STRUCTURAL ENGINEERING SUMMIT

CHICAGO: NOVEMBER 1-4, 2022

First Movements.

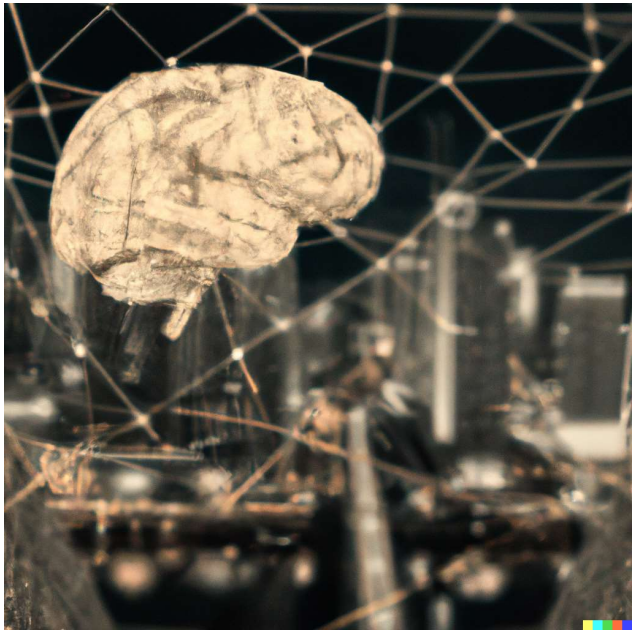
Application of Artificial Intelligence in Structural Design

Bill Mathers, PE

CUHACI PETERSON

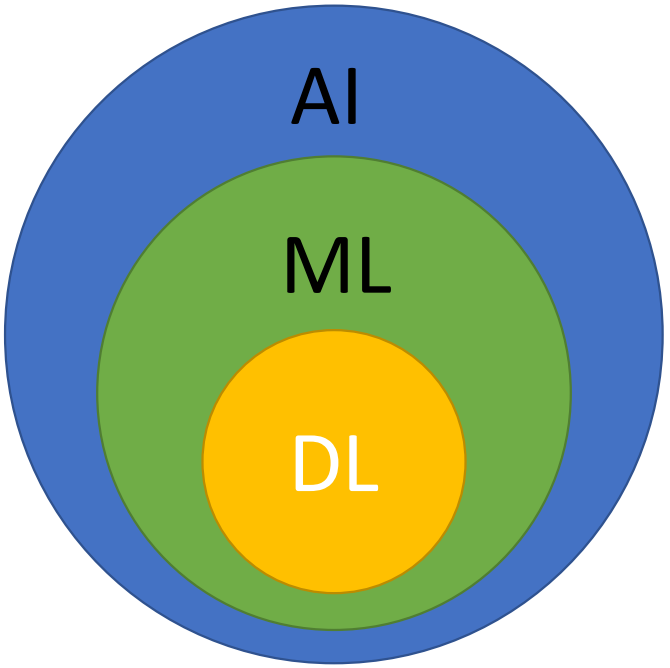
Learning Objectives

- Brief introduction to the current branches of AI
- Basic understanding of what an AI model is
- Spawn interest and provide a starting point
- Review current and potential applications



DALL-E2 "Structural Engineering Artificial Intelligence"

Artificial Intelligence



Artificial Intelligence

The theory and development of computer systems able to perform tasks that normally require human intelligence

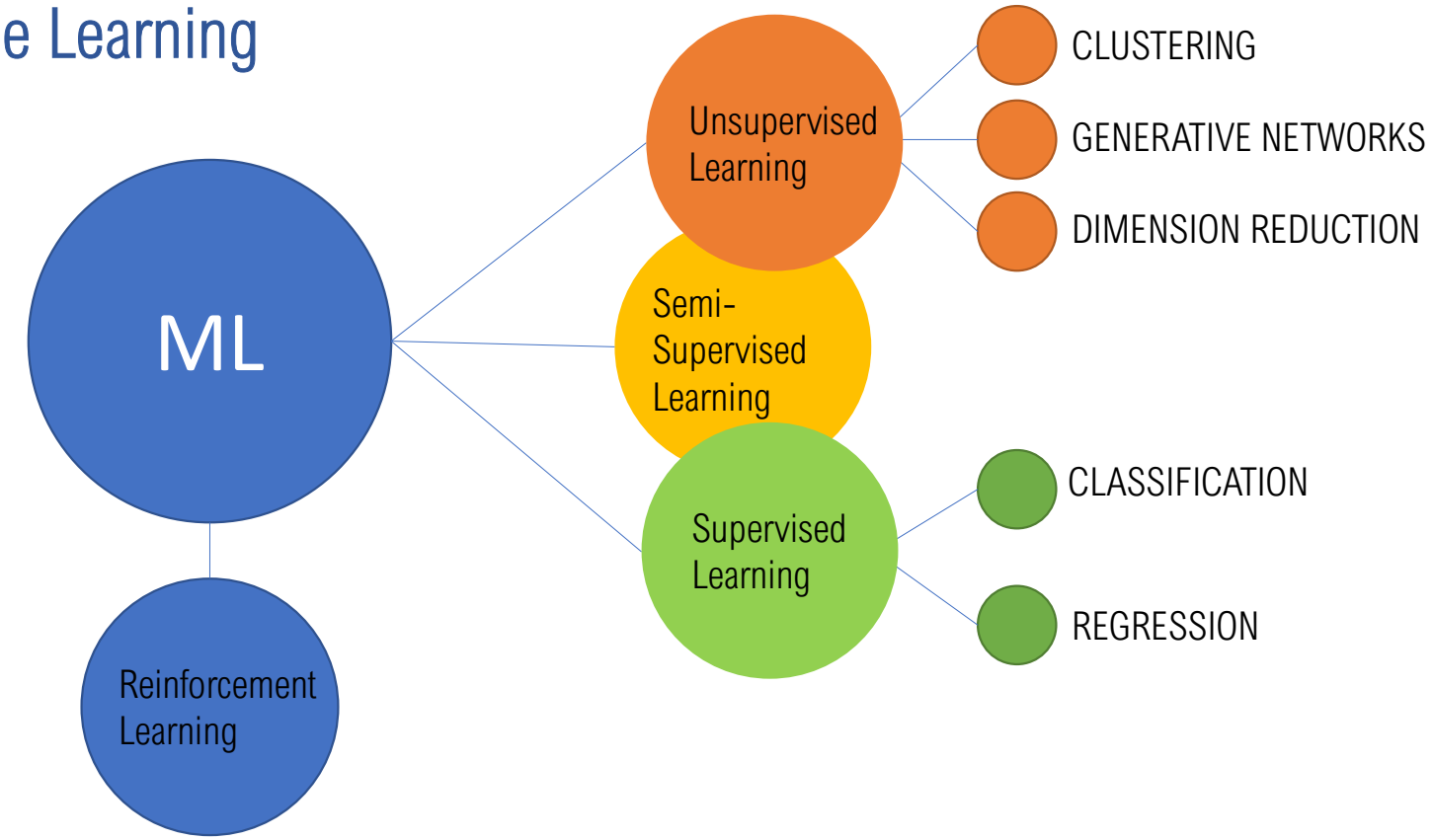
Machine Learning

A branch of artificial intelligence and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy

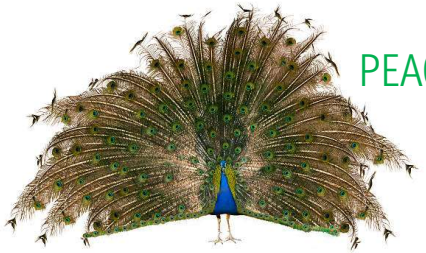
Deep Learning

A class of machine learning algorithms that uses a neural net of three or more layers to progressively extract higher-level features from the raw input

Machine Learning



Machine Learning vs. Deep Learning



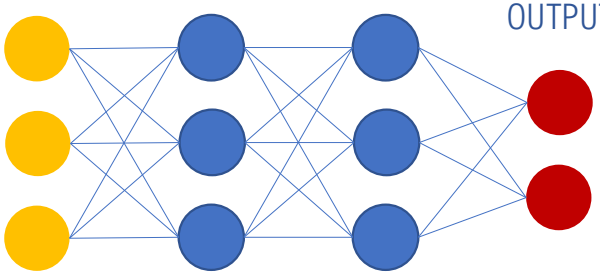
PEACOCK

PEACOCK

NOT PEACOCK

Machine Learning

DATA → FEATURE EXTRACTION (ONLY HUMAN) → INPUT → OUTPUT

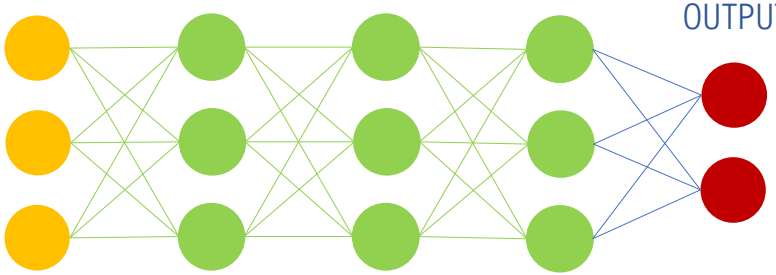


PEACOCK

NOT PEACOCK

Deep Learning

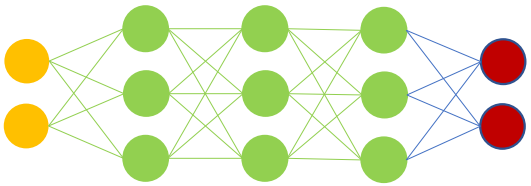
DATA → INPUT → OUTPUT



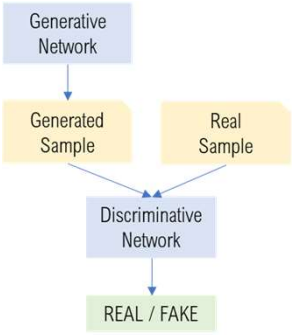
PEACOCK

NOT PEACOCK

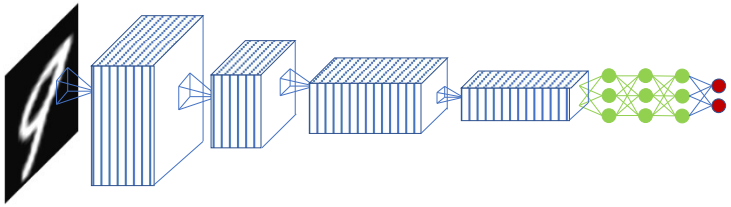
Dense Neural Network (DNN)



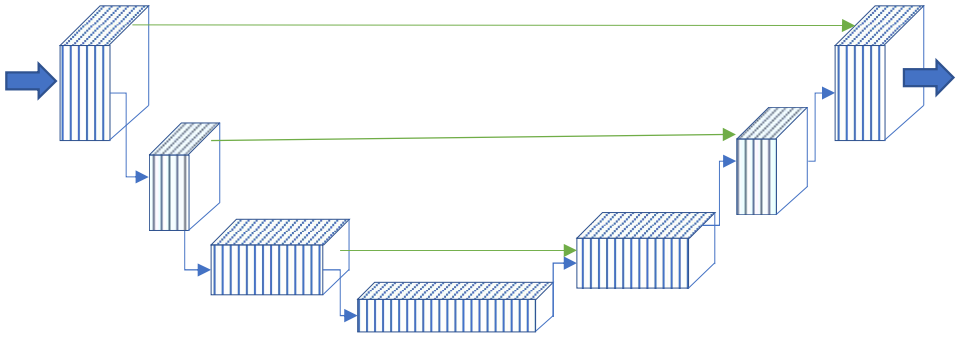
Generative Adversarial Networks (GANS)



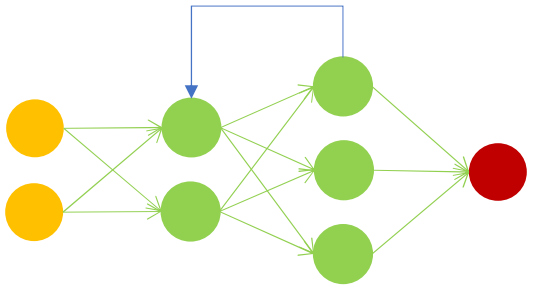
Convolutional Neural Network (CNN)



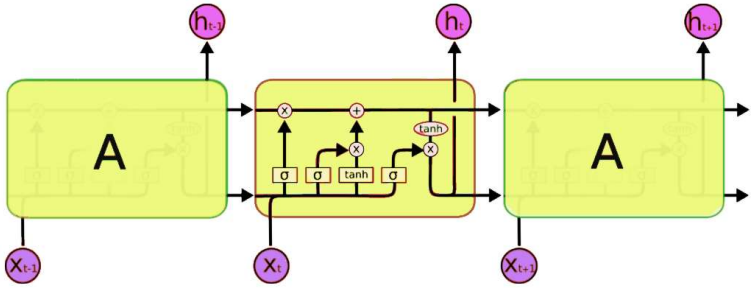
U-NET



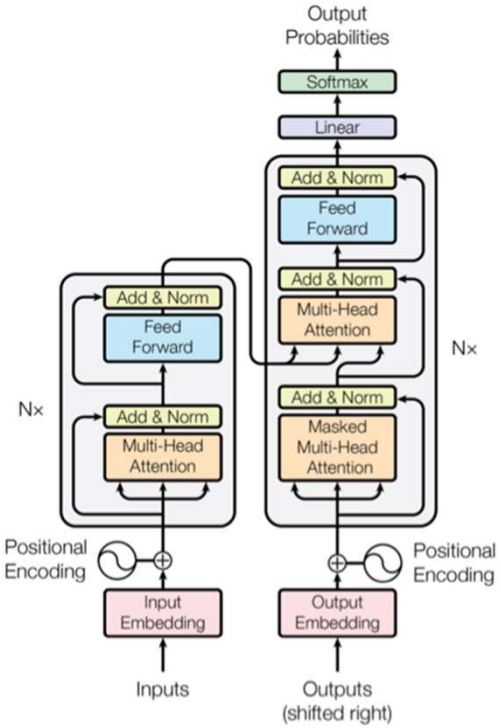
Recurrent Neural Network (RNN)



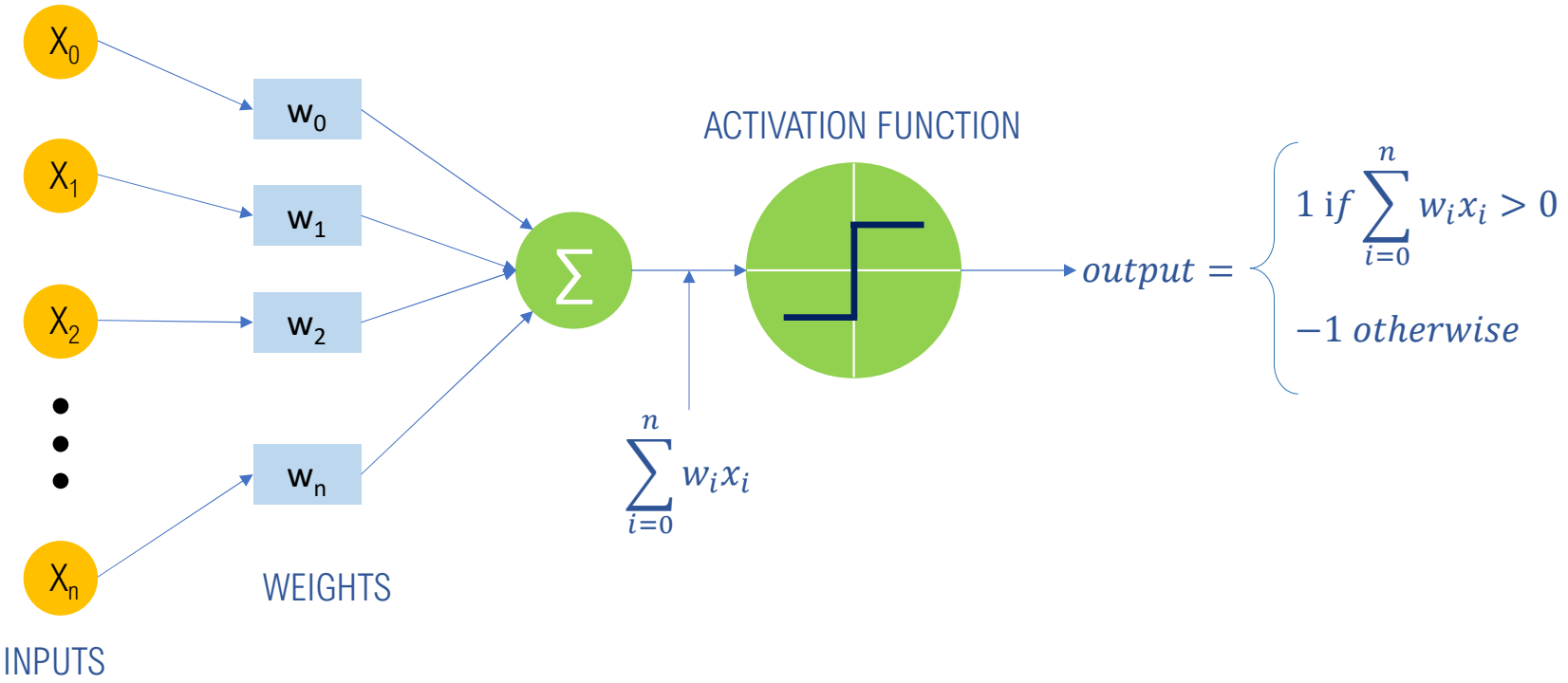
Long Short Term Memory (LSTM)



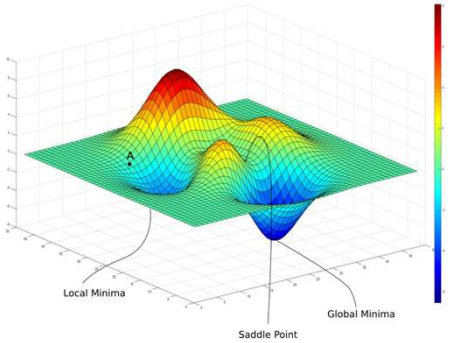
Transformer Networks



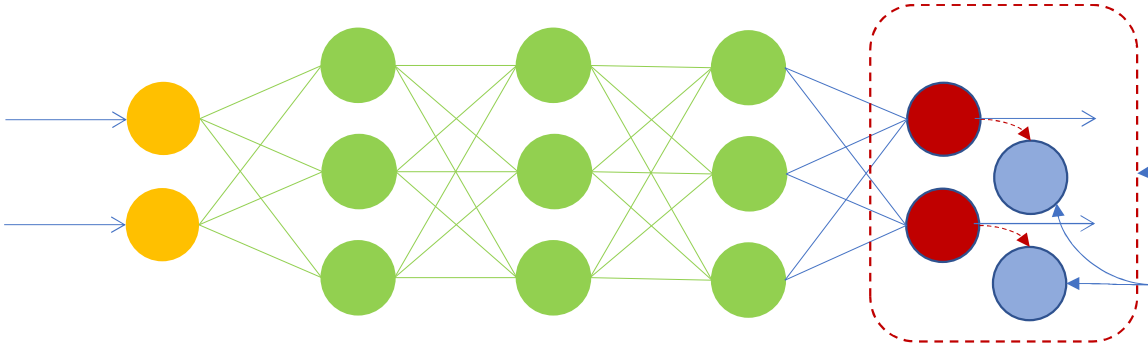
Perceptron: Fundamental Unit of a Neural Network



Feed Forward Network Basics



FORWARD PASS
COMPUTE ACTIVATIONS



COMPUTE LOSSES
BETWEEN OUTPUT AND
TRAINING DATA

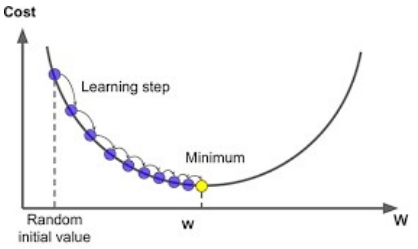
TRAINING DATA

INPUT LAYER

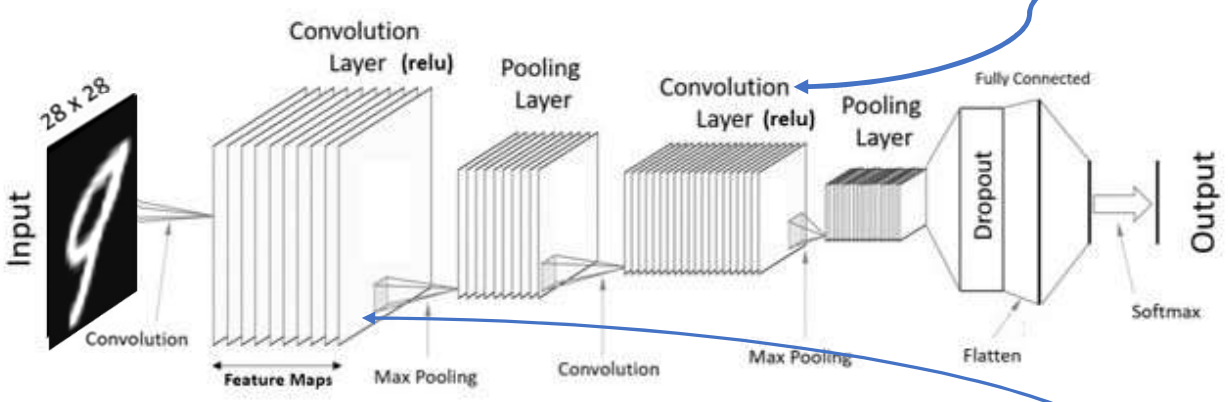
HIDDEN LAYERS

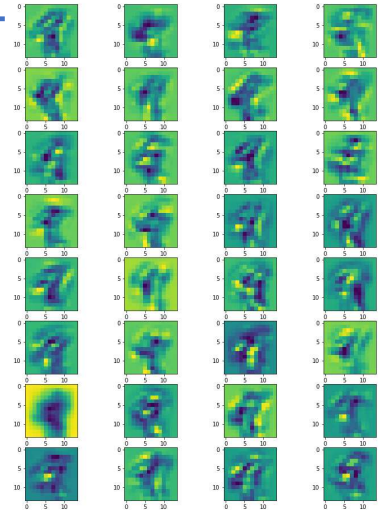
OUTPUT LAYERS

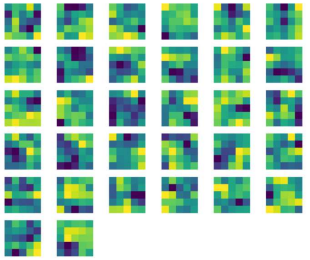
BACKWARD PASS
BACK PROPOGATION

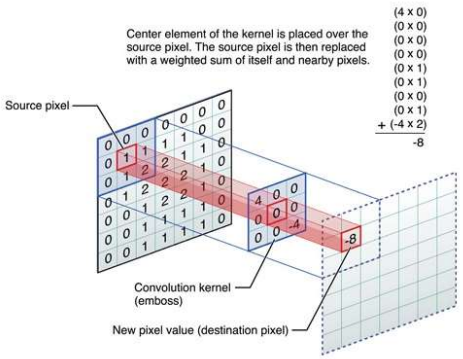


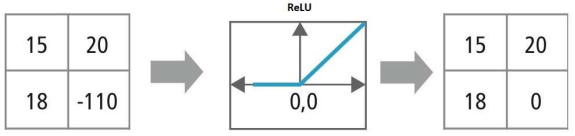
Convolutional Neural Networks

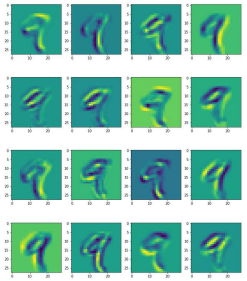


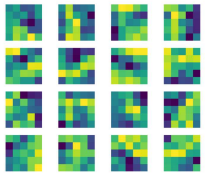




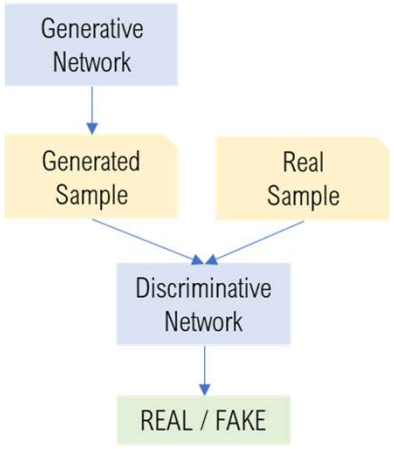


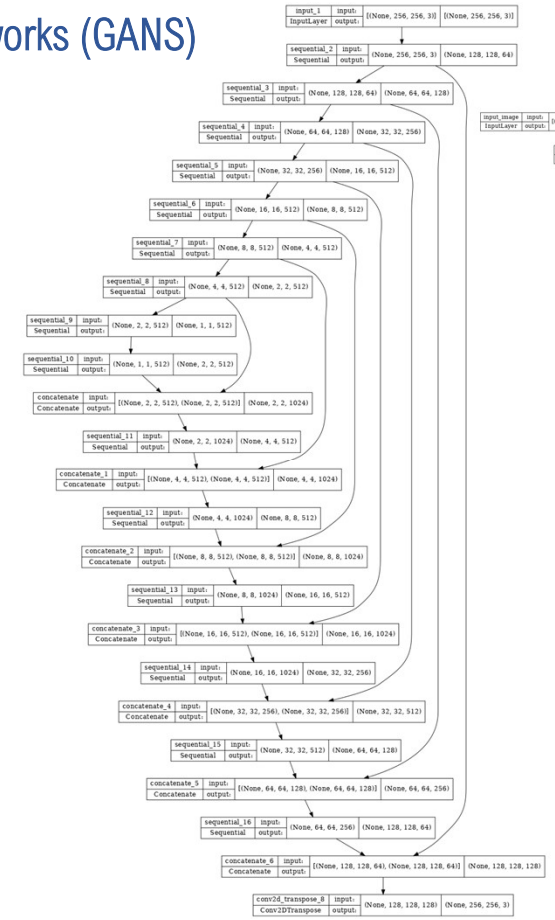


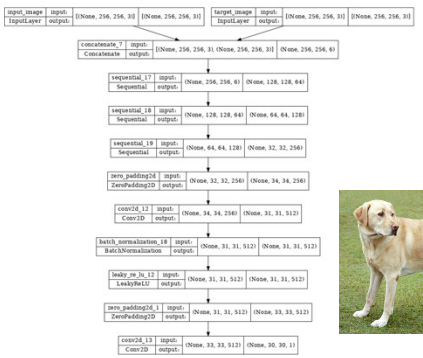


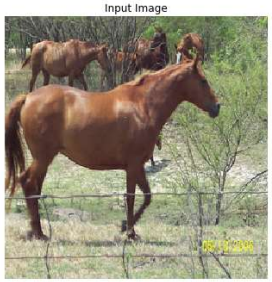


Generative Adversarial Networks (GANS)









Where do I begin?


python™

Computer needs

Google Colaboratory

Where to learn?

kaggle

I forgot most of your math skills…..

Khan Academy

YouTube

Platforms

Free models to use

Practical Application

Goal:
To develop an AI model that can do some basic framing layouts.

Development of Framebot – Version 0.0

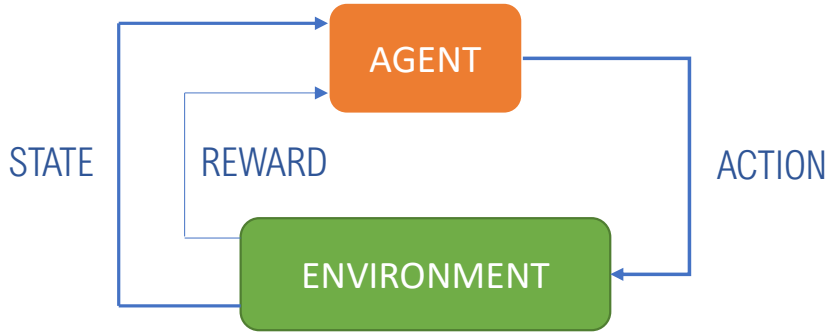
"Going in the wrong direction but making really good time"

Reinforcement Learning

- Bellman equations: Immediate reward plus discounted future values

Q-learning:

$$Q(s_t, a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}}$$



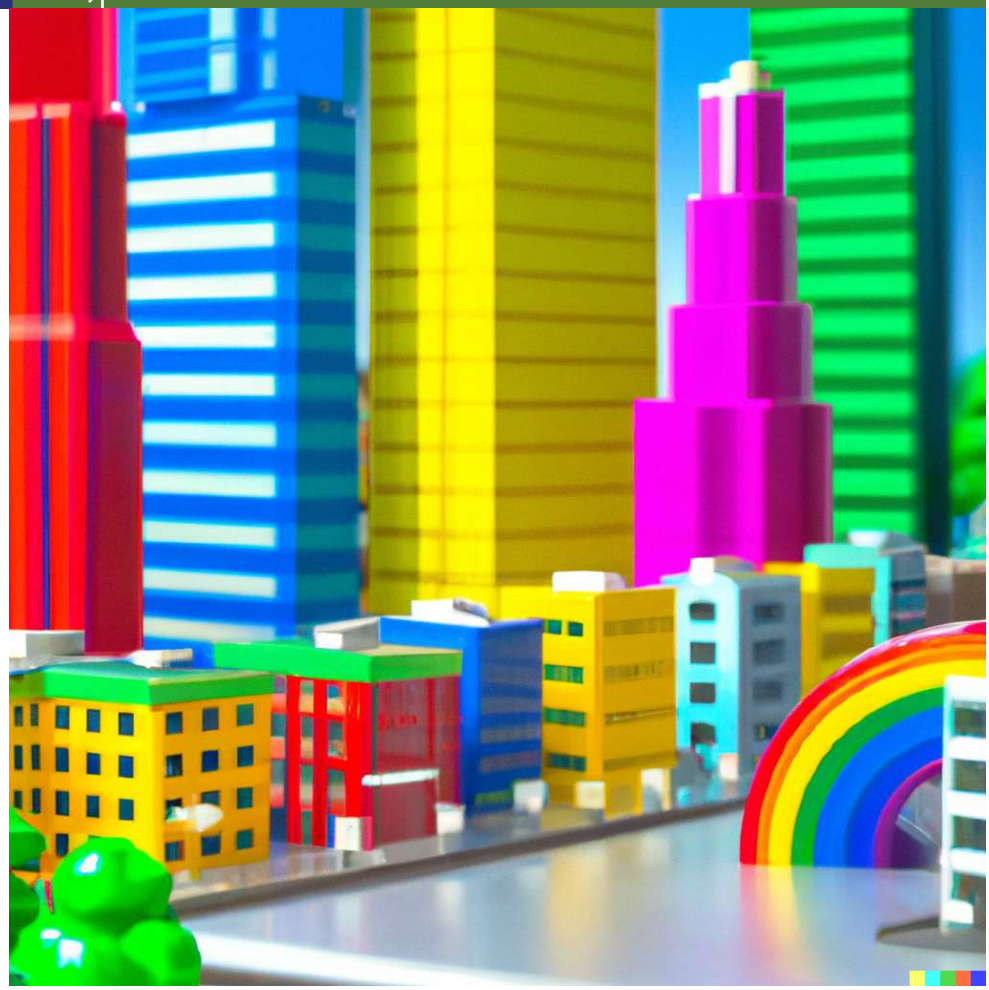
ACTION-REWARD FEEDBACK LOOP

ENVIRONMENT

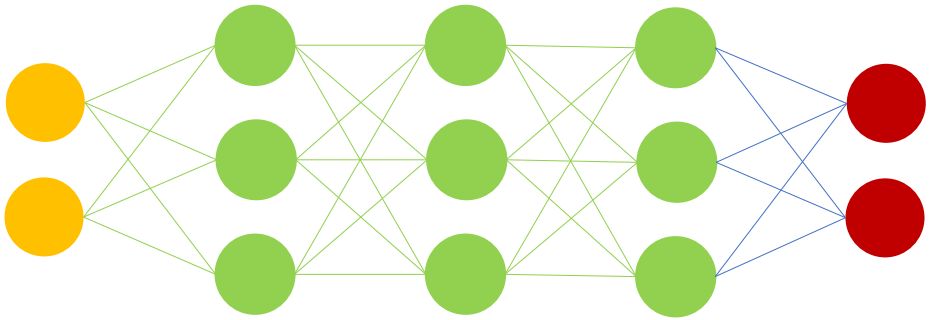
Deep Learning Challenges

- Large amounts of training data required
- No data is available

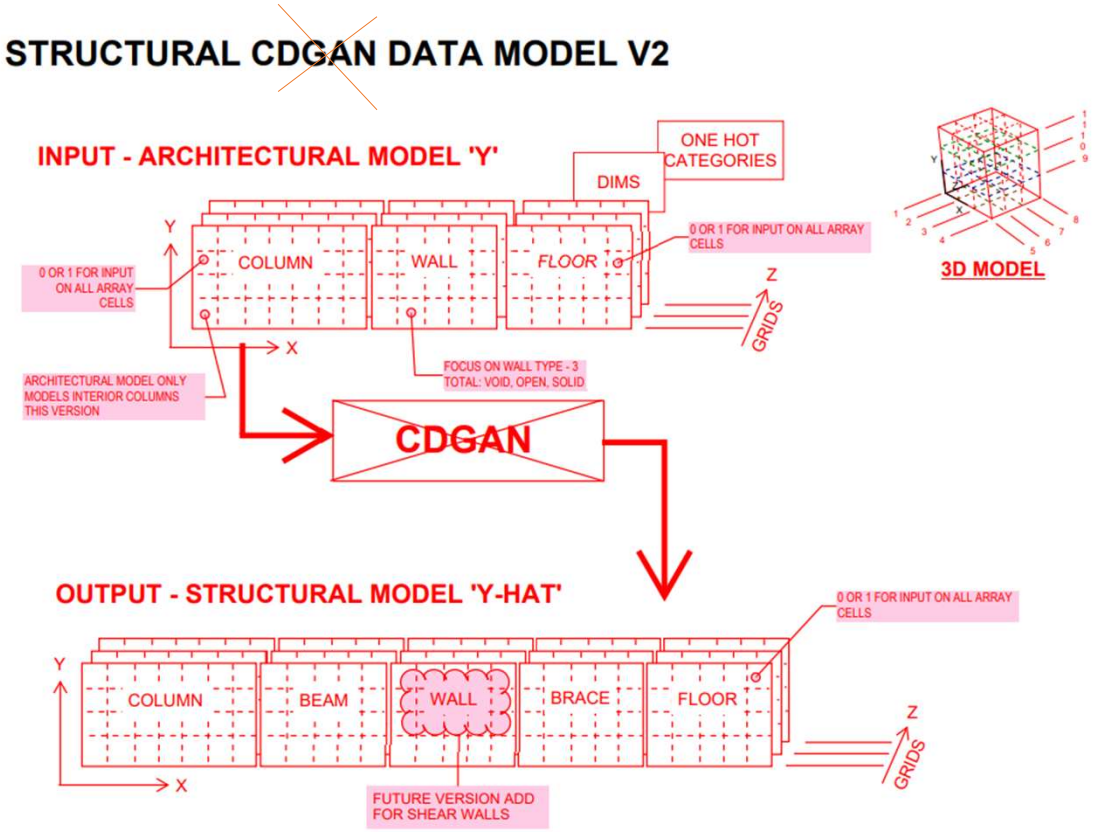
Synthetic Data



Framebot V1.0 – Dense Neural Network (DNN)

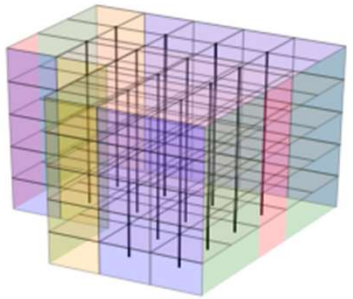


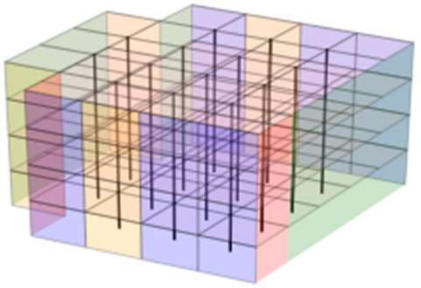
Custom Data

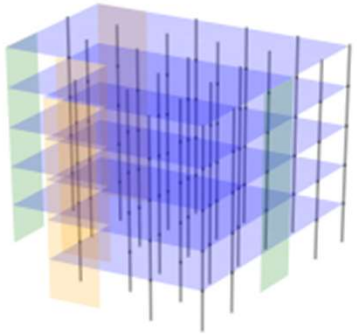

STRUCTURAL CDGAN DATA MODEL V2
INPUT - ARCHITECTURAL MODEL 'Y'
ONE HOT CATEGORIES
DIMS
0 OR 1 FOR INPUT ON ALL ARRAY CELLS
0 OR 1 FOR INPUT ON ALL ARRAY CELLS
COLUMN
WALL
FLOOR
Y
X
Z
GRIDS
3D MODEL
ARCHITECTURAL MODEL ONLY MODELS INTERIOR COLUMNS THIS VERSION
FOCUS ON WALL TYPE - 3 TOTAL: VOID, OPEN, SOLID
CDGAN
OUTPUT - STRUCTURAL MODEL 'Y-HAT'
0 OR 1 FOR INPUT ON ALL ARRAY CELLS
COLUMN
BEAM
WALL
BRACE
FLOOR
Y
X
Z
GRIDS
FUTURE VERSION ADD FOR SHEAR WALLS

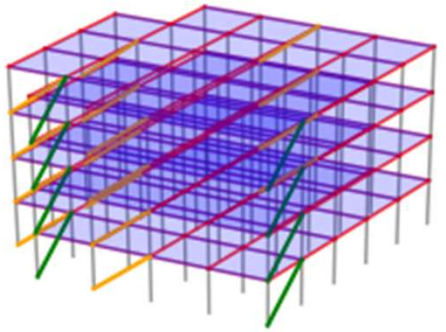
Rendered Training Data

- Random shapes and heights
- Braced frame or concrete with shear walls
- 200,000 data pairs +/-



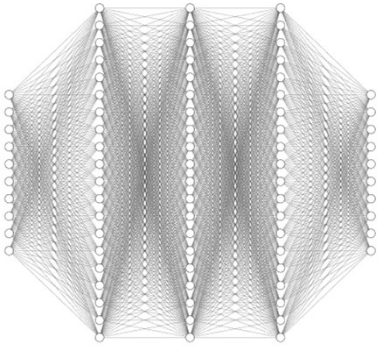






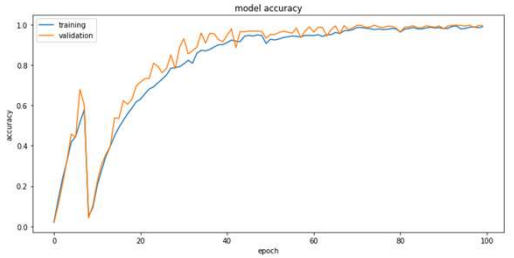
DNN Model

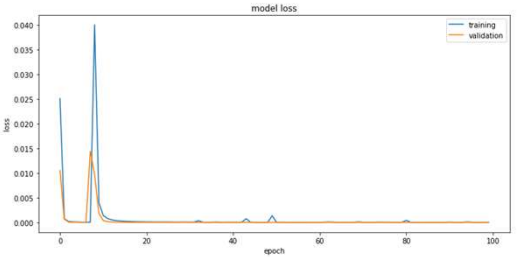
- Huge model
- Input with 1050 nodes
- 3 hidden layers with 2048 nodes each
- Output has 1620 nodes
- 13,872,724 parameters
- Accuracy 98%

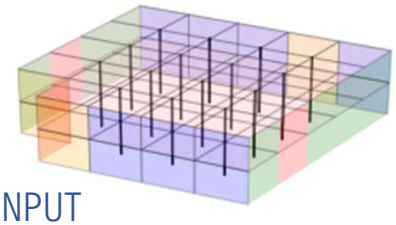


MODEL

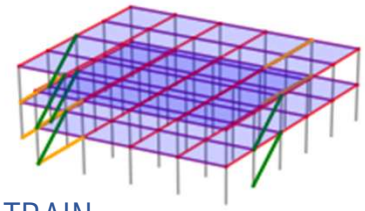
```
with tpu_strategy.scope():
    model = tf.keras.Sequential([
        tf.keras.layers.Input(shape=(1050,)),
        # unit 0
        tf.keras.layers.Dense(2048, activation='elu'),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Dropout(rate=0.2),
        # unit 1
        tf.keras.layers.Dense(2048, activation='elu'),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Dropout(rate=0.2),
        # unit 2
        tf.keras.layers.Dense(2048, activation='elu'),

        tf.keras.layers.Dense(1620, activation='sigmoid')
    ])

    model.compile(optimizer=tf.keras.optimizers.Adam(),
                  loss=tf.keras.losses.BinaryCrossentropy(from_logits=False),
                  metrics=['accuracy'])
```



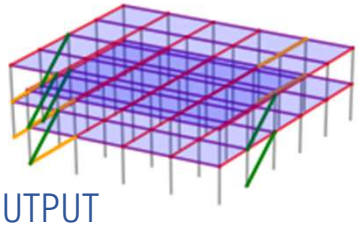




INPUT



TRAIN



OUTPUT

Framebot V1.0 Demo

Framebot V2.0 UNET