AN INTRODUCTION TO COMPUTER VISION

Session by Arshika Lalan

“

Computer Vision is the subfield of artificial intelligence which tries to imitate the human vision capabilities i.e. the ability of humans to make sense of what they see

A RIDICULOUSLY BRIEF HISTORY

- Computer vision emerged in the late 60’s and developed almost parallely with the AI field.
- The first low-level tasks, such as color segmentation or edge detection, were already applied in the early days of the field and formed the foundation of many modern computer vision applications.
- However, by the 80’s, computer reasoning was still far from achieved

MORAVEC'S PARADOX

- Formulated by the computer scientist Hans Moravec
- You can make Deep Blue beat Kasparov in chess
- You cannot easily give a computer the capabilities of a toddler to recognise their parents, to find their favourite toy in the room, to walk without bumping (too much) onto walls.

CHALLENGES

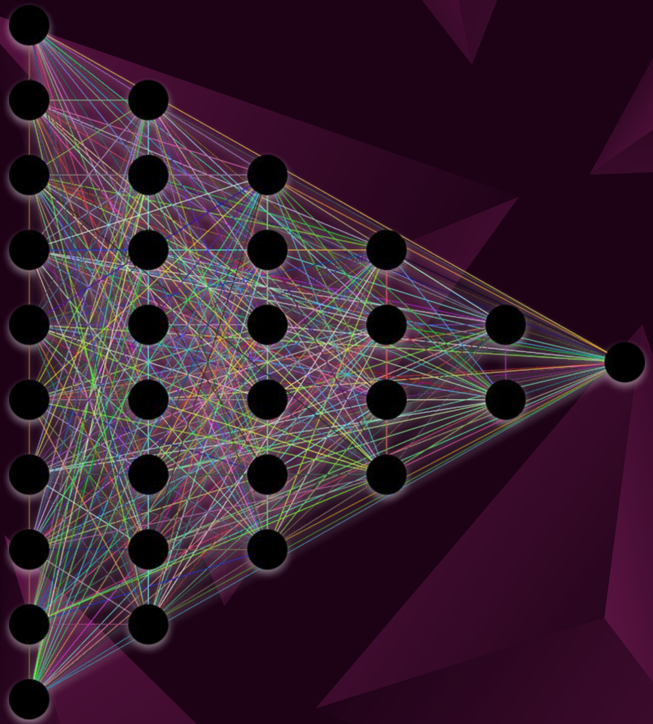
- Computers cannot extract higher-level information from images
- Issue of data representation.

COMPUTER VISION APPLICATIONS

- Object Classification
- Object Identification
- Object Verification
- Object Detection
- Object Landmark Detection
- Object Segmentation
- Object Recognition

THE BACKBONE OF CV: CNN

- Feedforward neural networks are fully connected networks
- Prone to overfitting and huge training time due to large number of parameters
- Can we have DNNs which are complex but have fewer parameters?



THE CONVOLUTION OPERATION

Consider pizza waiting times at a busy Italian restaurant.

X0 at t0, X1 at t1, X2 at t2

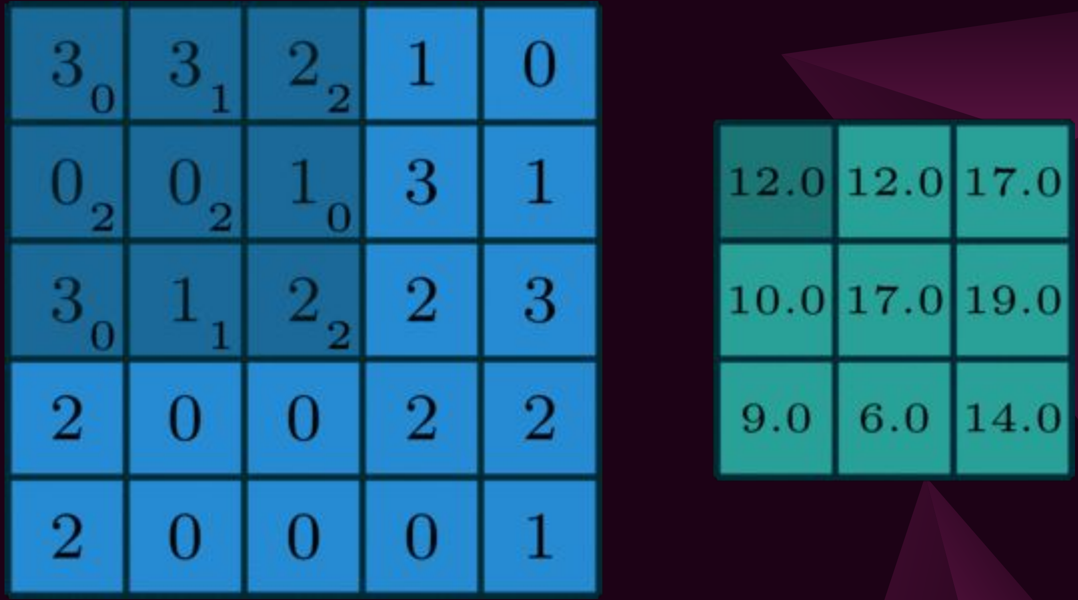
Average waiting time = W0*X0 + W1*X1 + W2*X2

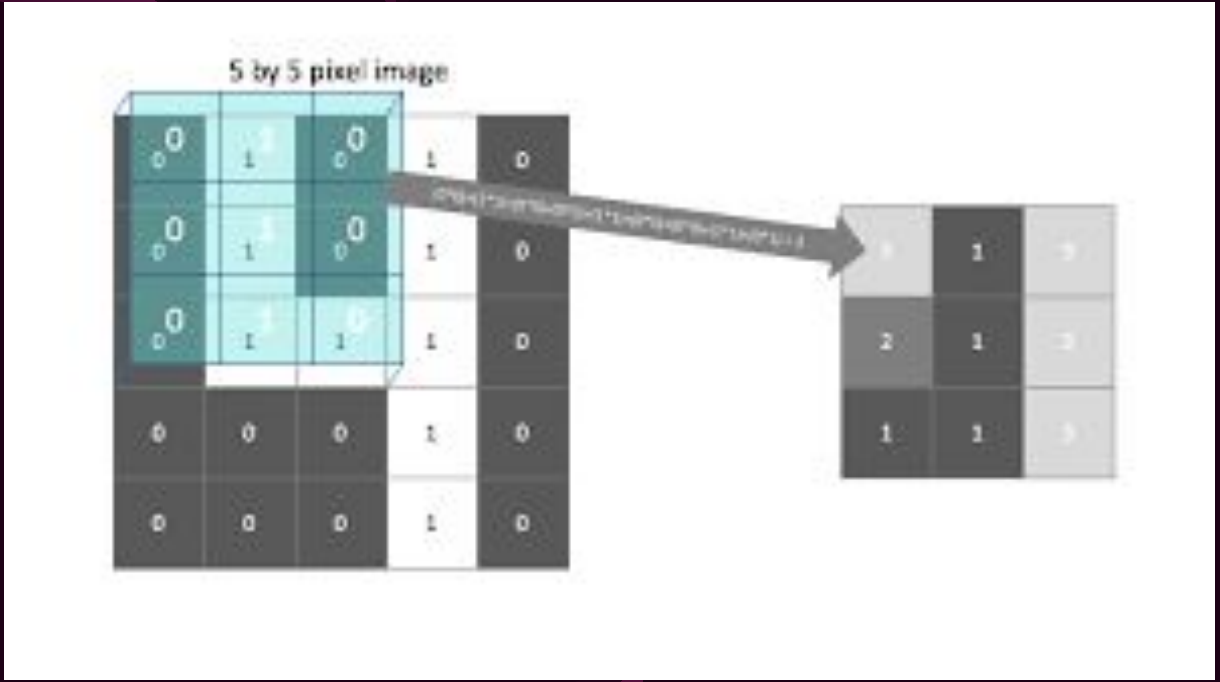
Convolution: Calculating the weighted average of all the previous neighbours to estimate the value of the current neighbour

1D CONVOLUTION

W	0.01	0.01	0.02	0.02	0.04	0.04	0.05
X	1.00	1.10	1.20	1.40	1.70	1.80	1.90
							1.80

2D CONVOLUTION





2D convolution

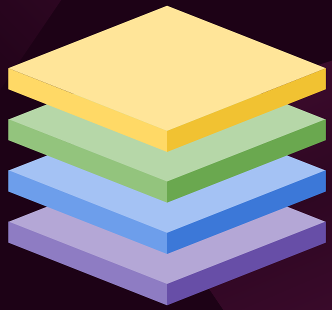
Input 30 x 30	conv	Kernel 3x3	Output 30x30 (blur)
	*	$\begin{vmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{vmatrix}$	

Input 30 x 30	conv	Kernel 3x3	Output 30x30 (Edge detection)
	*	$\begin{vmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{vmatrix}$	

Input 30 x 30	conv	Kernel 3x3	Output 30x30 (sharpens)
	*	$\begin{vmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{vmatrix}$	

CONVOLUTIONAL NEURAL NETWORKS

- Filters are analogous to the weights
- The entire convolutional output corresponds to the whole layer of neurons
- In FNNs, we consider all input values from the previous layer multiplied by their weights
- In CNNs, we consider only a small number of input values multiplied by the filter values
- Sparse connectivity and weight sharing
- Result? Huge reduction in the number of parameters and faster training time!!



- Padding
- Strided Convolutions
- Average Pooling and Max pooling
- ReLU, LeakyReLU, pReLU
- Occlusion
- Transfer Learning - AlexNet, VGGNet, ResNet

- RCNNs
- YOLO
- OpenCV

- UNet
- K-means Clustering
- Reinforcement Learning

REFERENCES

- Intuitively Understanding Convolutions for Deep Learning
- Stanford Spring 2020: Convolutional Neural Networks (CNNs / ConvNets)
- Feature Visualisation
- Convolutional Neural Networks (LeNet)
- Transfer Learning with Convolutional Neural Networks in PyTorch

THANKS!

Any questions?