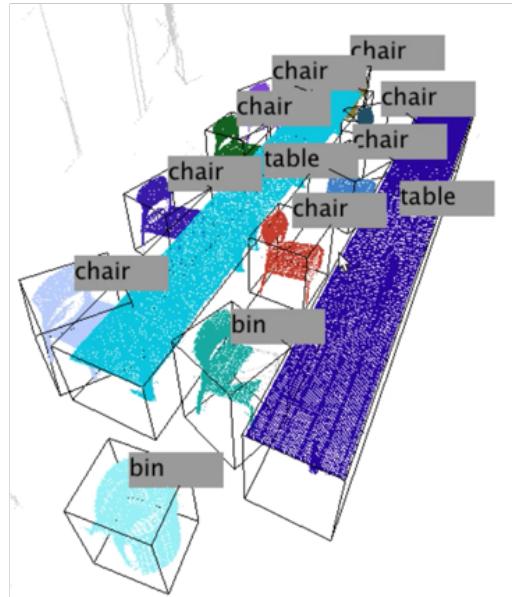
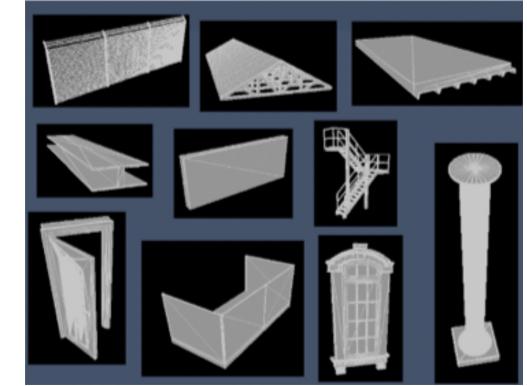


Deep Learning Tools for Understanding and Modeling the Built Environment



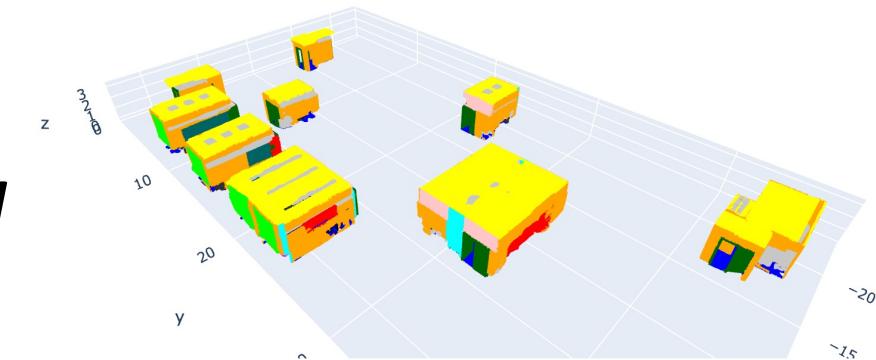
Jingdao Chen, Yong Cho

Seongyong Kim, Jisoo Park



July 28, 2024

Pittsburgh, Pennsylvania



Workshop Materials



https://github.com/jingdao/i3ce2024_DL_Workshop

A screenshot of a GitHub repository page for 'jingdao/i3ce2024_DL_Workshop'. The repository has 2 branches and 0 tags. The main content area shows files like data, LICENSE, README.md, etc. On the right, there's a 'Clone' section with options for HTTPS, SSH, and GitHub CLI. Below that is a link to clone using the web URL. A red circle highlights the 'Download ZIP' button, which is located under the 'Clone' section. The 'Download ZIP' button is accompanied by a small icon of a folder with a file. The rest of the repository details are visible, including commit history for files like pointnet.py, pointnet_classification.ipynb, and pointnet_segmentation.ipynb.

<https://colab.google/>

Outline

Background

Deep Learning

Point Cloud Classification

Point Cloud Segmentation

Outline

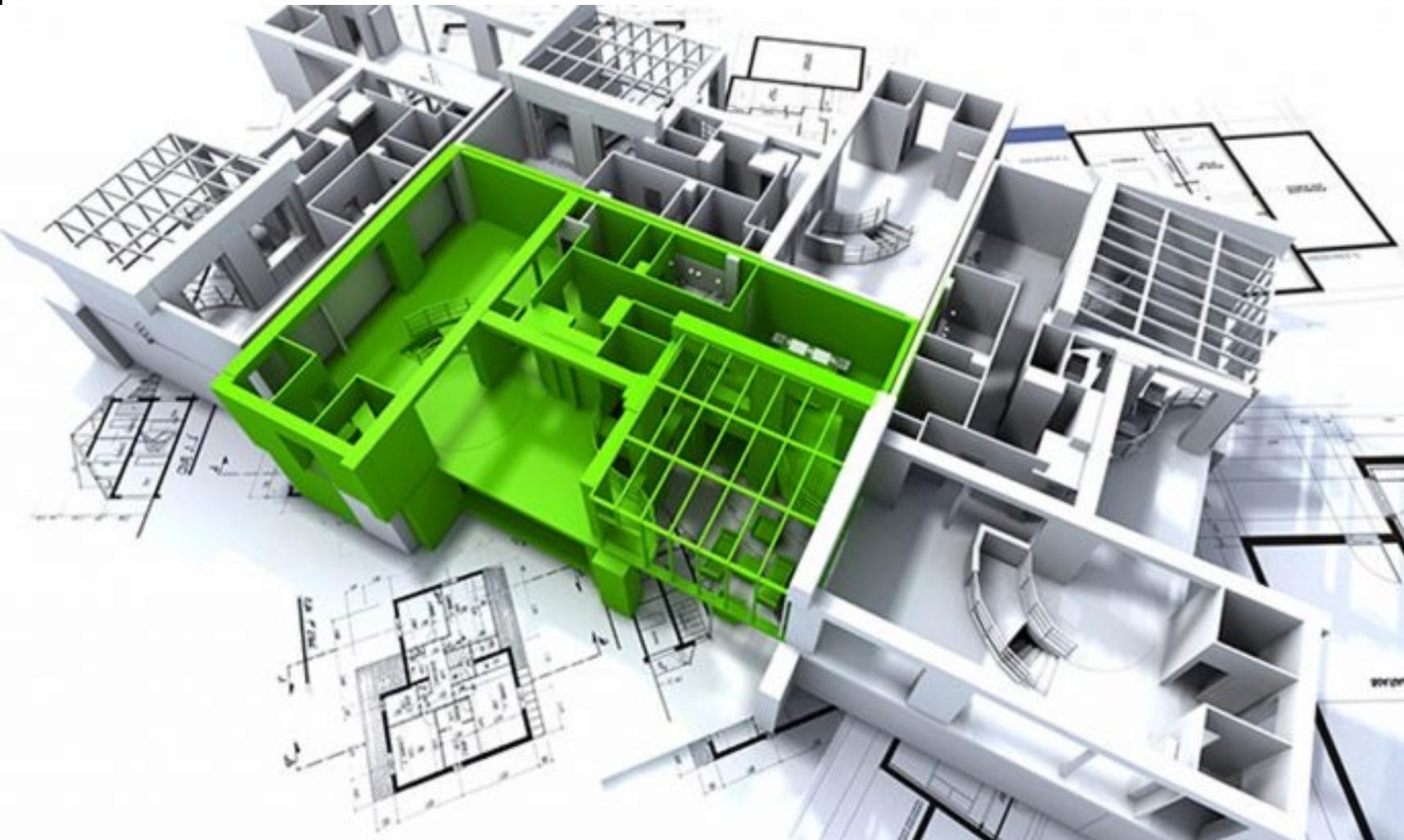
Background

Deep Learning

Point Cloud Classification

Point Cloud Segmentation

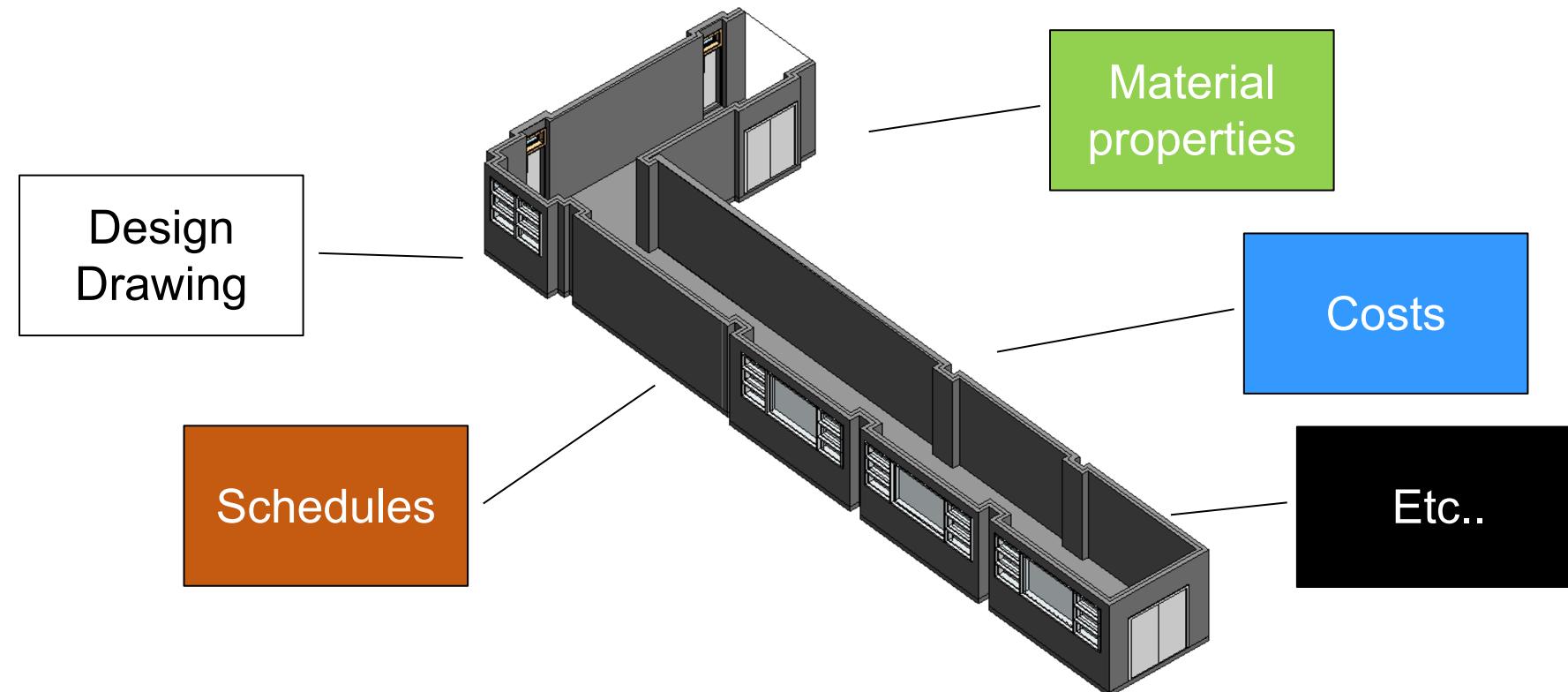
Modeling the Built Environment



<https://www.jlarchs.com/bim-and-revit-explained-the-digital-building-industry/>

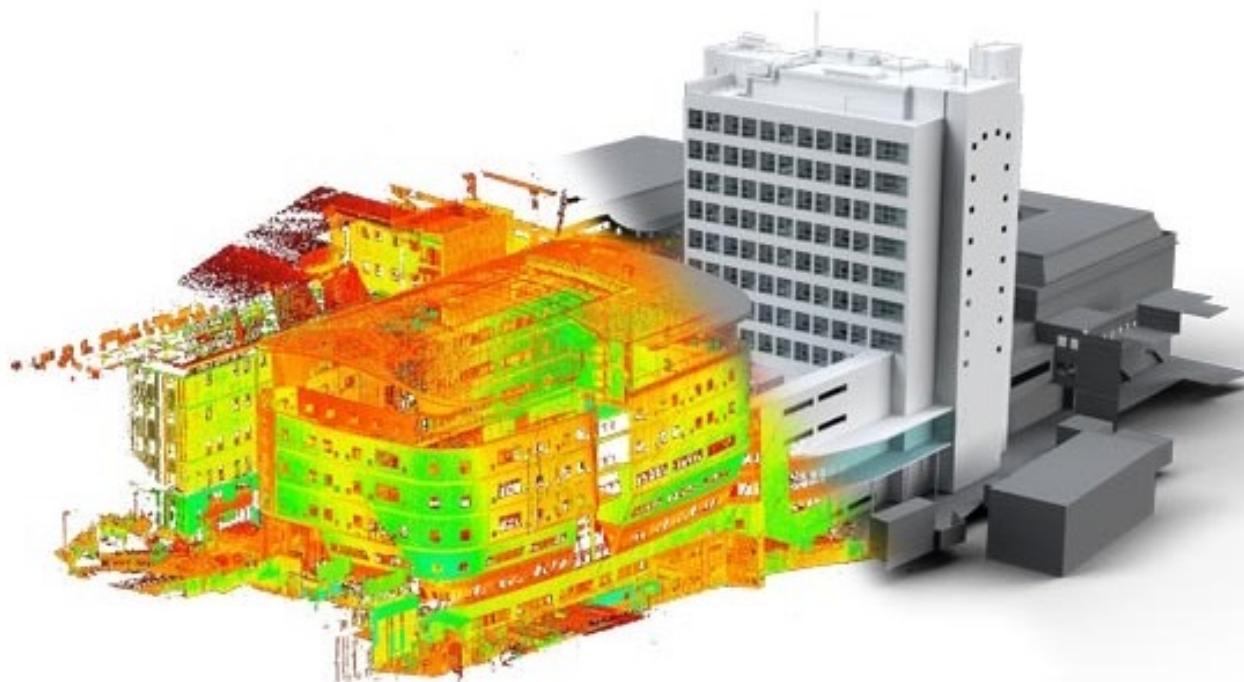
Building Information Modeling

- Building information modeling (BIM) is a process for creating digital representations of facilities that records all information relevant to a building's life cycle from construction to demolition



Building Information Modeling

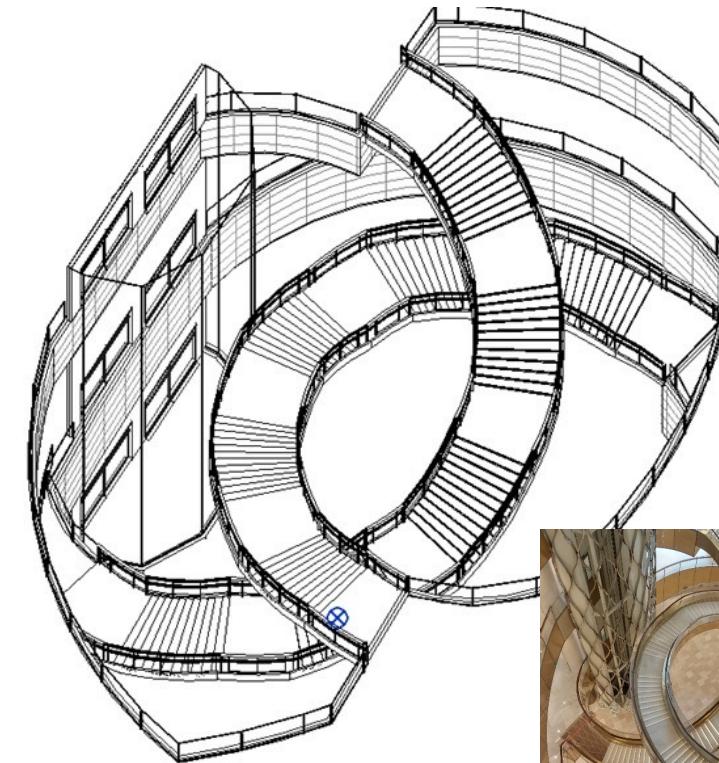
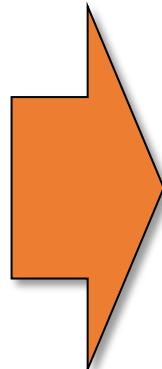
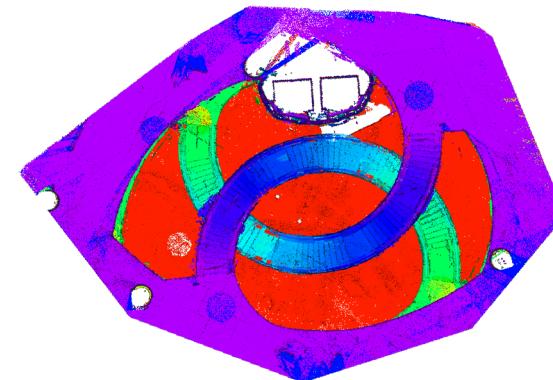
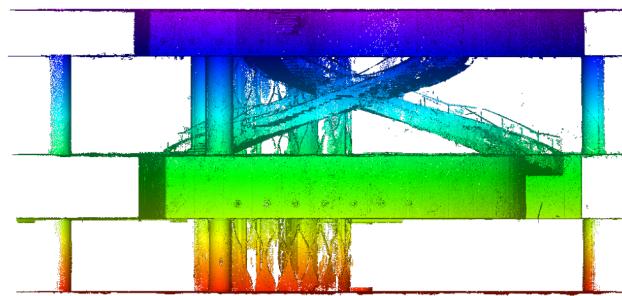
- BIM created in the design stage of a facility is called **as-designed BIM**, and the BIM that reflects a facility in its as-built condition is called **as-built BIM**



- Detection of defects
- Timely quality control
- Support for decision making
- Maintenance of buildings

Building Information Modeling

- Typically in BIM tools, a 3D laser-scanned point cloud is used as a guide enabling modelers to effectively identify and trace the object's shape

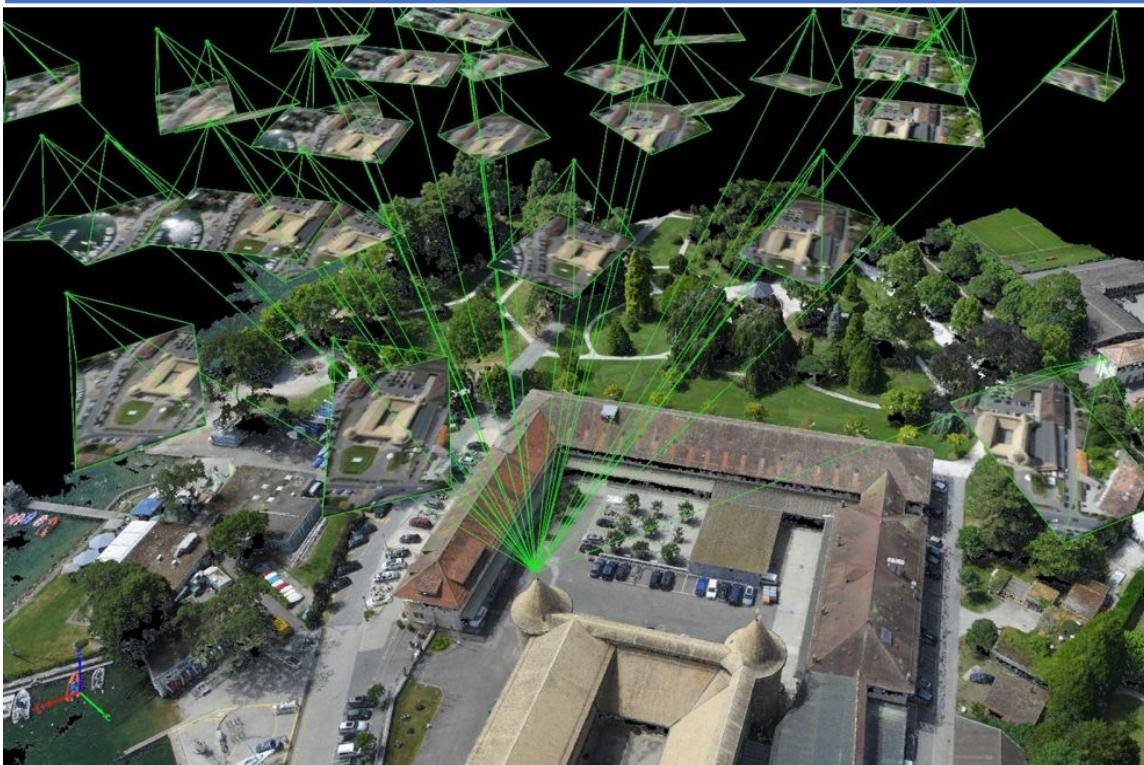


3D Point Clouds

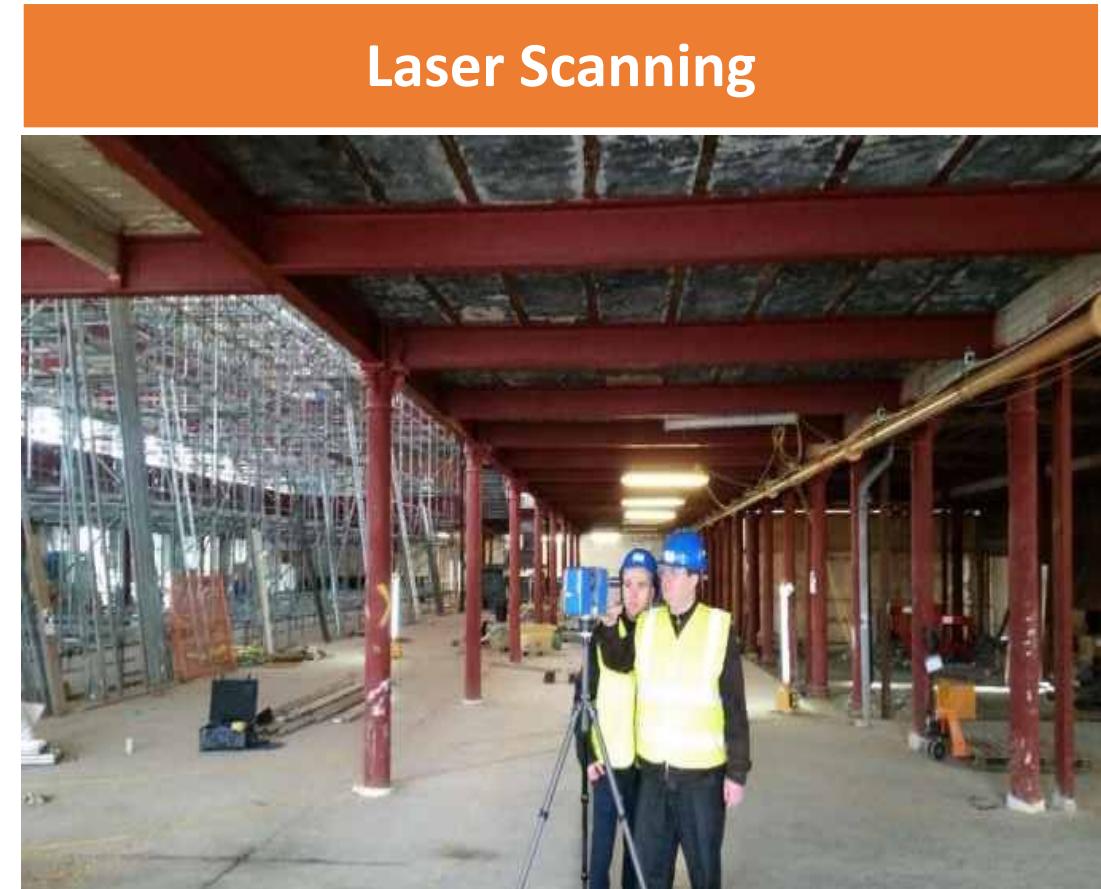
How to sense and represent the physical environment?

3D Point Clouds

Camera + Photogrammetry



Laser Scanning



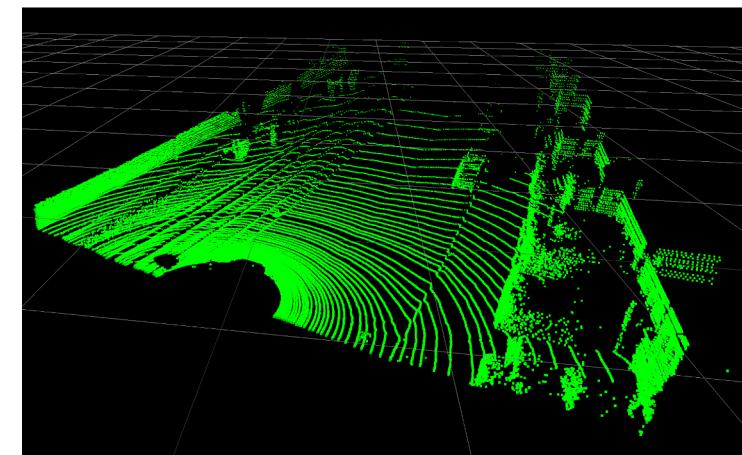
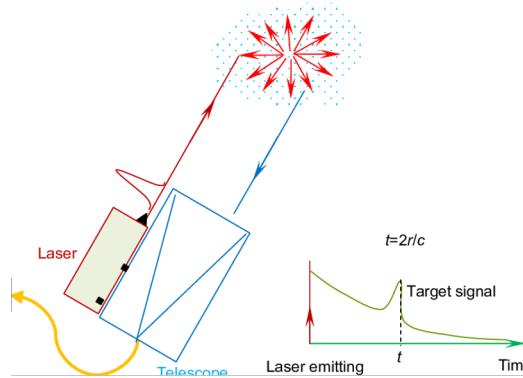
3D Point Clouds

- **Structure from Motion (SfM)** – 3D reconstruction from 2D images
- Matching pixels from multi-view images can be triangulated into points in 3D space



3D Point Clouds

- Measures distances to objects using laser beams
- Often these lasers are mounted on a rotating platform. By knowing the angle of the motor, we can calculate the 3D point where that laser hit. These sensors can capture hundreds of thousands of points a second.



3D Point Clouds



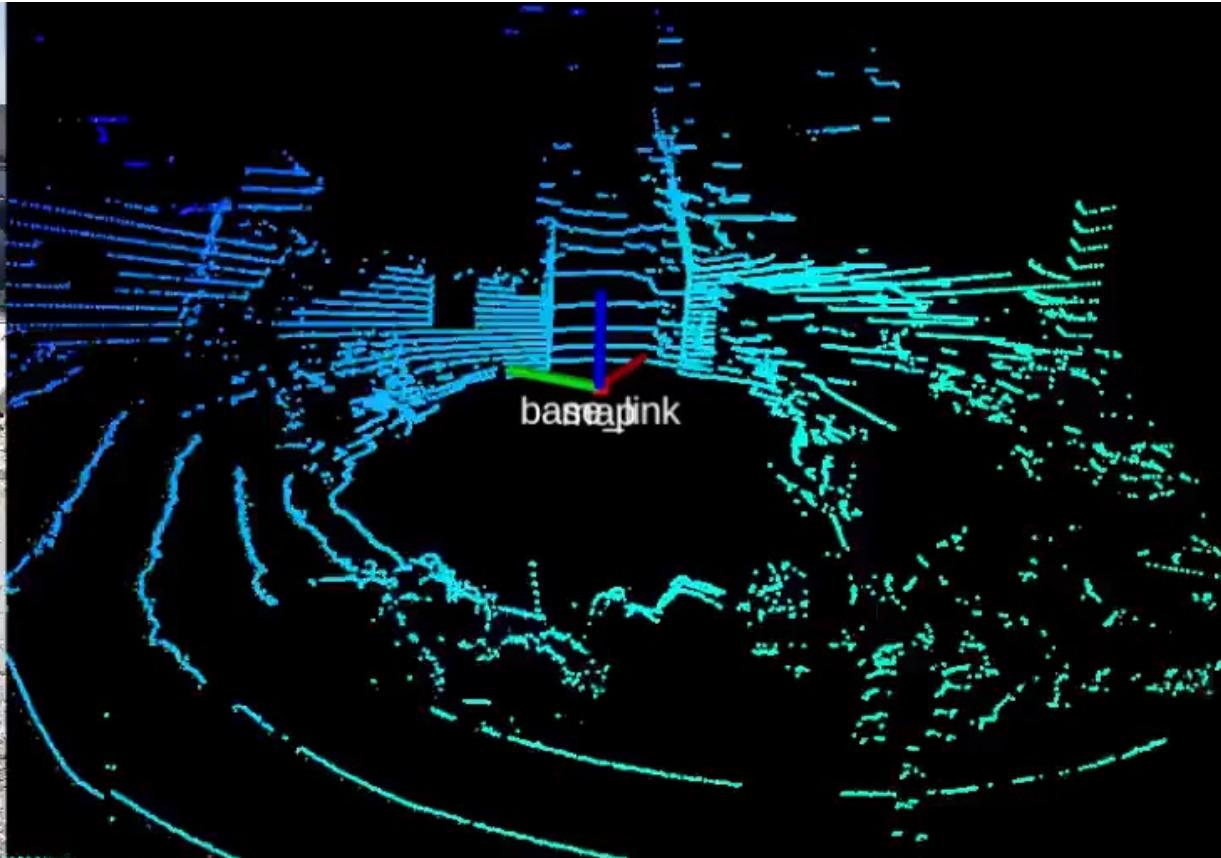
	RIEGL VZ-400i	TOPCON GLS-2000M	Leica ScanStation P30	FARO Focus ^S 150	Z+F IMAGER 5016
Type	ToF	ToF	ToF	PS	PS
Range	1.5–800 m	1–350	0.4–270 m	0.6–150 m	0.3–365 m
Accuracy	5 mm	3.5 mm Distance, 6" Angle	6 mm	3.5 mm	2 mm
Precision	3 mm		2 mm	1 mm	1 mm
Weight	9.7 kg	11 kg	12.25 * kg	4.2 kg	7.8 kg

Note: *w/o batteries; ToF: Time-of-Flight; PS: phase shift

3D Point Clouds

Robotic Scanning

LiDAR-scanned point clouds



Building Information Modeling

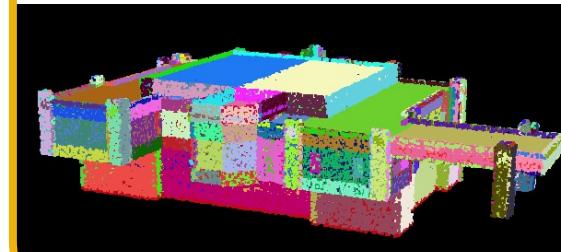
Raw laser scans

- Unstructured
- Cluttered
- Difficult to visualize



Processed point clouds

- Structured
- Interpretable



Building Information Model

- Semantically-rich
- Organized
- Portable



Data Organization

How to organize low-level data into high-level information?



- **Letters**

Data Organization

How to organize low-level data into high-level information?

*We shall not cease from exploration
And the end of all our exploring
Will be to arrive where we started
And know the place for the first time*

— T. S. ELIOT

- ✓ Words
- ✓ Sentences
- ✓ Meaning

Data Organization

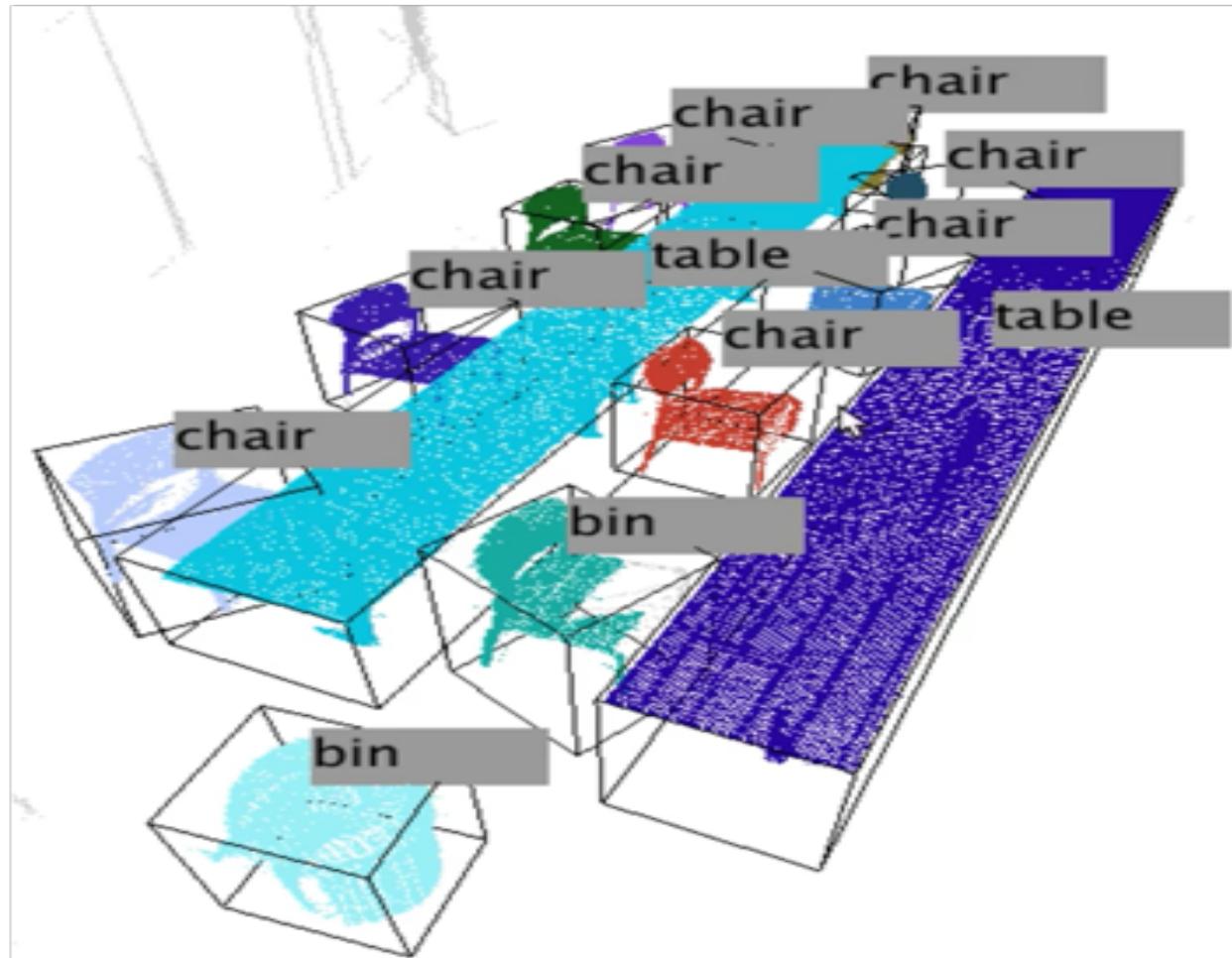
How to organize low-level data into high-level information?



- Points

Data Organization

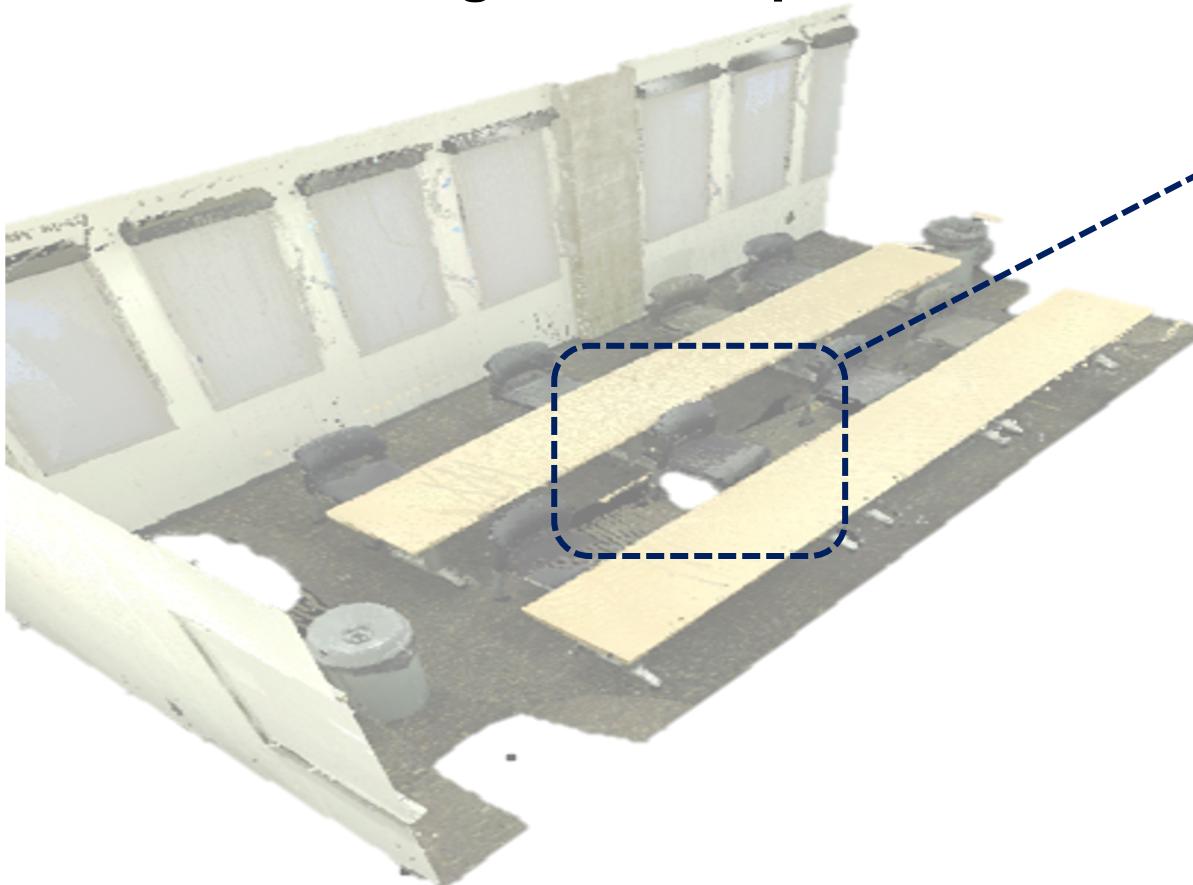
How to organize low-level data into high-level information?



- ✓ Surfaces
- ✓ Objects
- ✓ Structure

Machine Learning

Use machine learning + data to predict structure and semantics



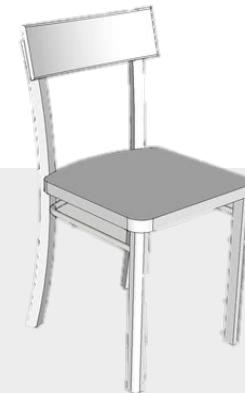
Object #10

Features

- Color: black
- Dimensions: 0.42 x 0.45 x 0.82
- Number of points: 12961

Inference

- Class: chair
- Material: plastic
- Parts: seat, legs



Outline

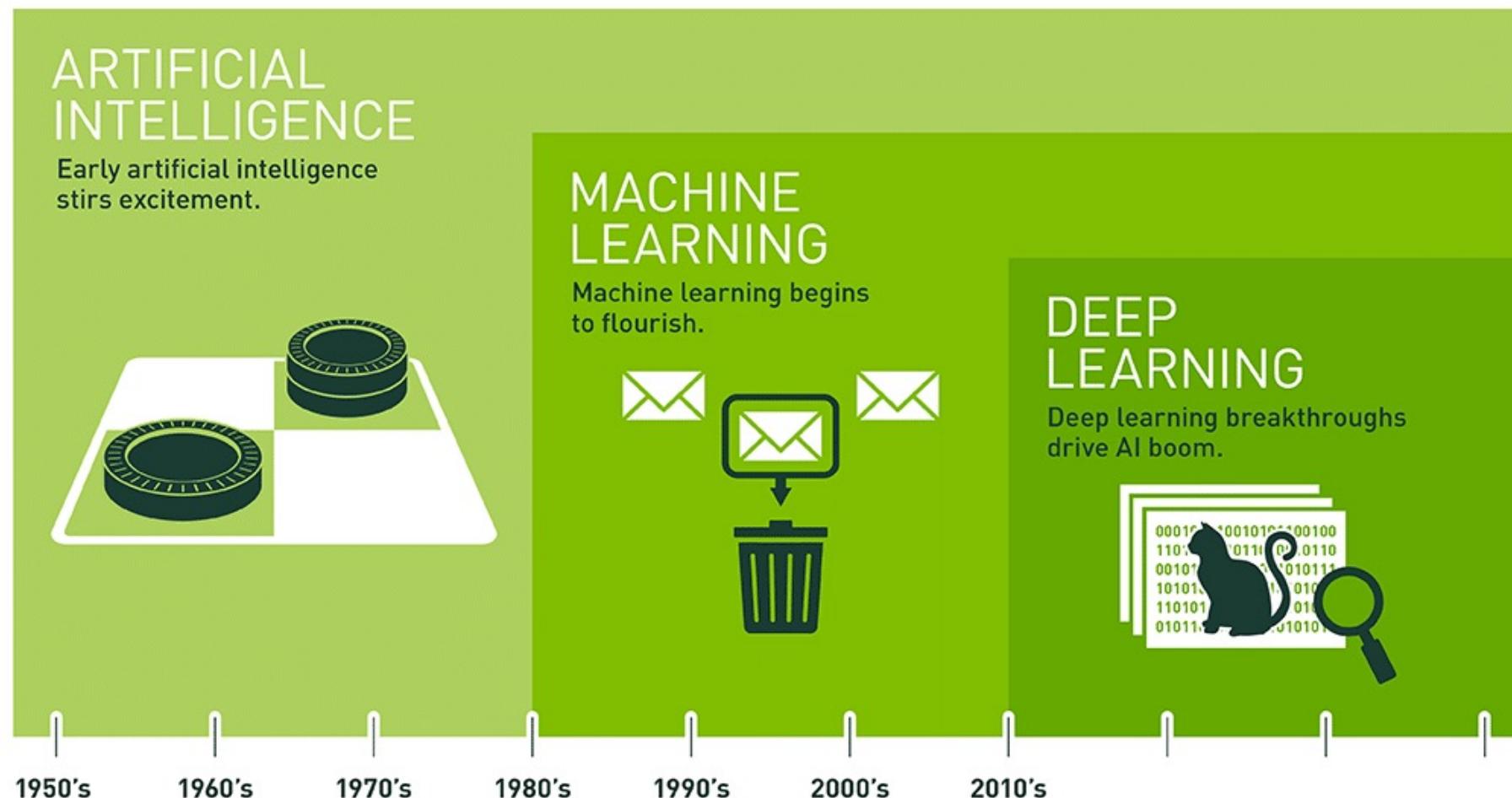
Background

Deep Learning

Point Cloud Classification

Point Cloud Segmentation

Deep Learning



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Deep Learning

The world's most valuable resource is no longer oil, but data

The data economy demands a new approach to antitrust rules



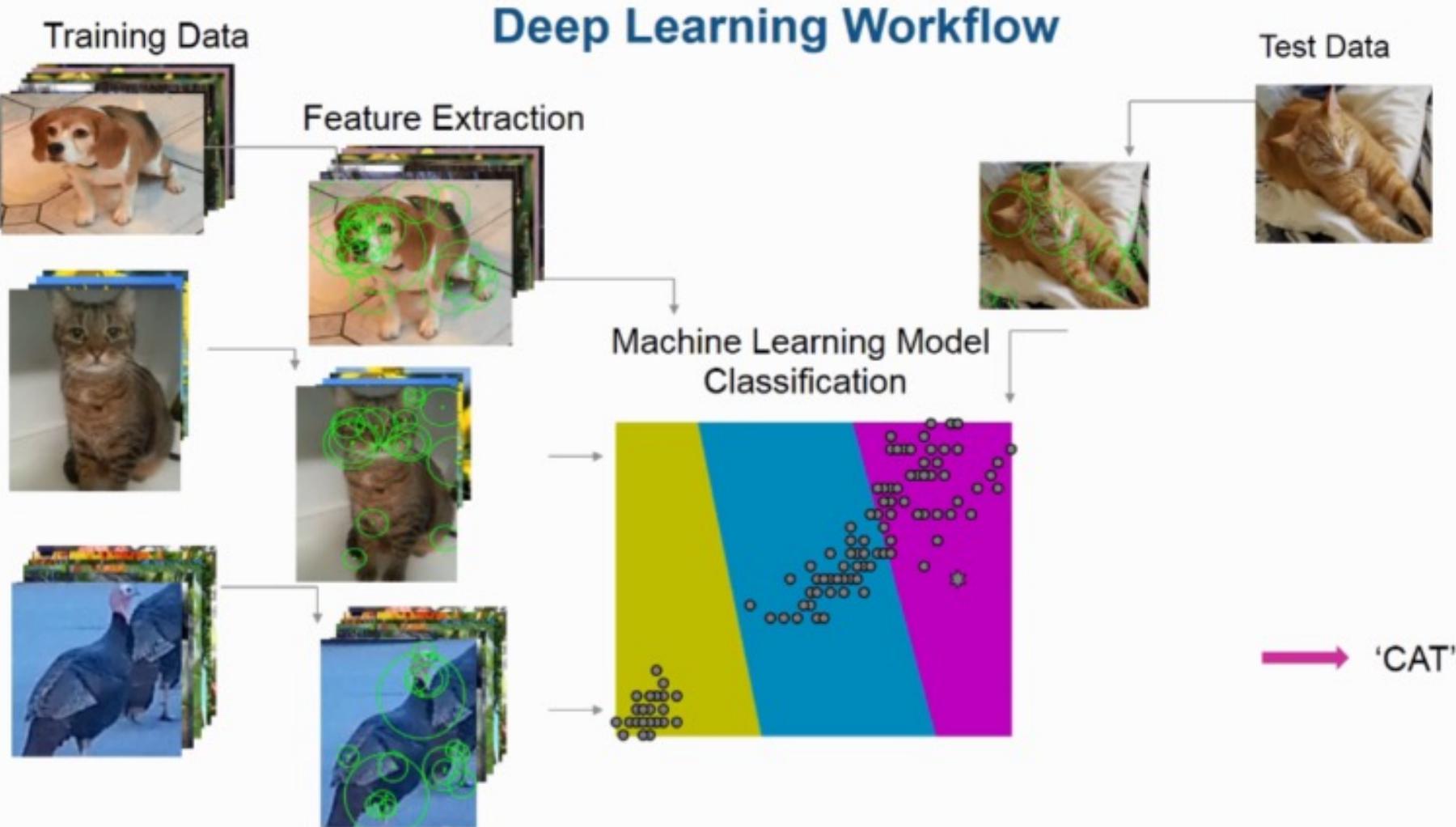
- Information revolution: large amount of data available
- Computing revolution: Parallel computing/cloud computing/GPU computing

Deep Learning

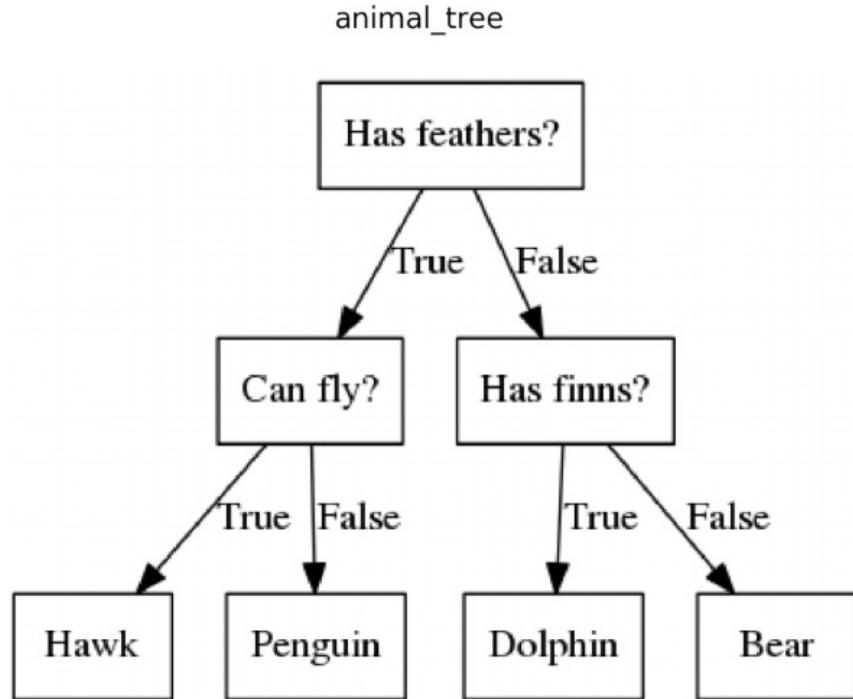
Advantages of Deep Learning

- Robustness
 - Can adapt to variations / uncertainty by directly learning them from data
- Flexibility
 - The same technique (e.g. convolutional neural networks) can be applied to many different tasks
- Scalability
 - Deep learning models have the capacity to handle >1000 object classes

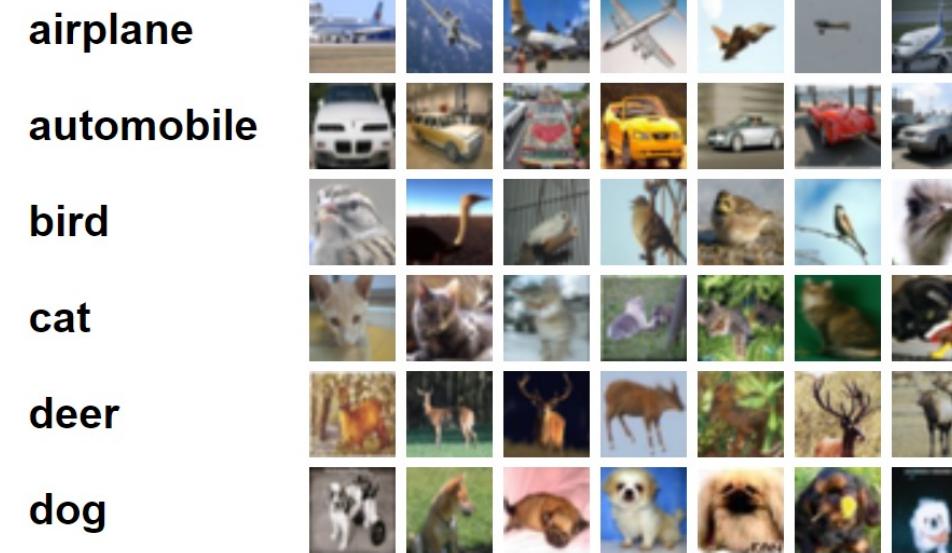
Object Recognition



Machine Learning

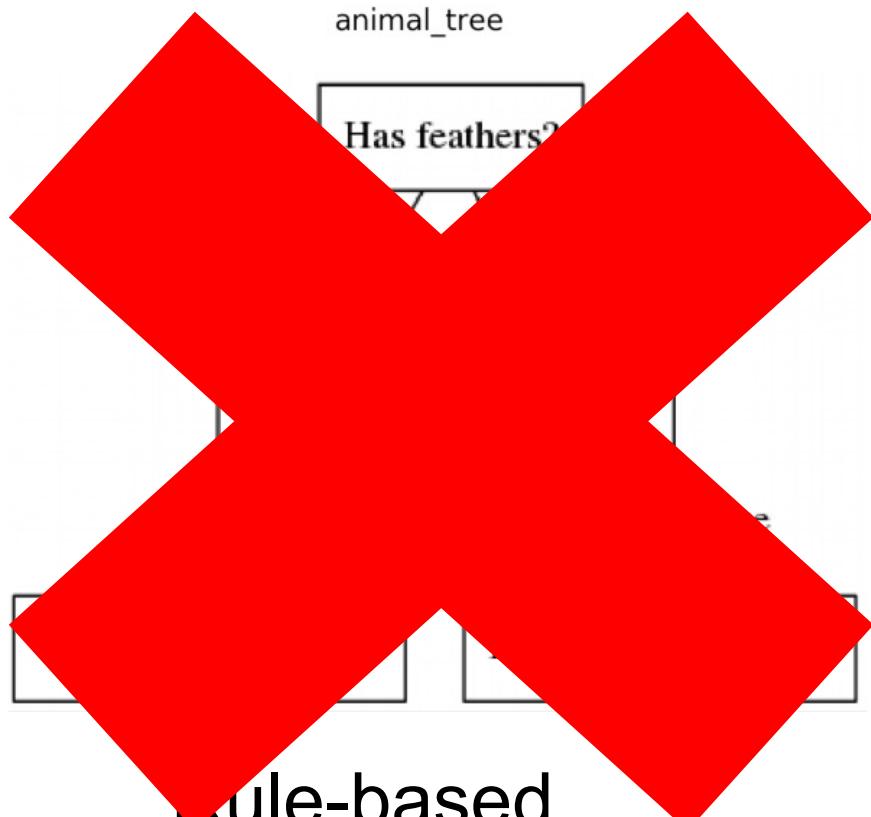


Rule-based
object recognition

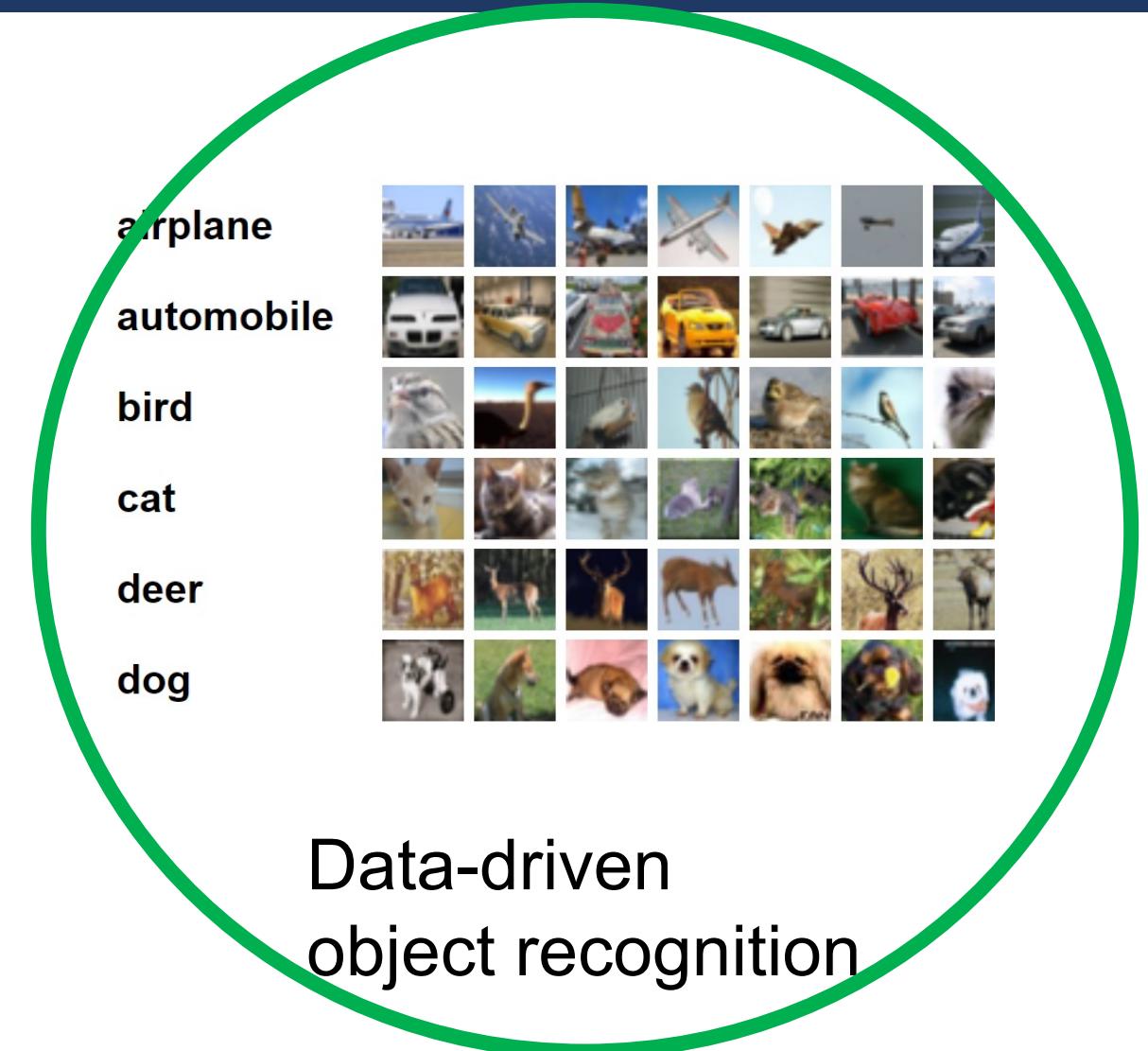


Data-driven
object recognition

Machine Learning

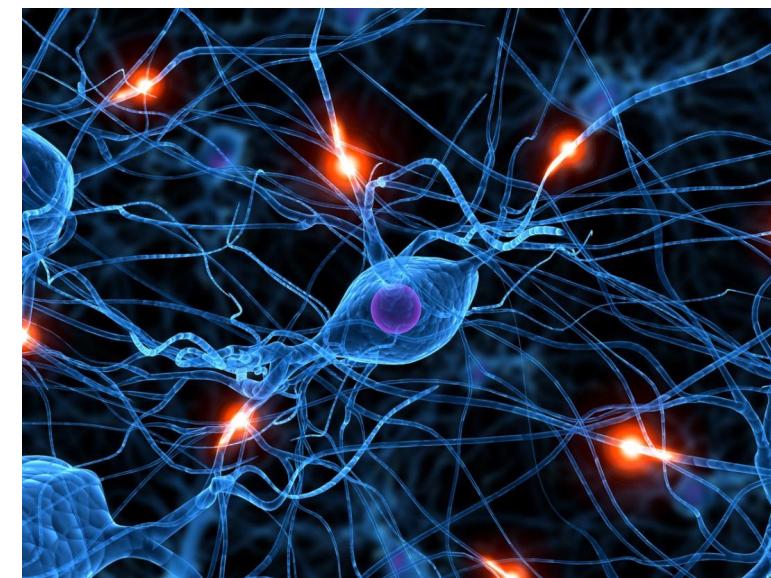
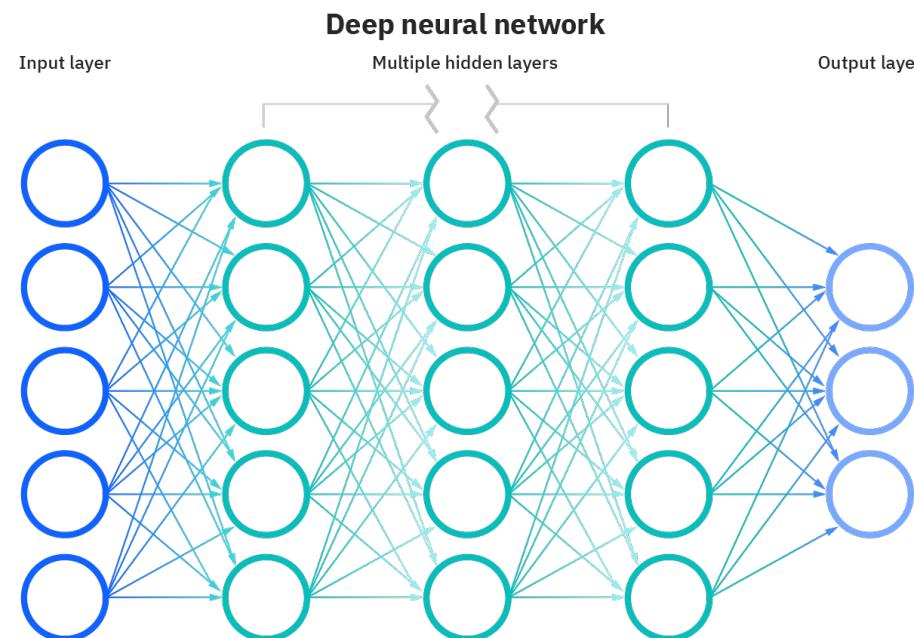


rule-based
object recognition



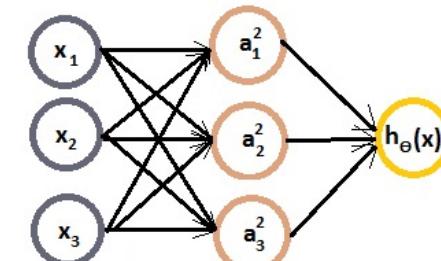
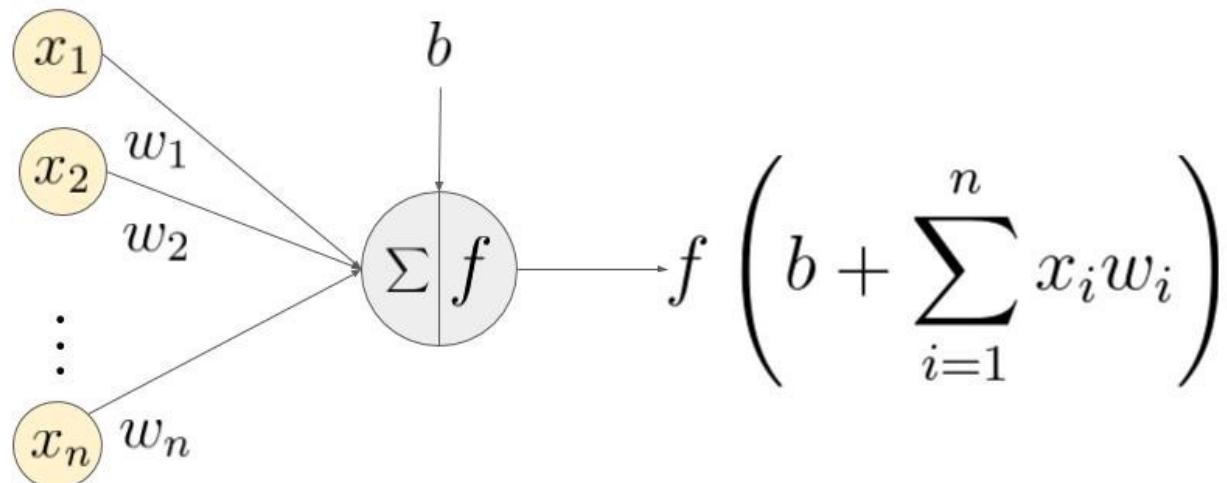
Deep Learning

- **Artificial neural network / deep neural network** is an algorithm, modeled loosely after the human brain, that is designed to perform machine learning / pattern recognition tasks
- **Neural networks** are constructed by stacking layers of different types of neurons together
- Also known as **deep learning**



Deep Learning

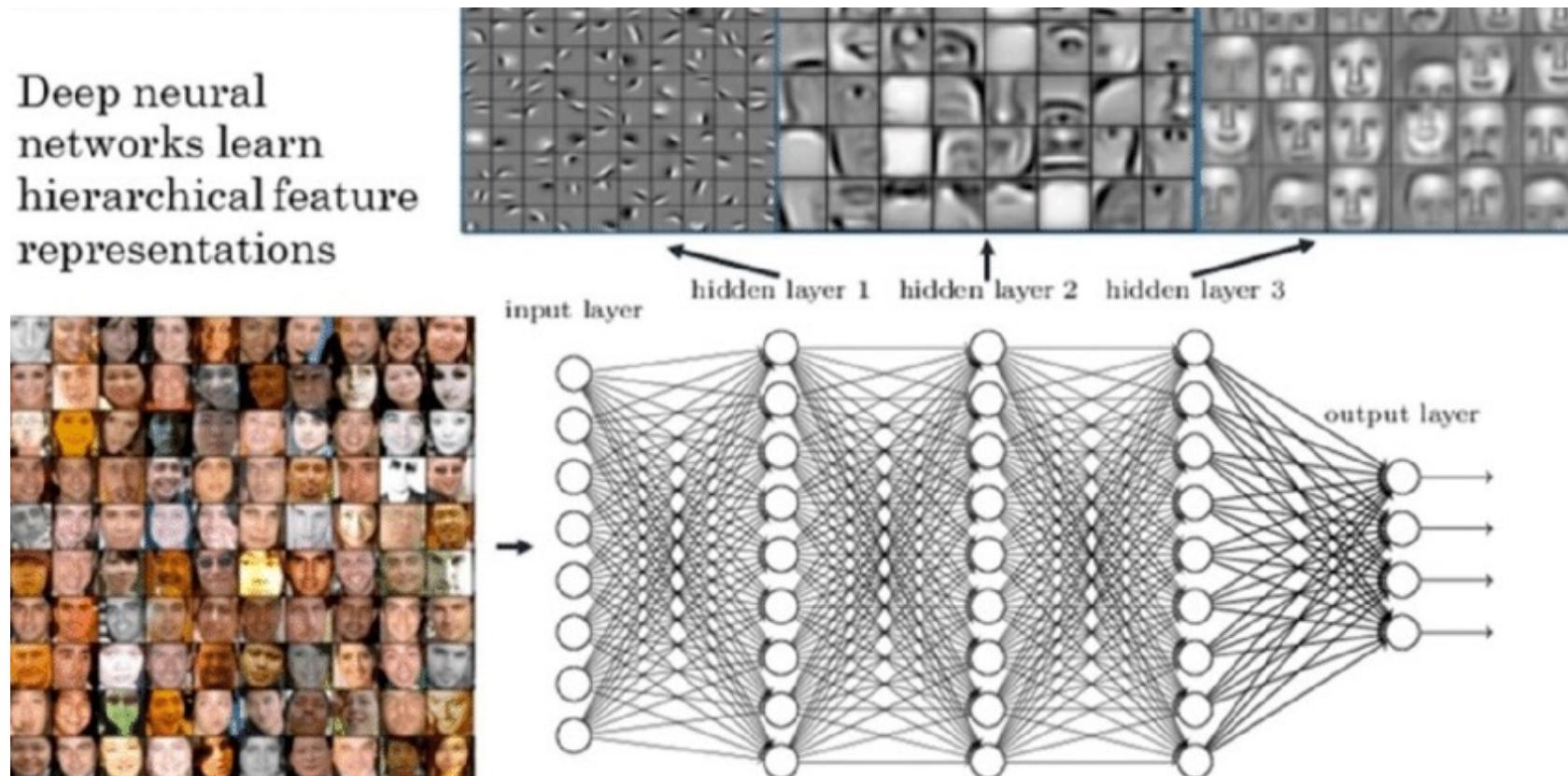
- Each unit of linear weights, bias and non-linear activation function is represented as a **neuron**
- Neurons can be stacked in complex ways to learn an arbitrary relationship between the input and output
- These properties allow a neural network to behave like a **universal function approximator**



$$\begin{aligned} a_1^{(2)} &= g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3) \\ a_2^{(2)} &= g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3) \\ a_3^{(2)} &= g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3) \\ h_\Theta(x) &= a_1^{(3)} = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)}) \end{aligned}$$

Deep Learning

- Able to handle large number of input features (i.e. high-dimensional input)
- Able to predict a large number of outputs (i.e. high-dimensional output)
- Able to learn intermediate low-level and high-level features
- Able to process input samples directly without hand-engineered features



Mathematical Perspective

- $y = f(x; \theta)$
- x : input y : output θ : model parameters / weights
- Objective function, $J(\theta)$ is a performance measure for the task, i.e. how closely does our predicted output match the ground truth output
- Aim to minimize objective function

$$J(\theta) = \frac{1}{N} \sum_{i=1}^N [f(x_i; \theta) - y_i]^2$$

N: number of data samples

Classification: y is an integer label

Regression: y is a floating-point value

Gradient Descent

$$\theta \leftarrow \theta - \alpha \frac{\partial J}{\partial \theta}$$

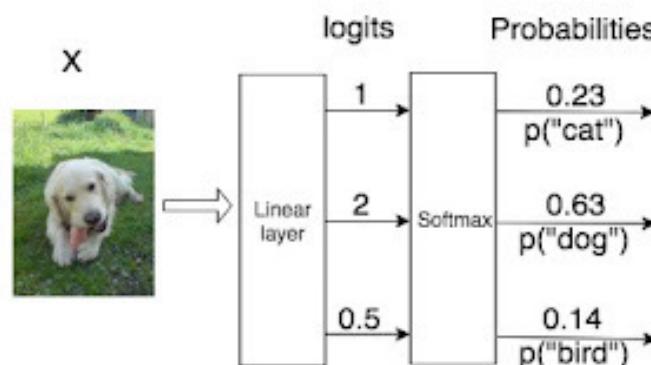
- θ : parameters to be estimated
- J : cost function to be optimized
- $\partial J / \partial \theta$: gradient of J with respect to θ
- α : step size, or learning rate
- Iterative algorithm: our estimate of θ improves slightly with each iteration

Classification

Cross-entropy loss

C: number of classes

$$J(\theta) = \frac{1}{N} \sum_{i=1}^N \sum_c -y_{c,i} \log f(x_i, \theta_c)$$

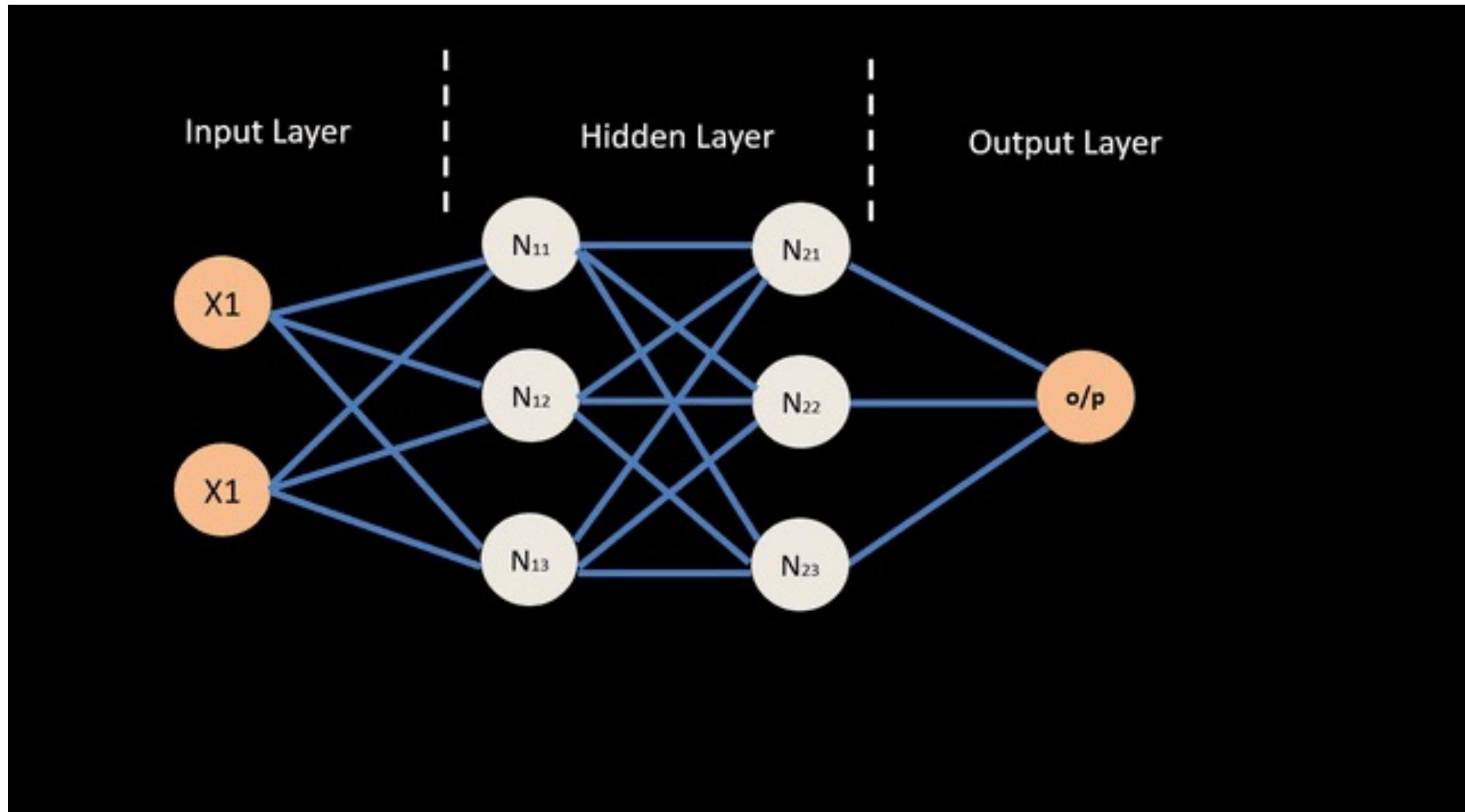


Ground truth
label

$$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\begin{aligned} J(\theta) &= -[(0)\log(0.23) \\ &\quad + (1)\log(0.63) + (0)\log(0.14)] \\ &= 0.46 \end{aligned}$$

Backpropagation



Outline

Background

Deep Learning

Point Cloud Classification

Point Cloud Segmentation

3D Point Clouds

Hundreds of Point cloud file formats: PLY, PTS, PCD, ...

1 # .PCD v.5 - Point Cloud Data file format
2 VERSION .5
3 FIELDS x y z
4 SIZE 4 4 4
5 TYPE F F F
6 COUNT 1 1 1
7 WIDTH 207001
8 HEIGHT1
9 POINTS 207001
10 DATA ascii
11 58.72083282 -72.24057770 301.85263062 55 55 55
12 59.08960724 -72.32469177 302.19873047 60 60 60
13 59.45919037 -72.40894318 302.54537964 62 62 62
14 59.80107117 -72.45877838 302.74816895 65 65 65
15 60.12423325 -72.48554993 302.85455322 64 64 64
16 60.45264053 -72.51833344 302.98605347 66 66 66
17 60.79652786 -72.56927490 303.19338989 67 67 67
18 61.14083862 -72.62021637 303.40066528 66 66 66
19 61.47315598 -72.65647125 303.54653931 64 64 64
20 61.79760742 -72.68312836 303.65228271 63 63 63
21 62.13186646 -72.72099304 303.80487061 61 61 61
22 62.49474716 -72.79183197 304.09512329 59 59 59
23 62.85828018 -72.86273956 304.38565063 60 60 60
24 63.21944809 -72.93024445 304.66189575 60 60 60

PCD

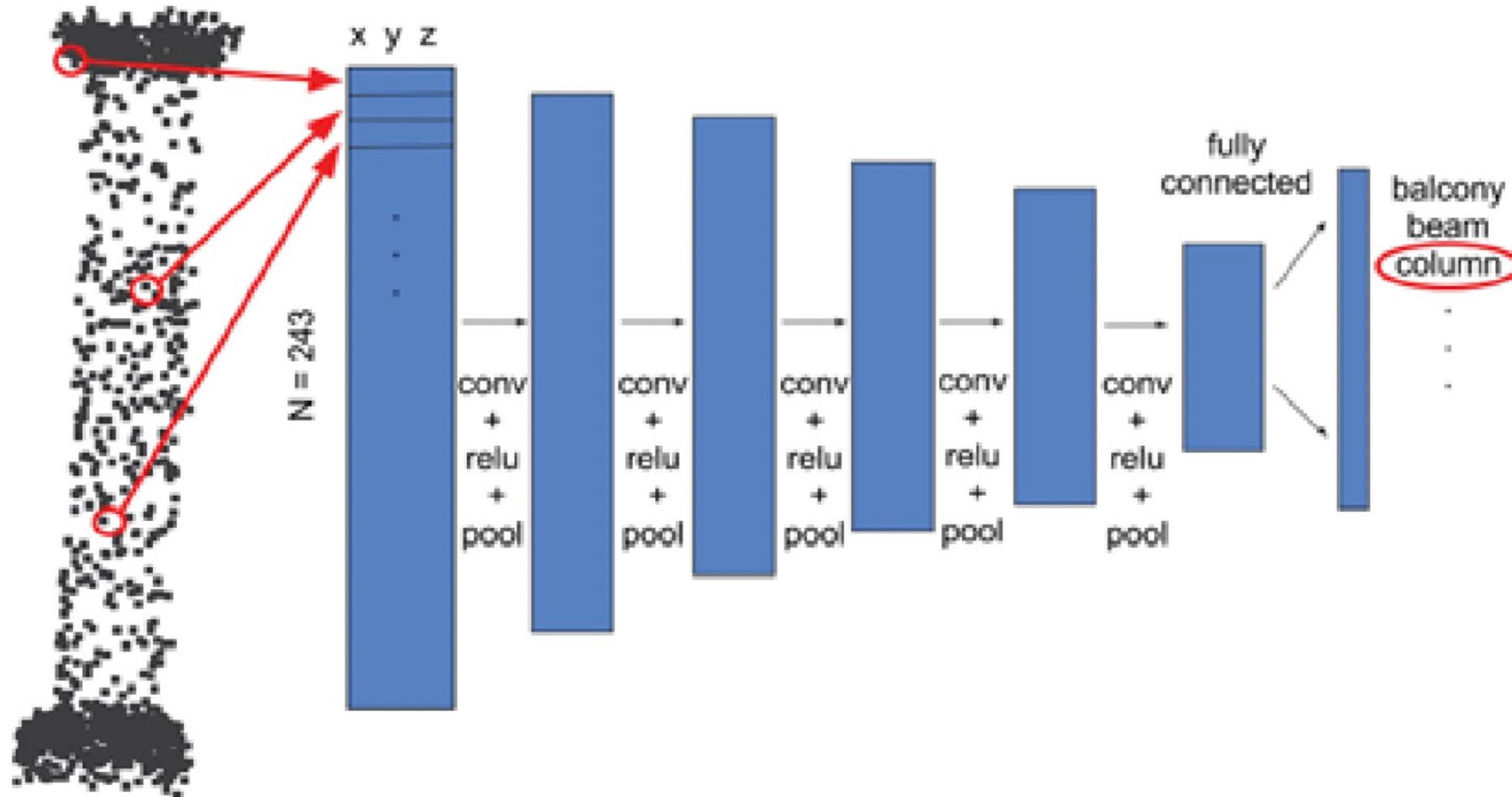
Header lines

Point attributes (x, y, z, R, G, B, (optional) intensity)

Header lines

- File format info.
- Dimensions
- Type of data
- Number of points

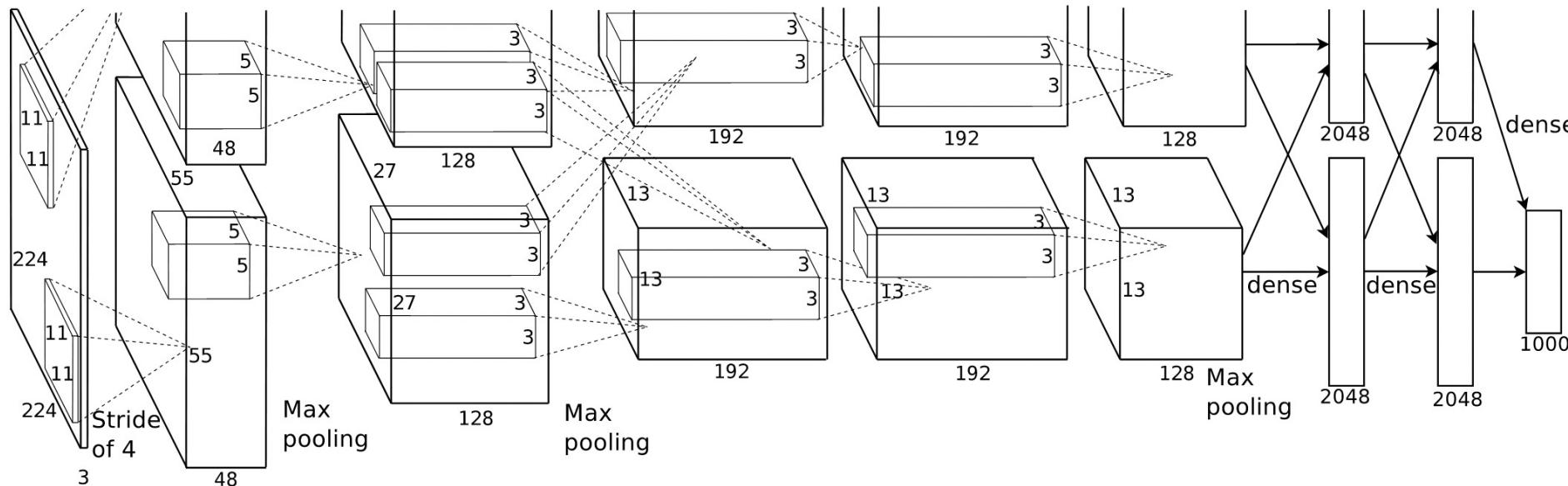
Building Element Classification



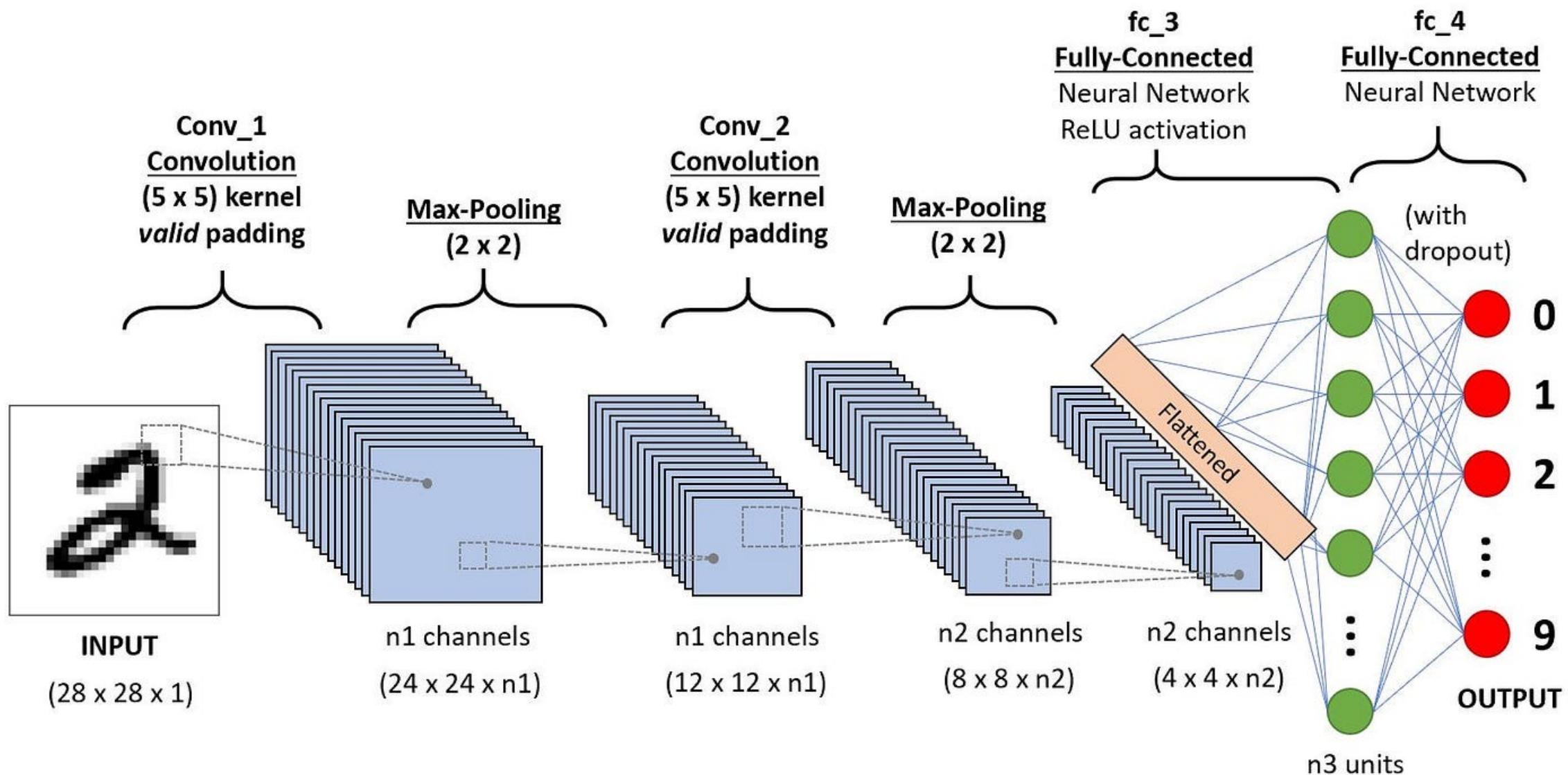
Neural Network Architectures

AlexNet (2012)

- Winner of the 2012 ImageNet challenge, with a top-5 error of more than 10.8% lower than the runner up
- With 60M parameters, AlexNet has 8 layers — 5 convolutional and 3 fully-connected
- Makes use of max pooling layers and ReLU activation function



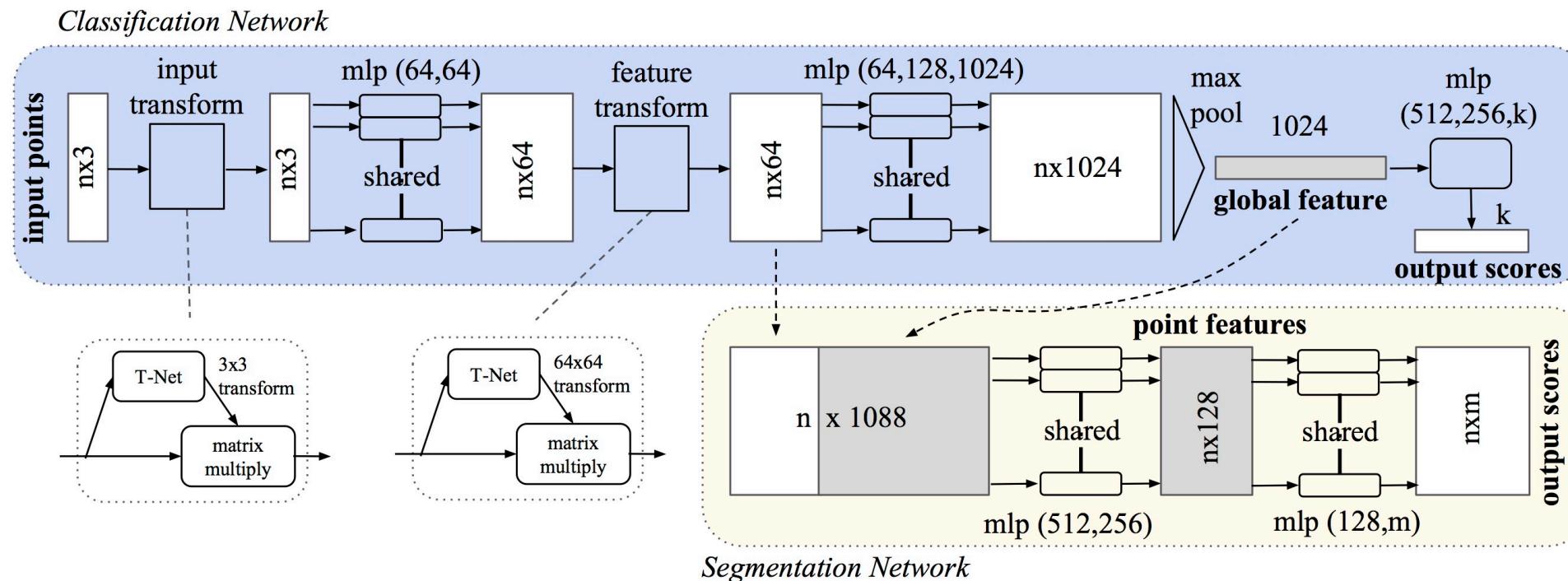
Neural Network Architectures



Neural Network Architectures

PointNet (2017)

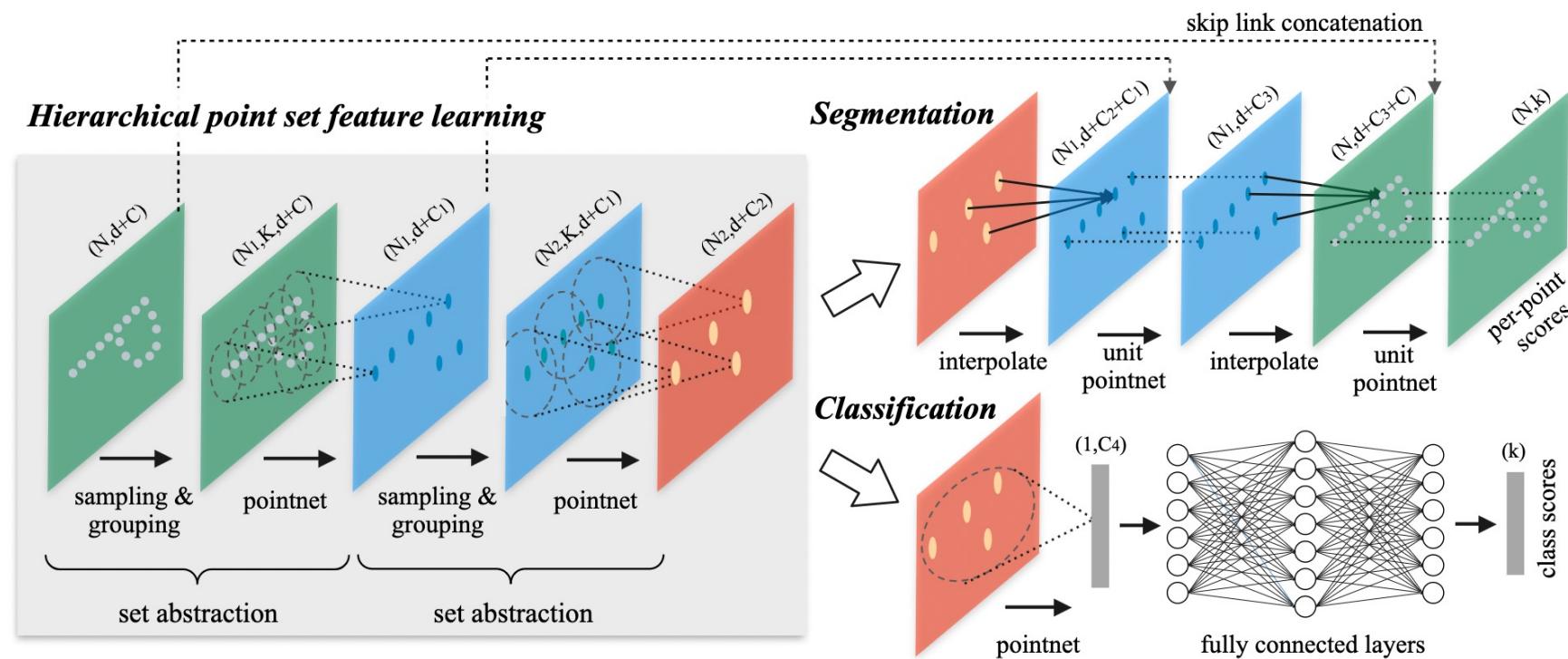
- One of the first architectures that can work directly with unordered point clouds
- Consists of point-wise convolution layers and a global pooling layer to combine features over the entire point cloud



Neural Network Architectures

PointNet++ (2017)

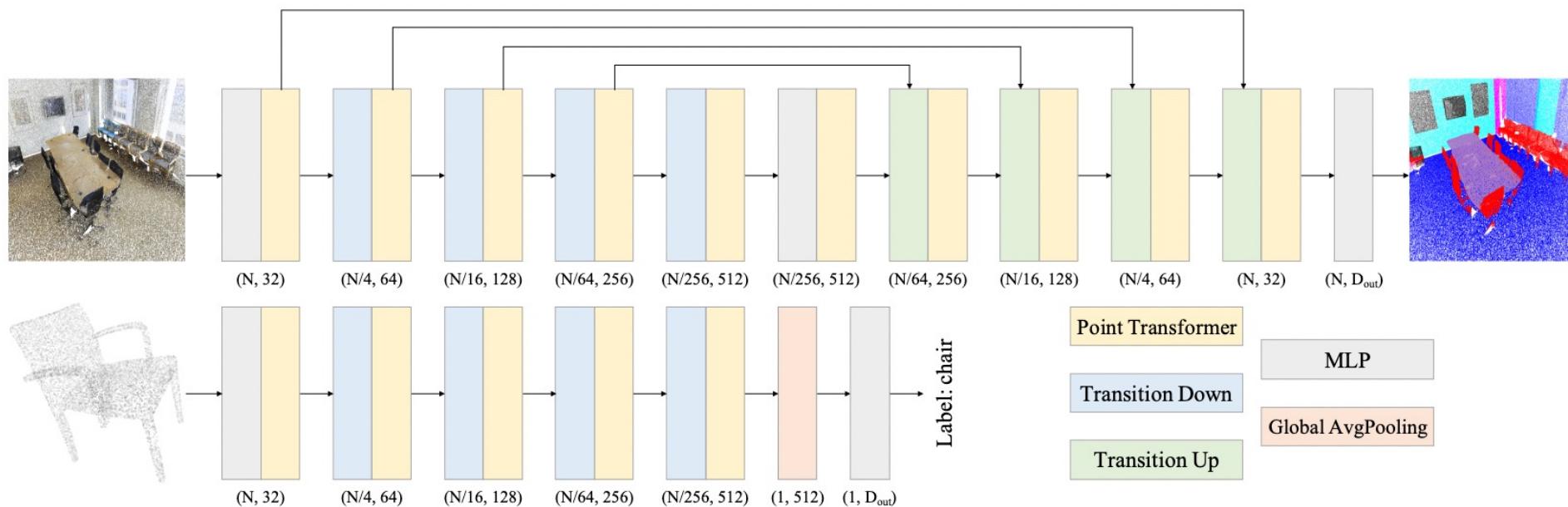
- Learn local features with increasing contextual scales
- Applies PointNet recursively on a nested partitioning of the input point set



Neural Network Architectures

Point Transformer (2021)

- Uses self-attention operators to extract features from local neighborhoods around each point, and encode of positional information in the network
- Point Transformer can serve as the backbone for multiple 3D point cloud understanding tasks



Hands-on Coding



GitHub repository interface showing the 'Clone' section:

Initial commit by **jingdao**

Local Codespaces

Clone

HTTPS SSH GitHub CLI

https://github.com/jingdao/i3ce2024_DL_Workshop

Clone using the web URL.

Open with GitHub Desktop

Download ZIP (circled in red)

File	Commit	Time
LICENSE	initial commit	1 hour ago
README.md	initial commit	1 hour ago
dataloader_modelnet.py	initial commit	1 hour ago
dataloader_s3dis.py	initial commit	1 hour ago
pointnet.py	initial commit	1 hour ago
pointnet_classification.ipynb	initial commit	1 hour ago
pointnet_segmentation.ipynb	initial commit	1 hour ago
utils.py	initial commit	1 hour ago

https://github.com/jingdao/i3ce2024_DL_Workshop

<https://colab.google/>

Hands-on Coding

Upload code and data files to Google Drive

The screenshot shows a Google Drive interface with the following details:

- Left Sidebar:** Includes links for Home, My Drive, Computers, Shared with me, Recent, Starred, Spam, Trash, and Storage. It also shows 8.81 GB of 15 GB used.
- Search Bar:** "Search in Drive".
- Breadcrumbs:** My Drive > i3ce 2024 DL Workshop > code
- File List:** A table showing the contents of the "code" folder. The columns are Name, Owner, Last modified, File size, and more options (three dots).
- Annotations:** Red arrows and circles highlight specific items:
 - A red arrow points from the text "Folder containing sample point cloud data" to the "data" folder, which is circled in red.
 - A red arrow points from the text "Click this to open the Python notebook in Google Colab" to the "pointnet_classification.ipynb" file, which is also circled in red.

Name	Owner	Last modified	File size	More
__pycache__	me	Jul 24, 2024	—	⋮
data	me	Jul 23, 2024	—	⋮
dataloader_modelnet.py	me	Jul 24, 2024	1 KB	⋮
dataloader_s3dis.py	me	Jul 24, 2024	4 KB	⋮
pointnet_classification.ipynb	me	2:32 PM	357 KB	⋮
pointnet_classification.pth	me	2:31 PM	3.1 MB	⋮
pointnet_segmentation.ipynb	me	2:20 PM	43 KB	⋮
pointnet.py	me	2:28 PM	2 KB	⋮
utils.py	me	Jul 24, 2024	2 KB	⋮

Folder containing sample point cloud data

Click this to open the Python notebook in Google Colab

Hands-on Coding

pointnet_segmentation.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Files

{x}

code

data

dataloader_m...

dataloader_s...

pointnet.py

pointnet_class...

pointnet_class...

pointnet_segm...

utils.py

i3ce 2024 Works...

icon

media

49.33 GB available

+ Code + Text

31s

T4 GPU

Comment Share ⚙ J

T4 RAM Disk Gemini

Python code

3D Point Cloud Visualization

z

10

30s completed at 4:31 PM

trace 0

trace 1

trace 2

· trace 3

· trace 4

· trace 5

· trace 6

· trace 7

· trace 8

· trace 9

```
for i in range(num_rooms_to_visualize):
    pcd_object = o3d.geometry.PointCloud()
    pcd_object.points = o3d.utility.Vector3dVector(train_dataset.points[i][:, :3])
    # assign colors based on the ground truth class label for each point
    pcd_object.colors = o3d.utility.Vector3dVector(np.array(class_colors)[train_dataset.labels[i]])
    visualized_objects.append(pcd_object)

# Visualize the point cloud using a 3D web viewer
draw_geometries(visualized_objects, show_axes=True)
```

Outline

Background

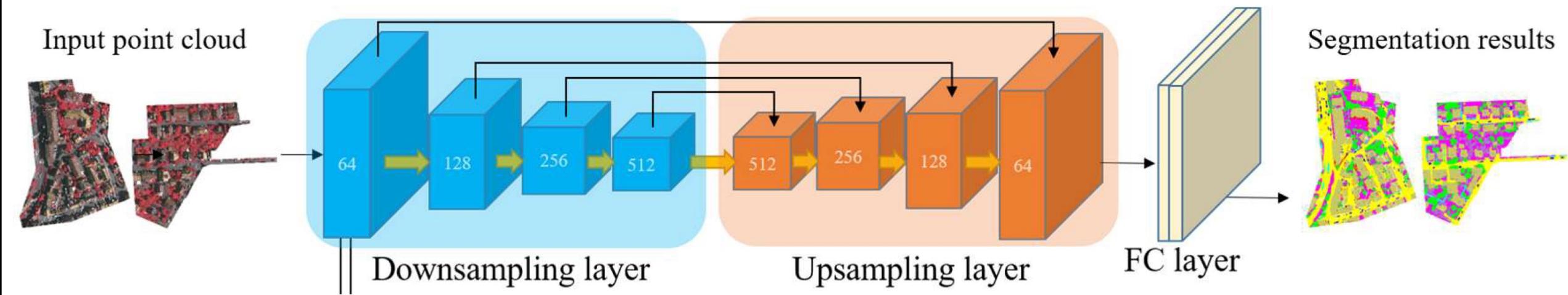
Deep Learning

Point Cloud Classification

Point Cloud Segmentation

Neural Network Architectures

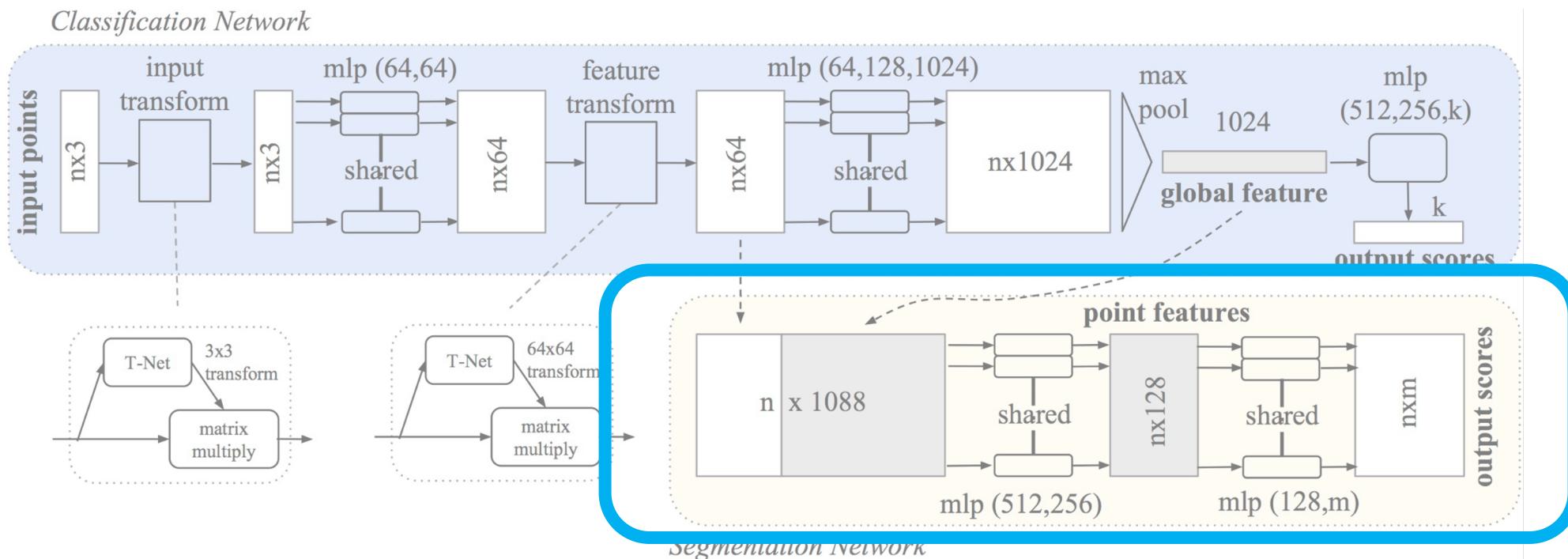
- Neural network architecture needs to be modified to handle the segmentation task instead of the classification task
- Output layer predicts a separate class label for each point, instead of a single class label for the entire point cloud



Neural Network Architectures

PointNet (2017)

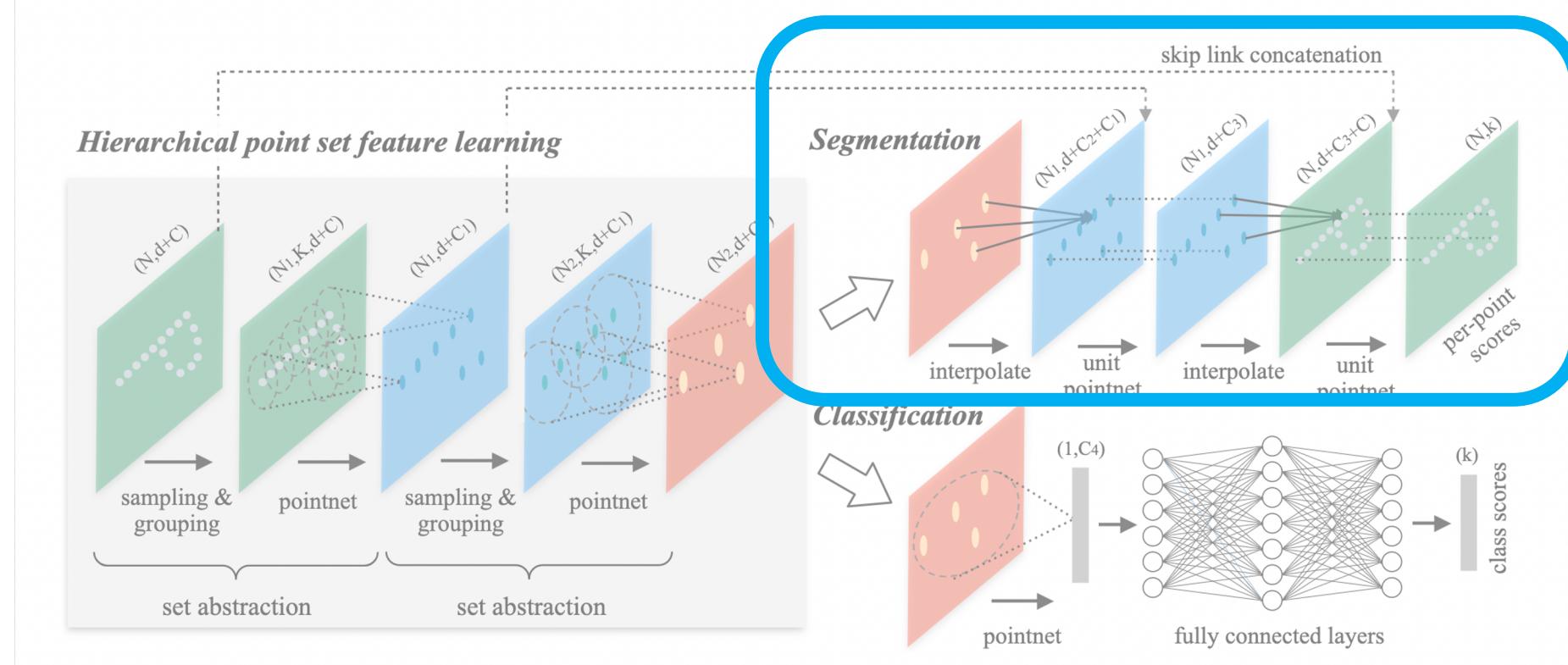
- One of the first architectures that can work directly with unordered point clouds
- Consists of point-wise convolution layers and a global pooling layer to combine features over the entire point cloud



Neural Network Architectures

PointNet++ (2017)

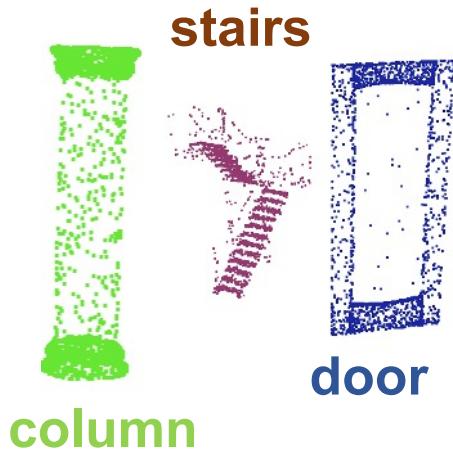
- Learn local features with increasing contextual scales
- Applies PointNet recursively on a nested partitioning of the input point set



Point Cloud Segmentation

Terminology in Point Cloud Object Recognition

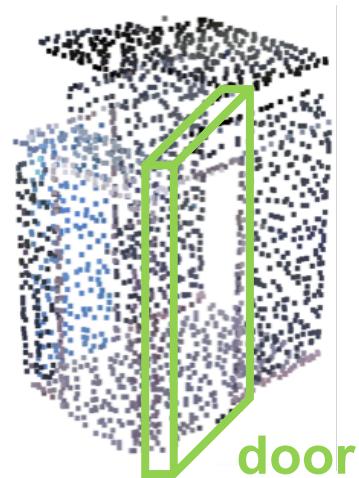
Classification



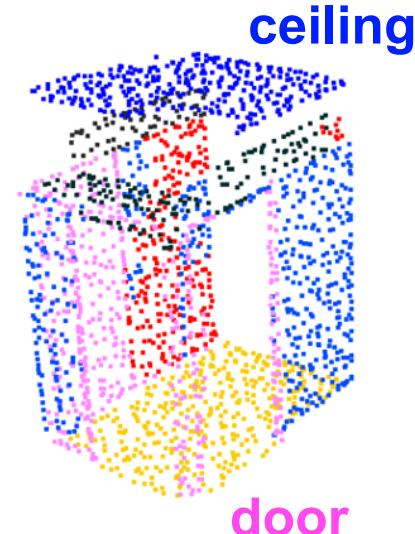
Part Segmentation



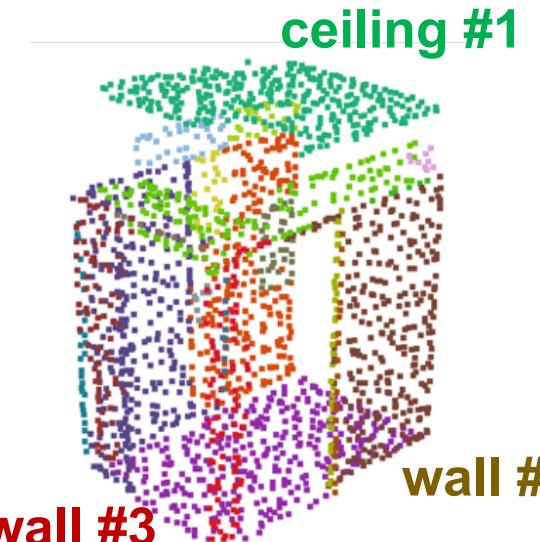
Bounding Box Detection



Semantic Segmentation



Instance Segmentation



Input: Object
Output: Object Label

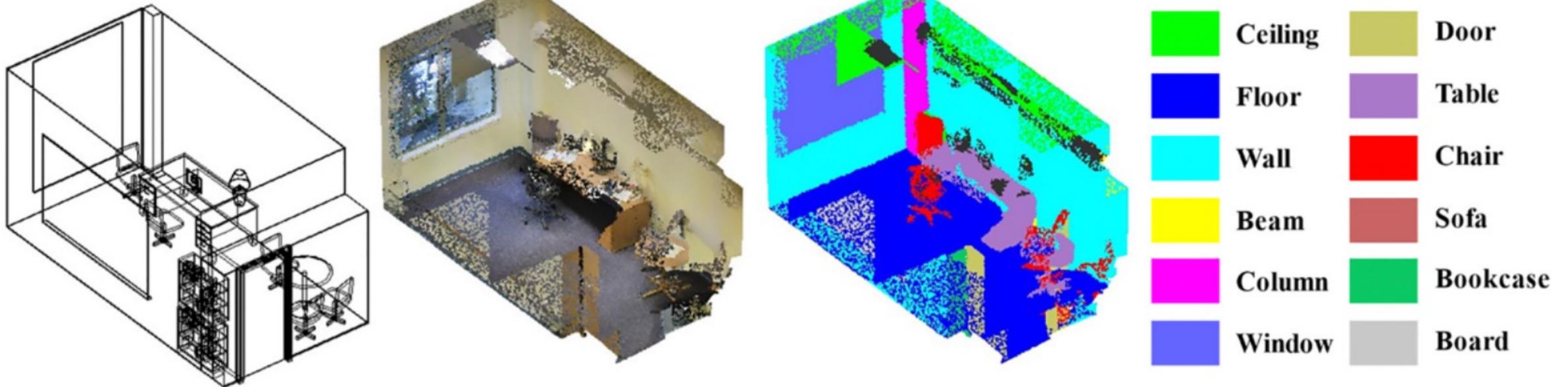
Input: Object
Output: Point Labels

Input: Scene
Output: Bounding Box

Input: Scene
Output: Point Labels

Input: Scene
Output: Instance ID + Object Labels

Point Cloud Segmentation



Ma et al. (2020). Semantic segmentation of point clouds of building interiors with deep learning: Augmenting training datasets with synthetic BIM-based point clouds. *AutoCon*.

Point Cloud Segmentation

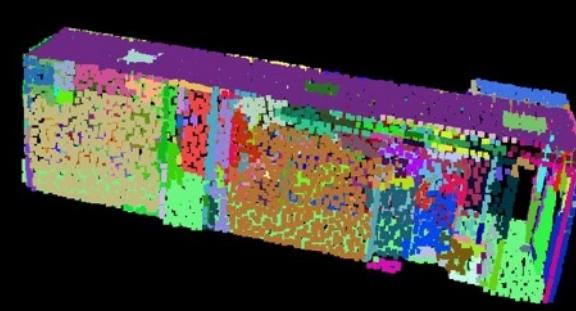
Hallway scene



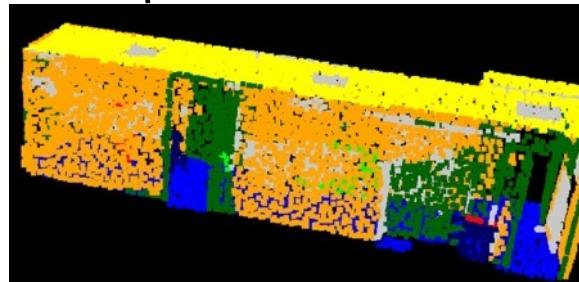
Input point cloud



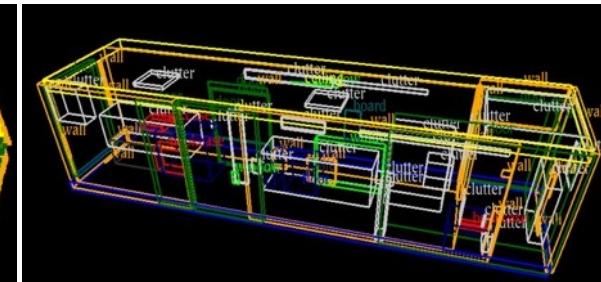
Output instance labels



Output semantic labels



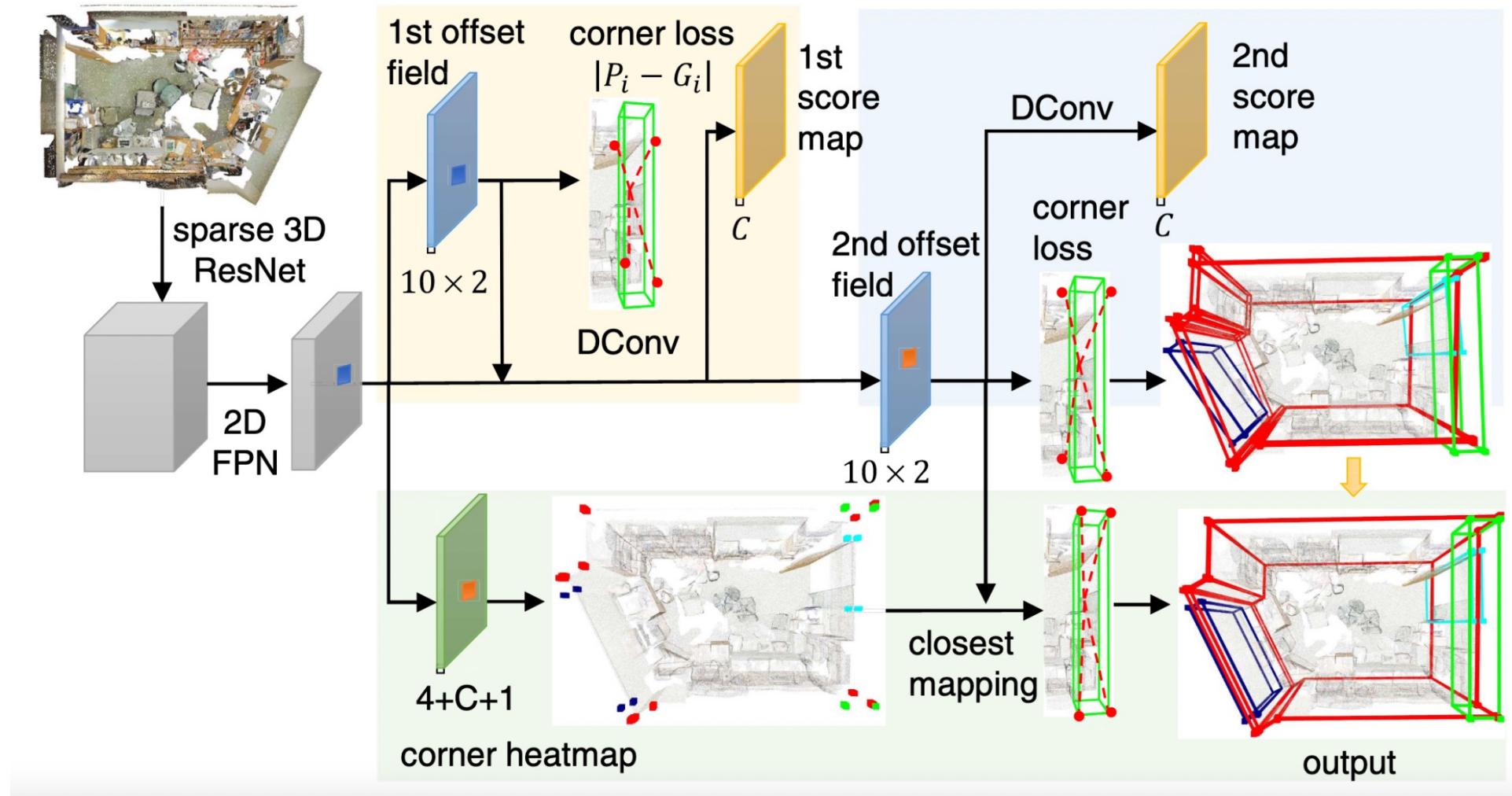
Output bounding boxes



● ceiling ● floor	● wall ● door	● window ● bookcase	● table ● sofa	● chair ● column	● beam ● board	● clutter
--	---	--	---	---	---	---

Chen et al. (2019). Deep Learning Approach to Point Cloud Scene Understanding for Automated Scan to 3D Reconstruction. JCCE.

Point Cloud Bounding Box Detection



Xu et al. (2019). CorDet: Corner-Aware 3D Object Detection Networks for Automated Scan-to-BIM. JCCE.



Future Work



Development Tools



Research Ideas



Discussion

Point Cloud Processing



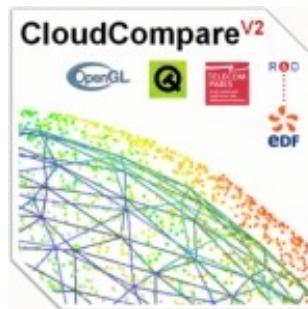
Open3D

<http://www.open3d.org/docs/release/>



Point Cloud Library

<https://pointclouds.org/>



CloudCompare

<https://www.danielgm.net/cc/>

Deep Learning



TensorFlow



DL4J



ONNX



PyTorch



PaddlePaddle

mxnet

Open-Source Code and Datasets



POINTCEPT

Point Cloud Perception Codebase



Papers With Code



Codebase for PyTorch implementations of popular deep learning models

<https://github.com/Pointcept/Pointcept/>

Collection of 3D datasets with CAD models, point clouds, 3D scenes etc.

<https://paperswithcode.com/datasets?mod=3d&page=1>

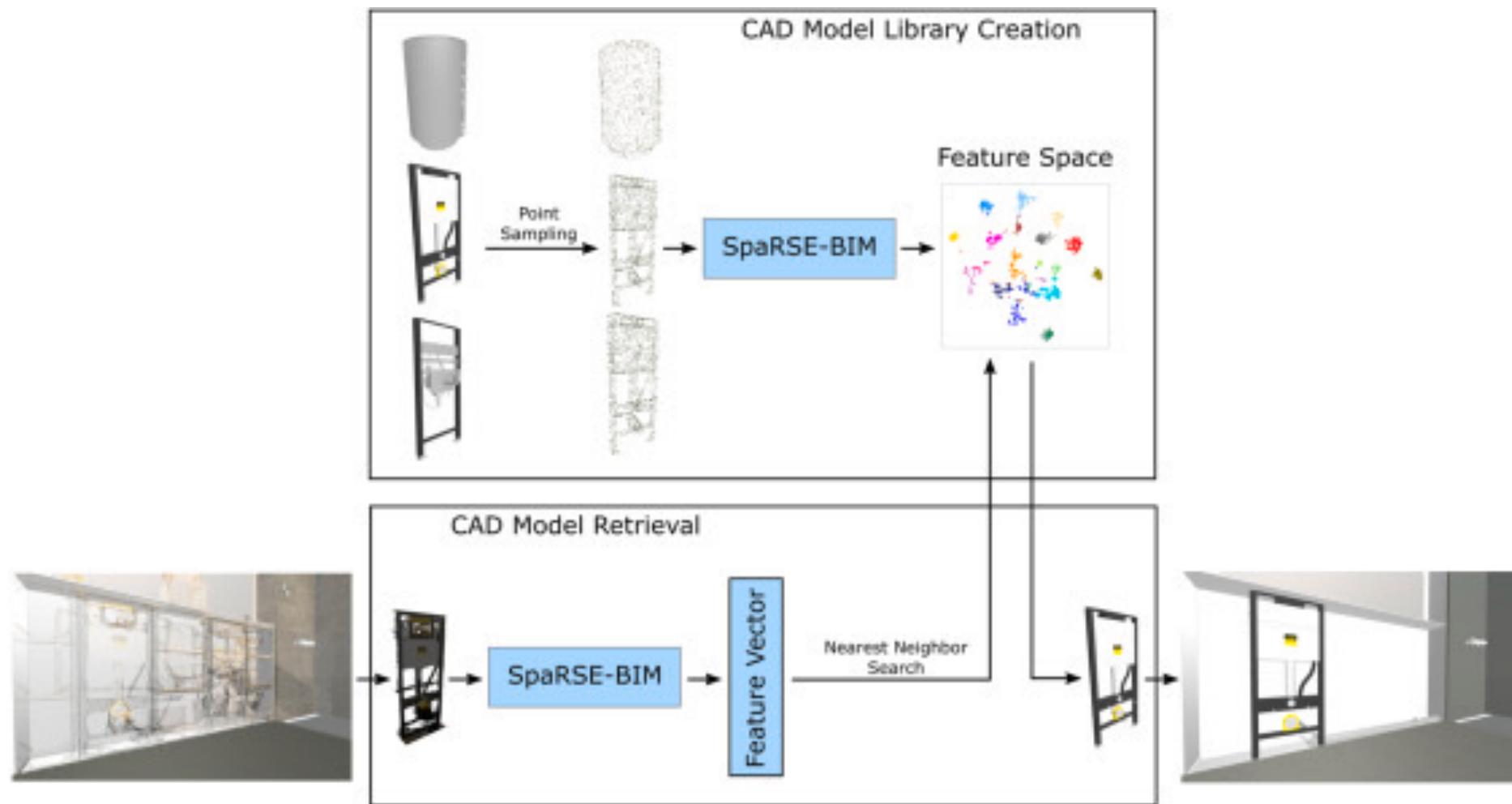
LivingBIM dataset

<https://zenodo.org/records/8084421>

Workshop on Computer Vision in the Built Environment at CVPR

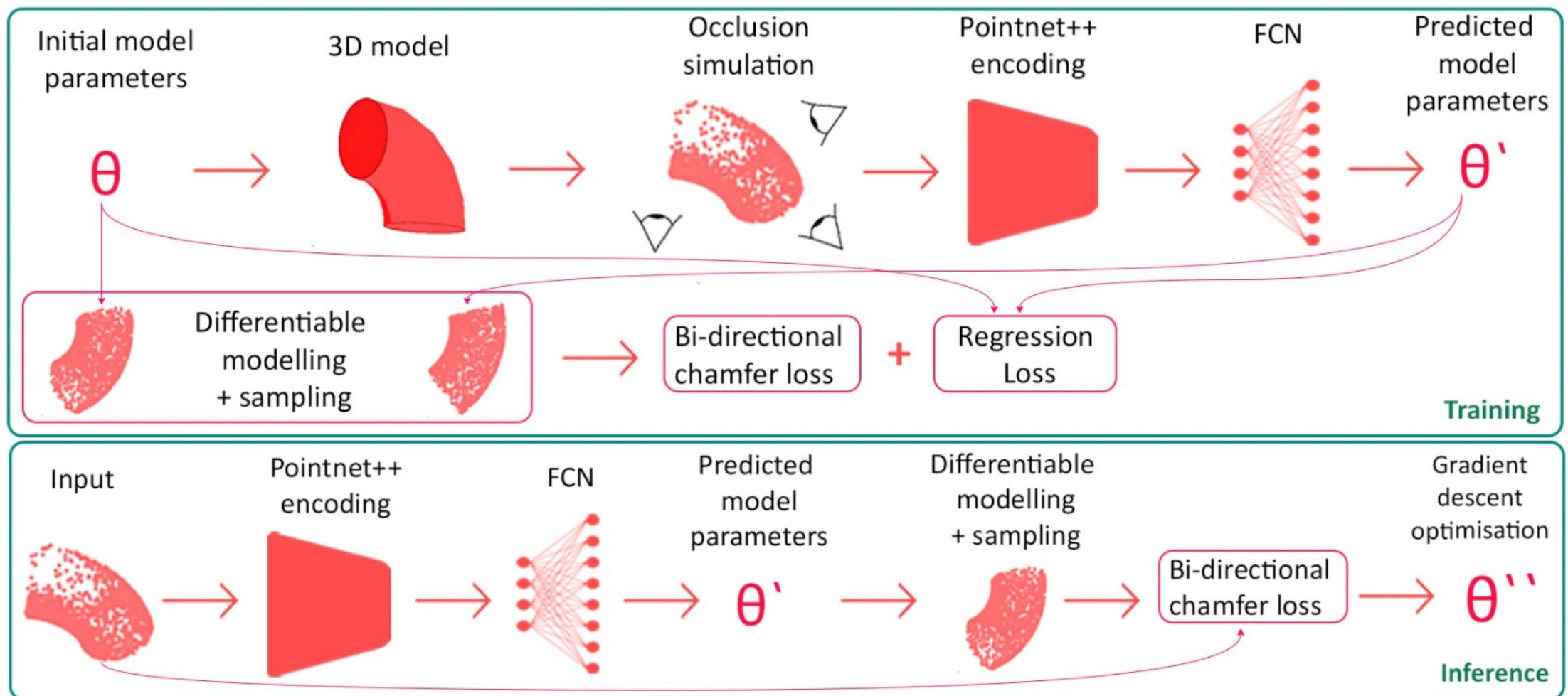
<https://cv4aec.github.io/>

Research Ideas



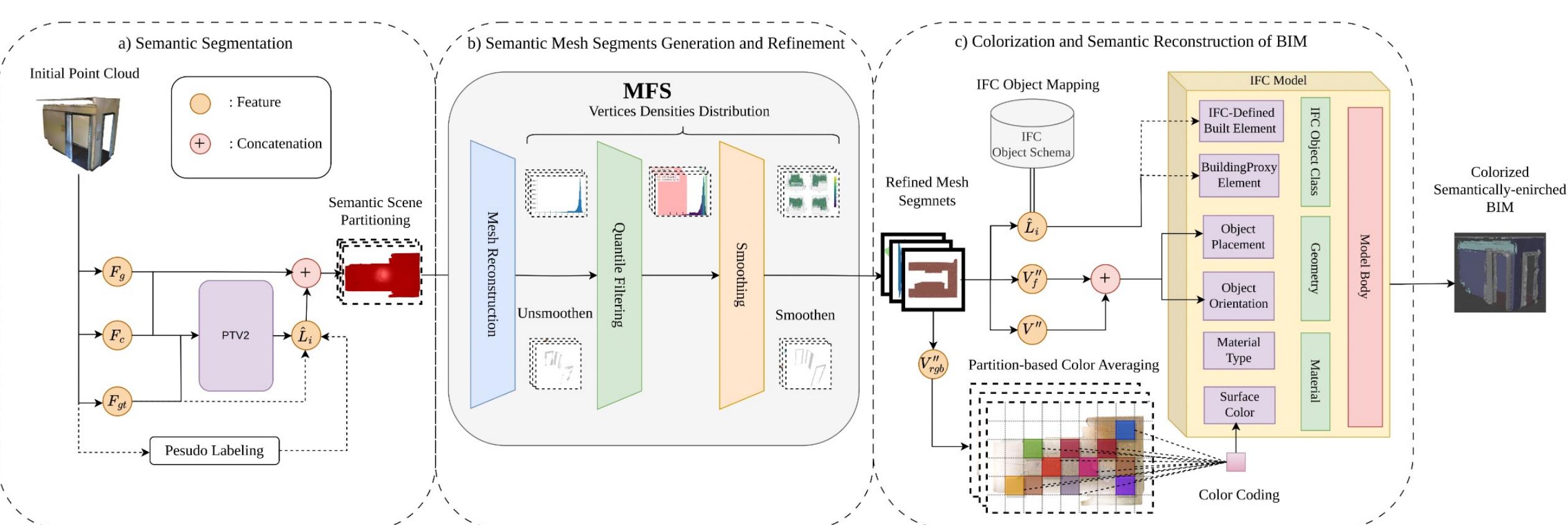
Emunds et al. (2022). SpaRSE-BIM: Classification of IFC-based geometry via sparse convolutional neural networks. Advanced Engineering Informatics

Research Ideas



Jayasinghe et al. (2023). Learnable Geometry and Connectivity Modelling of BIM Objects. BMVC

Research Ideas



Cheung et al. (2024). Towards Automating the Retrospective Generation of BIM Models: A Unified Framework for 3D Semantic Reconstruction of the Built Environment

Group Discussion

- What are some potential application areas in civil engineering for 3D deep learning technology?
- What are some industry needs or challenges that remain unmet by current point cloud or deep learning technology?
- What is needed to achieve scalability for a Scan-to-BIM system at the level of commercial AI systems such as ChatGPT or Google Bard?

Challenges

