PANTECH SOLUTIONS®
Technology Beyond the Dreams

AI Master Class series – Day 10

Overview on CNN | Designing Neural Network

Day-10 Agenda.

01.

Neural Network

Neural Network & Perceptron

02.

Deep Learning Algorithm

Overview & Types

03.

CNN & CNN Architecture

Architecture with Softmax Activation Function

04.

Implementation of NN

Implementation & Basic Syntax on Keras

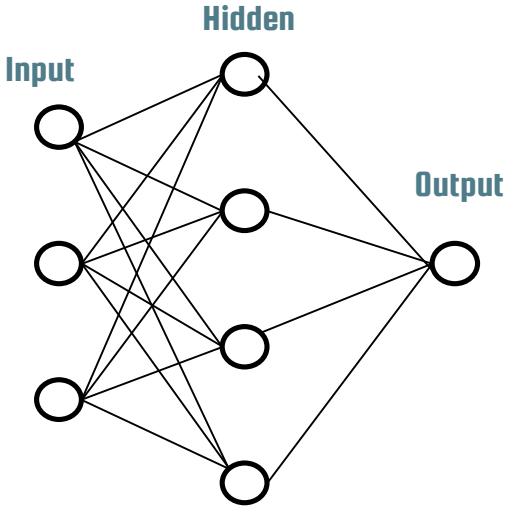
05.

Designing Neural Network

Pima Indians diabetes classification

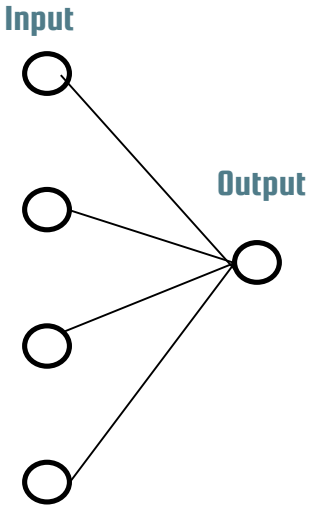
Neural Network.

- It has Interconnected input and output In which each connection has an associated weights
- It will adjust the weight during the learning process



Perceptron.

- Only Input and output Layer no hidden layer
- Simple decision making



Deep Learning Algorithm.

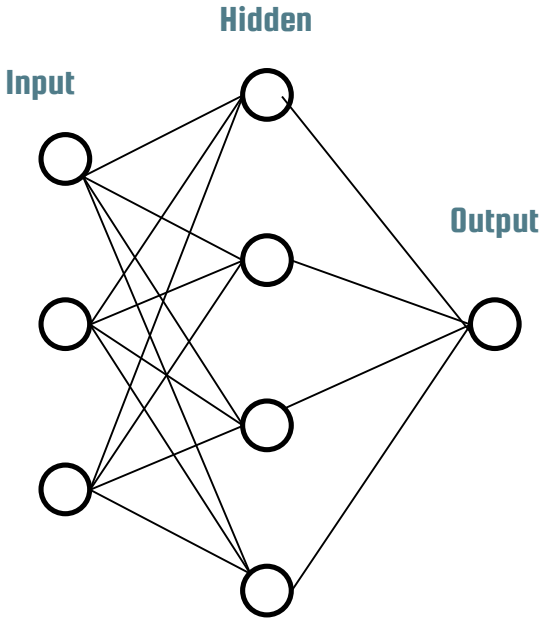
01 ARTIFICIAL NEURAL NETWORK (ANN)

02 RECURRENT NEURAL NETWORK (RNN)

01 CONVOLUTIONAL NEURAL NETWORK (CNN)

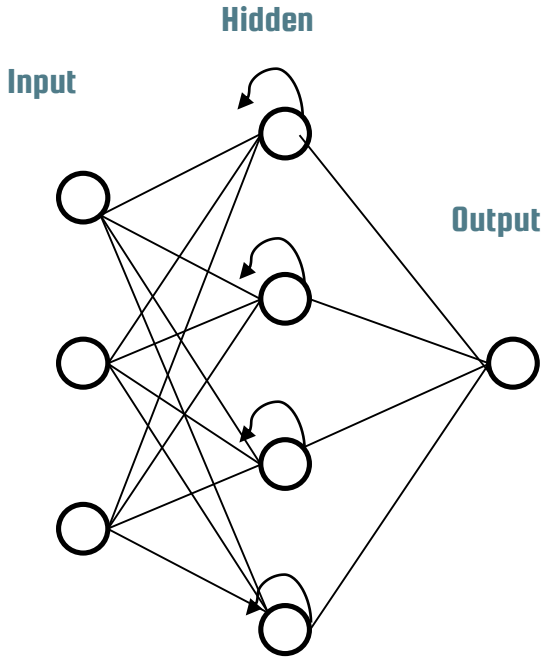
ANN.

- Learns any Non-Linear Function, It is known as Universal Function Approximators
- Activation Function introduce non linear property to network, so it will identify complex relationship between input & output
- Output of each neuron is the activation of weighted sum of Input, If there is no Activation function, network can't learn non-linear function
- Feed Forward Neural Network – Input processed in one direction
- When hidden layer is more than one, that is Deep Neural Network



RNN.

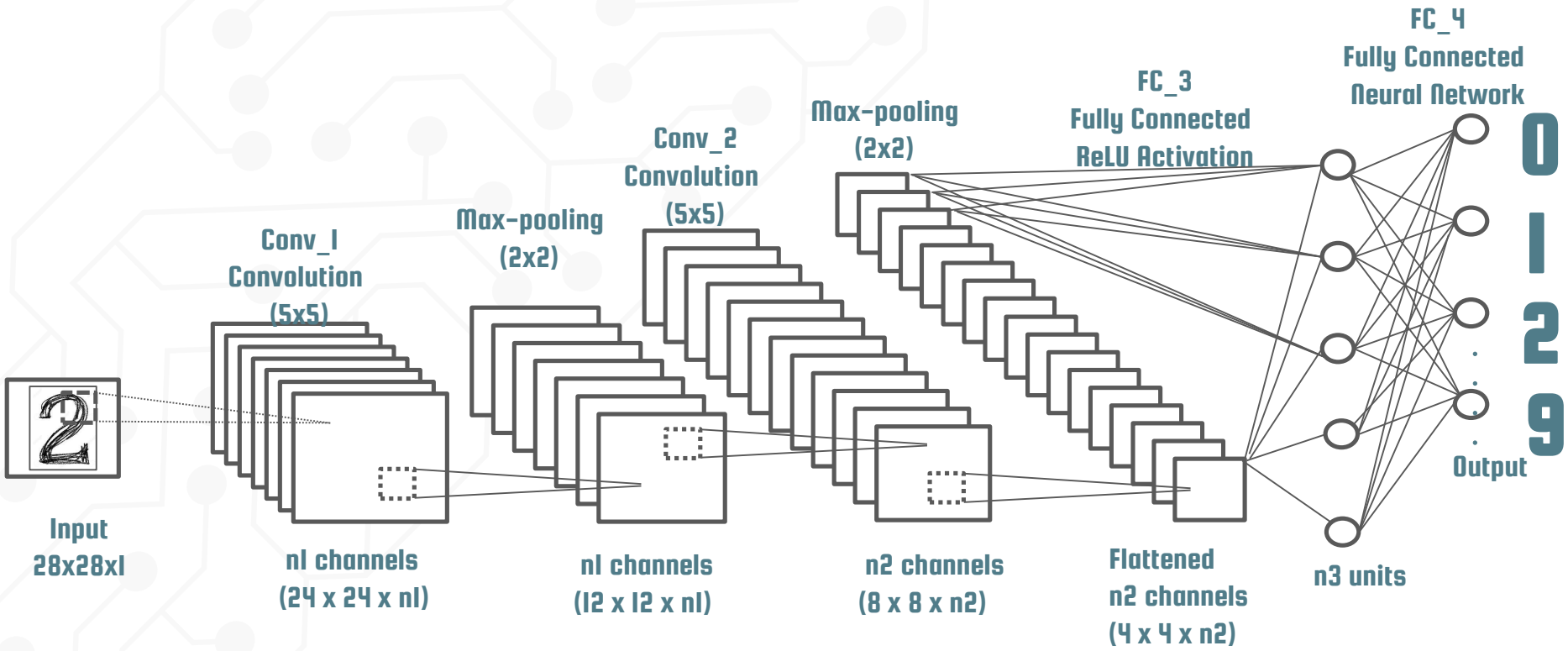
- Looping system in hidden layer of ANN is known as RNN
- It captures sequential info of input data, that is dependency between words to make prediction. Whereas, ANN cannot capture sequential information
- RNN shares parameters across different time steps, so that there will be few parameter to train
- It is the time series version of ANN. Common Recurrent layers are LSTM(Long Short Term Memory) & GRU (Grated Recurrent Units)
- GRU is used to how much pass data needed to flow through model
- It is mostly used in NLP (Natural Language Processing)



CNN.

- CNN learns the filter automatically to extract the right features from the data
- It captures spatial features (Arrangement of pixels) whereas ANN can't.
- It also follows parameter sharing like RNN, applies single filter in different part of single image. Whereas ANN can't.
- It don't have recurrent connections like RNN, instead it has convolution type of hidden layers
- Convolution and pooling functions are used as activation functions
- CONVOLUTION: Input image and other as Filter on input image(Kernel) produces output image.
- POOLING: picking maximum value from selected region is Max pooling and vice versa.

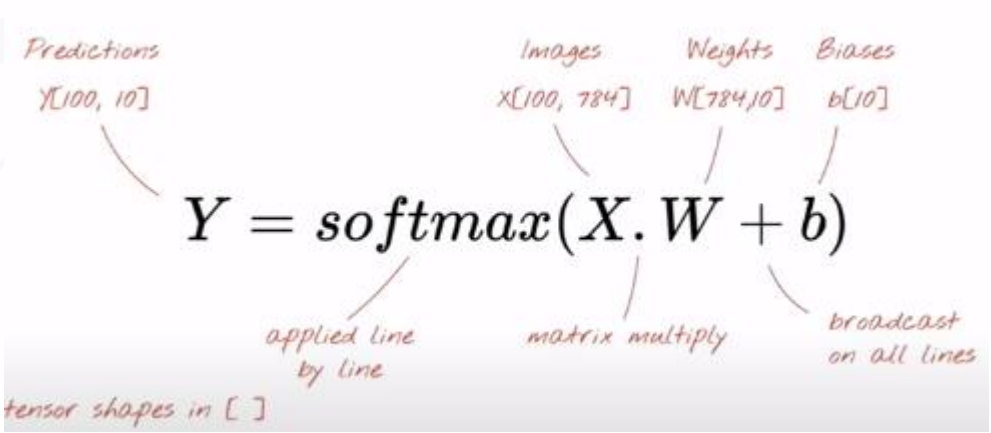
CNN Architecture.



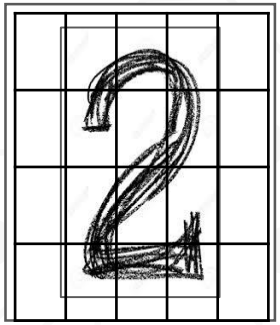
Simple Softmax Classification.

$$softmax(L_n) = \frac{e^{L_n}}{||e^{L}||}$$

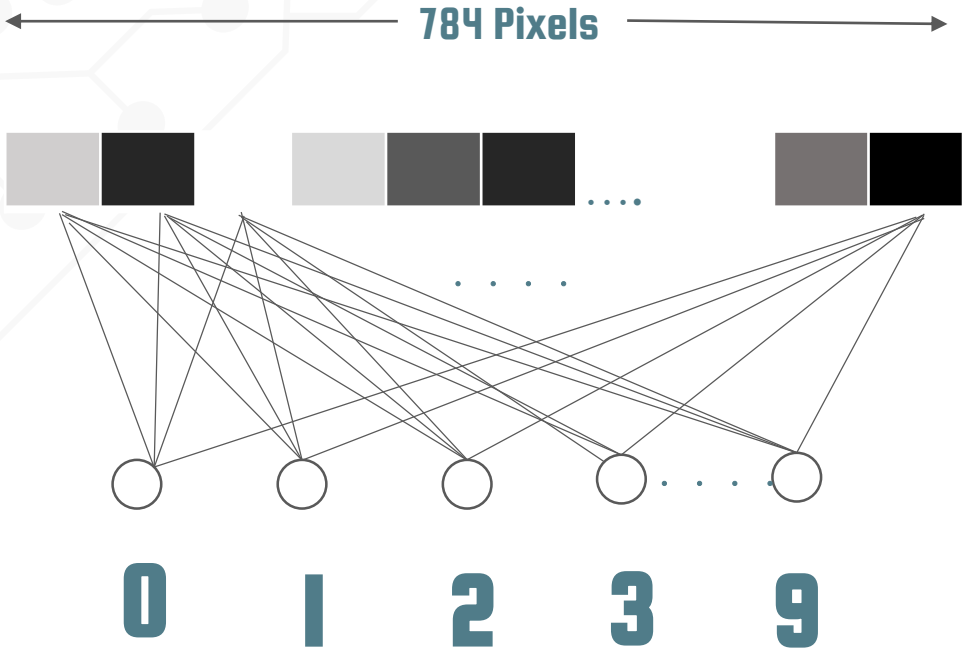
$$L = X.W + b$$



Simple Softmax Classification.



Input
28x28xl



100 image at a time.

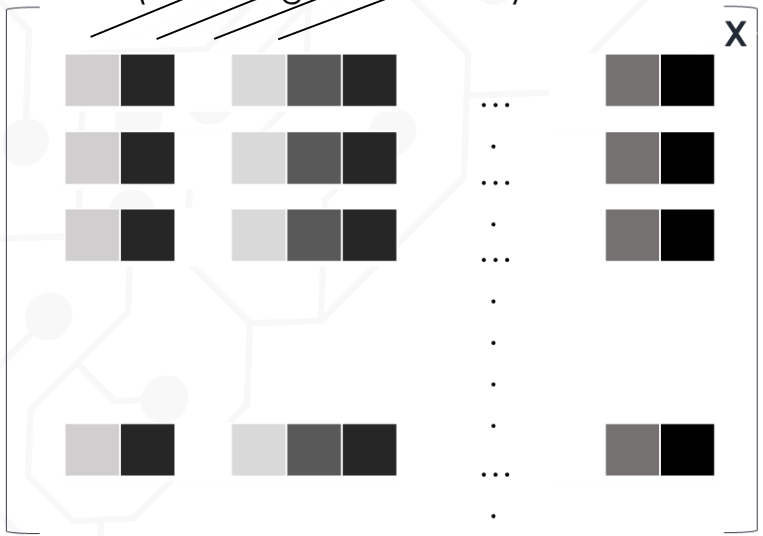
Weights
784x10 Matrix

$$\begin{bmatrix} W_{0,0} & W_{0,1} & W_{0,2} & \dots & W_{0,9} \\ W_{1,0} & W_{1,1} & W_{1,2} & \dots & W_{1,9} \\ W_{2,0} & W_{2,1} & W_{2,2} & \dots & W_{2,9} \\ W_{3,0} & W_{3,1} & W_{3,2} & \dots & W_{3,9} \\ W_{4,0} & W_{4,1} & W_{4,2} & \dots & W_{4,9} \\ \vdots & & \vdots & & \\ W_{783,0} & W_{783,1} & W_{783,2} & \dots & W_{783,9} \end{bmatrix}$$

Dot Product

X

Input Image
100x784 (100 images Flattened)



100x10 Matrix

$$\begin{bmatrix} L_{0,0} & W_{0,1} & W_{0,2} & \dots & W_{0,9} \\ L_{1,0} & W_{1,1} & W_{1,2} & \dots & W_{1,9} \\ L_{2,0} & W_{2,1} & W_{2,2} & \dots & W_{2,9} \\ L_{3,0} & W_{3,1} & W_{3,2} & \dots & W_{3,9} \\ L_{4,0} & W_{4,1} & W_{4,2} & \dots & W_{4,9} \\ \vdots & & \vdots & & \\ L_{99,0} & L_{99,1} & L_{99,2} & \dots & W_{99,9} \end{bmatrix}$$

+

Bias
1x10 Matrix

$$\begin{bmatrix} b_0 & b_1 & b_2 & b_3 & \dots & b_9 \end{bmatrix}$$

In TensorFlow.

Y[100x10]

X[100x784]

W[784x10]

b[10]

```
Y = tf.nn.softmax(tf.matmul(X,W)+b)
```

Cross Entropy

Actual Probability

0	1	2	3	4	5	6	7	8	9
0	0	1	0	0	0	0	0	0	0

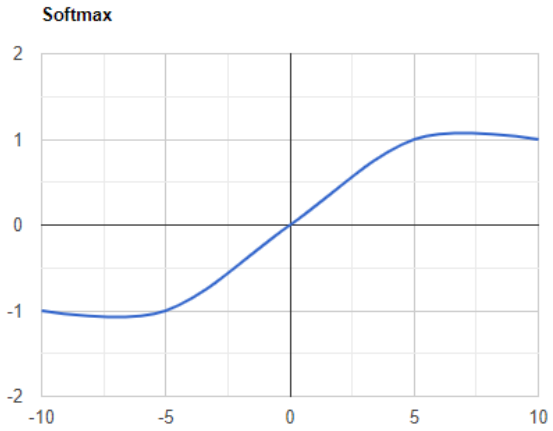
$$-\sum Y'_i . log(Y_i)$$

Computed Probability

0	1	2	3	4	5	6	7	8	9
0.1	0.2	0.9	0.5	0.3	0.1	0.2	0.1	0.3	0.1

SOFTMAX Function.

- Softmax activation function will be applied in the last layer of Neural network, instead of ReLU, tanh, Sigmoid.
- It is used to map the non-normalized output of a network to a probability distribution over predicted output class. That is it converts output of last layer into a essential probability distribution.



$$softmax(L_n) = \frac{e^{L_n}}{||e^{L}||}$$

Implementation of NN.

Feed Forward

- Set of Input features and random weights
- Weights will be optimized by back propagation

Back Propagation.

- Calculating error between predicted output and target output and use Gradient descent method to update weights

Gradient Descent.

- Machine Learning algorithm
- It operates iteratively to find the optimal values for its parameters. user-defined learning rate, and initial parameter values

Vanishing & Exploding Gradient.

- It is very common problem in every Neural Network, which is associated with Backpropagation.
- Weights of network are updated through backpropagation by finding gradients.
- When the number of hidden layer is high, then the gradient vanishes or explodes as it propagates backward. It leads instability in network, unable to learn from training
- The explosion occurs through exponential growth by repeatedly multiplying gradients through the network layers that have values larger than 1.0
- It can be fixed by redesigning the network, using Long Short Term Memory networks, Gradient clipping, etc.

Keras Basic Syntax.

Adding Layers

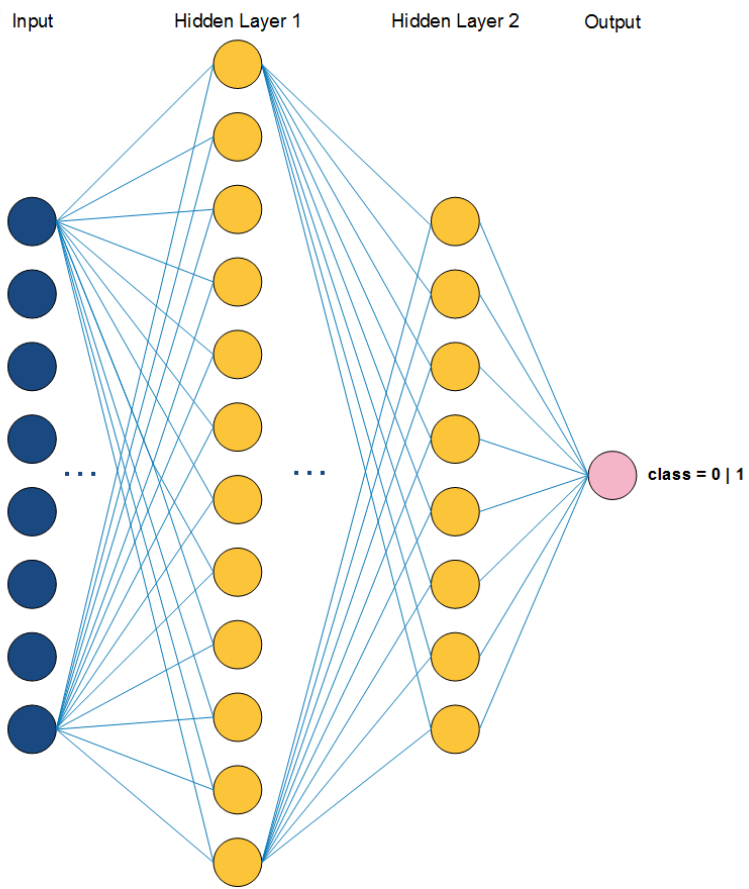
```
model.add(Dense(12, input_dim=8, init='uniform', activation='relu'))

model.add(Dense(8, activation='relu'))
```

Compile Model

```
model.compile(loss='binary_crossentropy',
              optimizer='adam', metrics=['accuracy'])
```

Optimizer – String identifier of an existing optimizer
loss function – This is the objective that the model will try to minimize
list of metrics – For any classification problem you will want to set this to metrics=['accuracy']



Batch vs Epoch.

Training occurs over epochs and each epoch is split into batches.

Epoch – One pass through all of the rows in the training dataset
Batch – One or more samples considered by the model within an epoch before weights are updated.

Keras Basic Syntax.

Save & Load Model.

Save

```
model_json = model.to_json()
with open("model.json", "w") as json_file:
    json_file.write(model_json)

model.save_weights("model.h5")
print("Saved model to disk")
```

Load

```
json_file = open('model.json', 'r')
loaded_model_json = json_file.read()
json_file.close()
loaded_model = model_from_json(loaded_model_json)
loaded_model.load_weights("model.h5")
```

Practical session

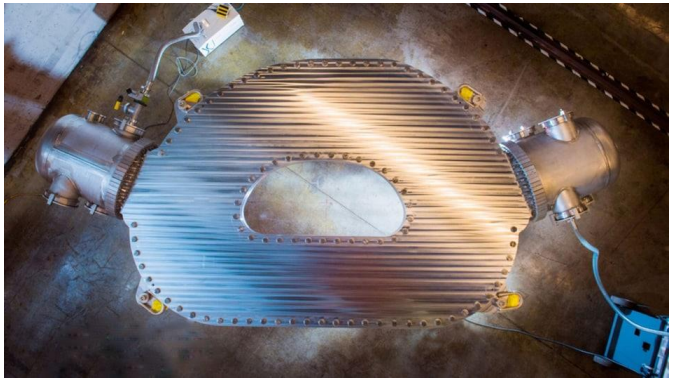
Designing your First Neural Network


Ganesh Chaturthi OFFER
3 COURSE AT ₹999
List of Master Class Bundle:
A.I Master Class
Artificial Intelligence + Data Analytics + Machine Learning
ECE Master Class-1
Embedded System + Matlab + FPGA
ECE Master Class-2
Embedded System + Arduino + PCB Designing
Other Bundle
Embedded System + Raspberry Pi + Arduino
6 MONTHS VALIDITY + 3 CERTIFICATE
BOOK NOW
OFFER VALID TILL
10 SEPTEMBER | 11:45 PM
+91 9003249787

Today's Short Bytes – Tech News

MIT Researchers Build Powerful Superconducting Magnet That Can Lead to Clean Fusion Energy

The new technology could fundamentally provide an "inexhaustible" source of energy.



Thanks!

Connect with me on **LinkedIn:**
link in Description

Product & Project:
www.pantechsolutions.net

Course:
Learn.pantechsolutions.net

Tomorrow session

Object recognition