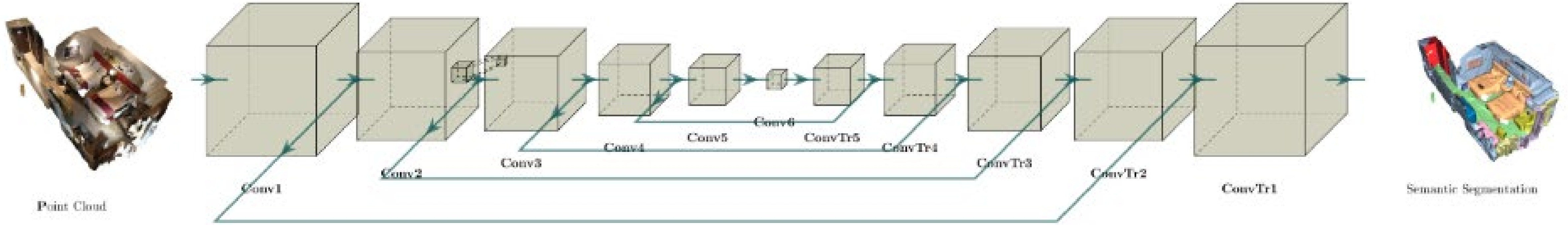
Sparse Convolution over Sparse Tensors

4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks
Christopher Choy, JunYoung Gwak, Silvio Savarese, CVPR, 2019

Minkowski Engine: Applications



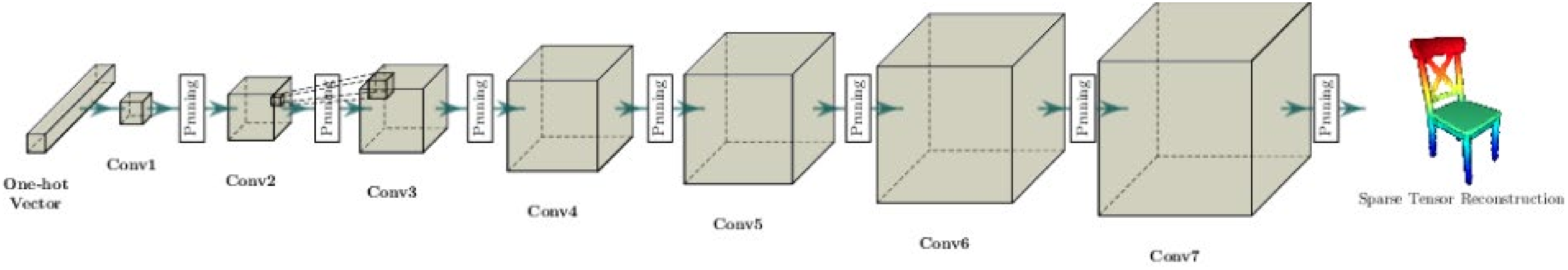
3D Shape Classification



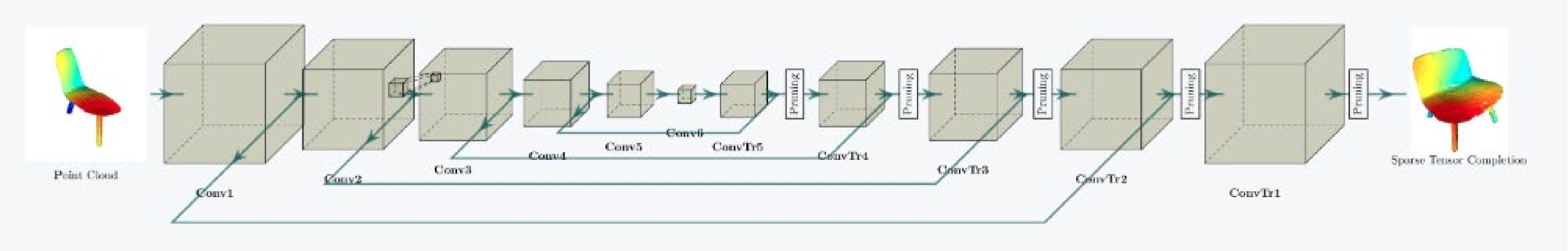
3D Scene Semantic Segmentation

4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks
Christopher Choy, JunYoung Gwak, Silvio Savarese, CVPR, 2019

Minkowski Engine: Applications



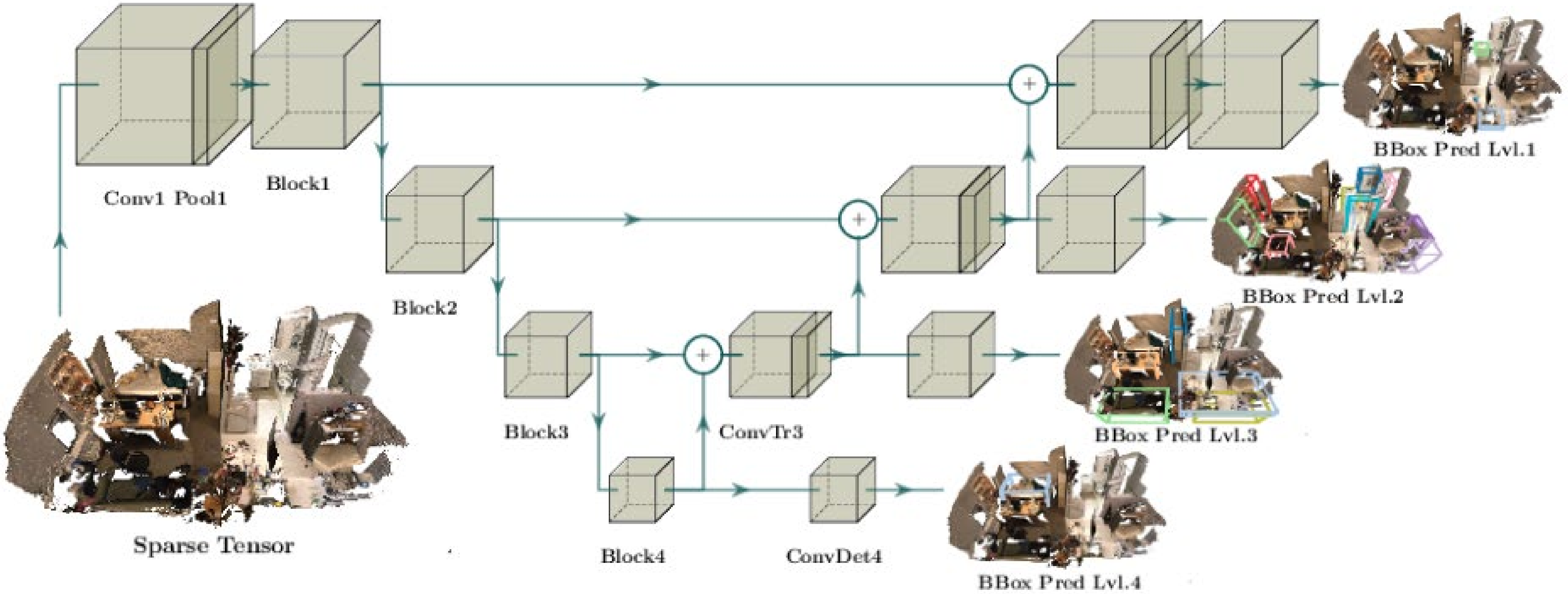
3D Shape Generation



3D Shape Completion

4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks
Christopher Choy, JunYoung Gwak, Silvio Savarese, CVPR, 2019

Minkowski Engine: Applications



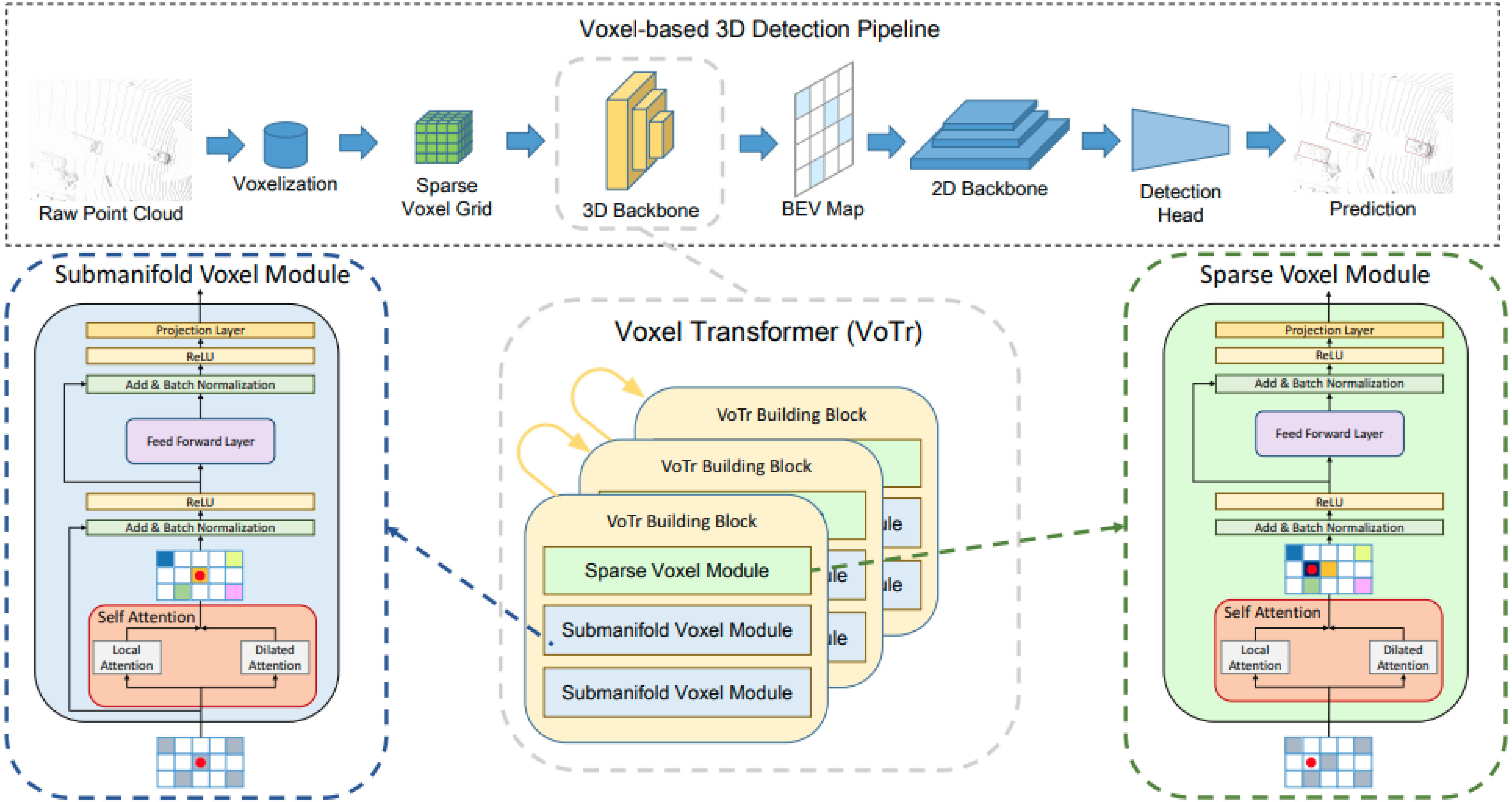
3D Scene Object Detection

4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks
Christopher Choy, JunYoung Gwak, Silvio Savarese, CVPR, 2019

3D Voxel Transformers

(Extend Visual Transformers to 3D Voxel Geometry)

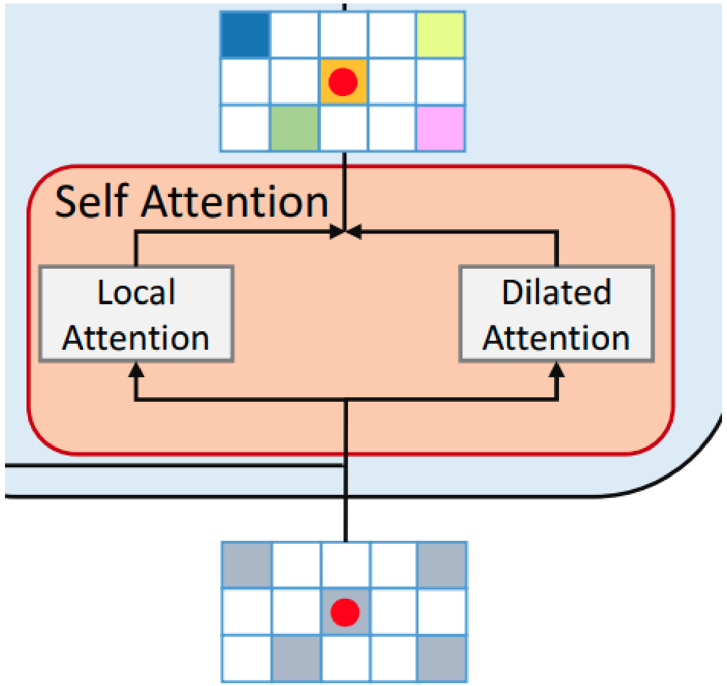
VoTr: Voxel Transformer

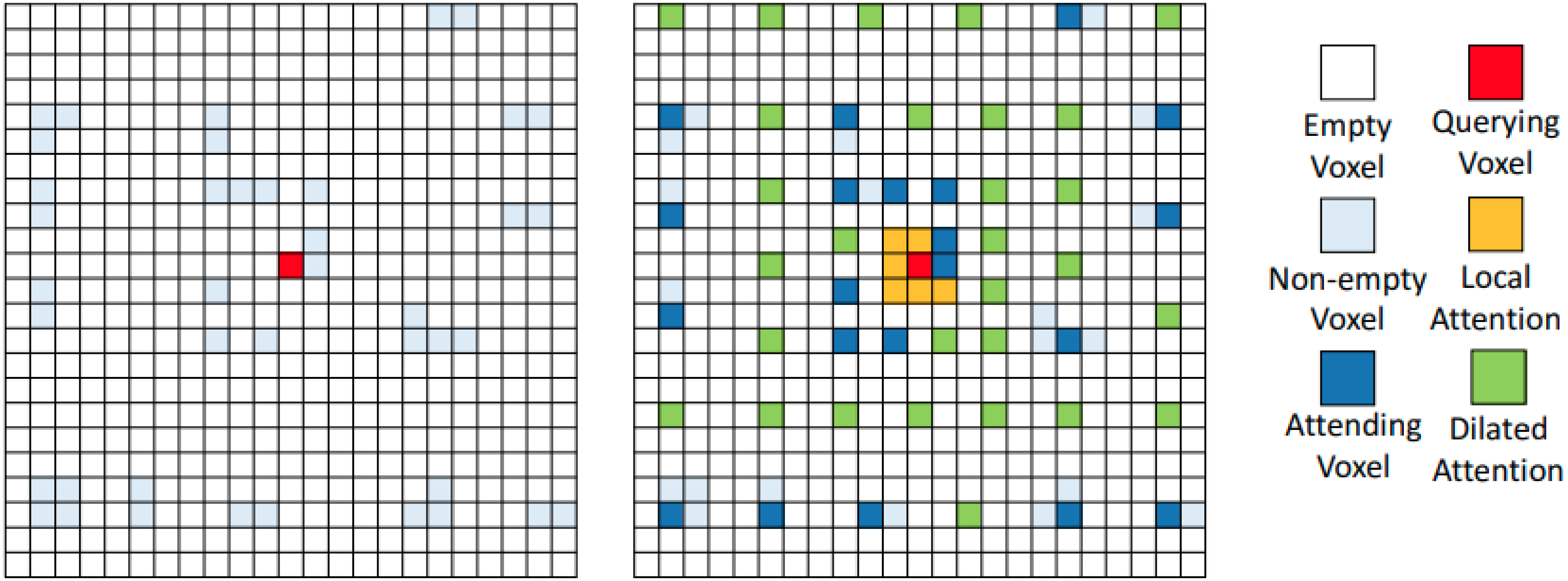


Voxel Transformer for 3D Object Detection

Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, ICCV, 2021

VoTr: Voxel Transformer





using efficient Sparse Operations for Querying, Retrieval, Convolution, Multiplication, etc.

Voxel Transformer for 3D Object Detection
Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, ICCV, 2021

Summary

- Brief Review: ML, DL, Deep Nets, CNNs, Transformers
- 3D Deep Learning
- 3D Voxel CNNs
- Hierarchical and Sparse 3D Voxel CNNs
- 3D Voxel Transformers

That’s All