WORLD OF AI

Sachin Shivakalimath
sachin.smath@gmail.com
9449764697

Introduction to World of AI

- **Deterministic AI** - AI systems that follow predefined rules or algorithms, producing the same output for a given input every time (e.g., traditional programming, rule-based systems).
- **Generative AI** - AI models that generate new content or data, such as text, images, music, or code, often by learning from large datasets (e.g., GPT-3, DALL·E). These models are typically based on deep learning and neural networks.
- **Futuristic AGI** - AGI refers to a still-hypothetical form of AI that would be capable of understanding, learning, and performing any intellectual task that a human being can do. It’s not just specialized, like current AI, but rather versatile and adaptable across domains.

Deterministic AI

- **Expert Systems** - These systems simulate human decision-making and provide solutions to complex problems using a knowledge base and a set of rules (e.g., medical diagnosis systems). They're largely deterministic and based on predefined knowledge from experts.
- **Rule Based Systems** - AI systems that follow a set of if-then-else rules to make decisions. These are straightforward systems that don't "learn" but execute predefined logic.
- **Machine Learning based systems-** A branch of AI that allows systems to learn and improve from experience (data) without being explicitly programmed. It's statistical and probabilistic rather than purely rule-based.

Expert System Based

- Expert systems are designed to emulate the decision-making ability of a human expert in a specific domain. These systems are typically broken down into components or types based on their architecture and application:
    1. Knowledge-Based Systems
    2. Inference Engines (Forward and Backward Chaining)
    3. Heuristic Systems
    4. Uncertainty Handling (Fuzzy Logic, Probabilistic Reasoning)

1. Knowledge-based systems

- These systems rely heavily on a **knowledge base**, which consists of facts and rules derived from human experts. They make decisions or provide recommendations based on this knowledge.
- **Example:** Medical diagnosis systems, legal advisory systems.

2. Inference Engine

- The inference engine is the core part of an expert system that applies the rules from the knowledge base to the facts in order to deduce new information or reach a conclusion.
- Types of Inference:
  - **Forward Chaining:** Start with the known facts and apply rules to deduce new facts, progressing towards a goal.
  - **Backward Chaining:** Start with a goal or hypothesis and work backwards, checking if there’s enough evidence to prove or disprove the hypothesis.

3. Heuristic Systems:

- These expert systems use heuristic (rule-of-thumb) reasoning, meaning they employ approximate methods or rules to solve complex problems more efficiently, particularly when an exact solution is either unknown or computationally expensive.
- **Example:** Systems used for diagnosis, troubleshooting, or decision support that don’t always rely on strict logic but on general heuristics.

4. Uncertainty Handling (Fuzzy Logic, Probabilistic Reasoning)

- Many expert systems include mechanisms for dealing with uncertainty, especially in domains like medicine where not every rule or fact is perfectly known.
  - Techniques:
    - **Fuzzy Logic:** Handles reasoning in cases where truth values can range between completely true and completely false.
    - **Probabilistic Reasoning (Bayesian Networks):** Uses probabilities to handle uncertainty in the decision-making process.

Rule Based Systems

Rule-based systems are deterministic and follow a set of predefined rules (if-then-else logic). The divisions are based on how rules are structured and the types of problems they solve.

1. Production Systems
2. Decision Trees
3. Business Rule Management Systems (BRMS)
4. Constraint-Based Systems
5. Expert Rule-Based Systems

1. Production Systems

- These are the simplest form of rule-based systems, where rules are defined as **if-then** (production) rules. They apply these rules in sequence to reach a conclusion or perform actions.
- **Components:**
  - **Rule Base:** Contains a set of predefined rules.
  - **Working Memory:** Stores the current state or facts of the system.
  - **Inference Engine:** Applies the rules from the rule base to the facts in the working memory to derive new information or perform actions.
- **Example:** Automated decision-making systems, workflow automation tools.

2. Decision Trees

- Rule-based systems can also be represented as decision trees, where each node in the tree represents a decision or test, and the branches represent the possible outcomes or rules to follow based on the decision.
- **Example:** Customer service systems that guide users through a series of questions and answers to troubleshoot problems.

3. Business Rule Management Systems (BRMS)

- These are rule-based systems designed to manage and execute business rules. They allow businesses to separate business logic (rules) from the core application code, making it easier to modify and update rules without changing the code.
- **Example:** Systems for managing compliance rules, financial transaction processing, loan approval systems.

4. Constraint-Based Systems

- These systems solve problems by applying a set of constraints (rules) to find solutions. They are often used in scheduling, resource allocation, or configuration problems.
- **Example:** Systems that assign employees to shifts or schedule tasks based on constraints like availability, skills, and workload.

5. Expert Rule-Based Systems

- These systems combine rule-based reasoning with domain-specific expertise. They may include large rule sets derived from human experts and typically handle more complex tasks than basic rule-based systems.
- **Example:** Systems used for credit scoring, fraud detection, or complex decision-making processes.

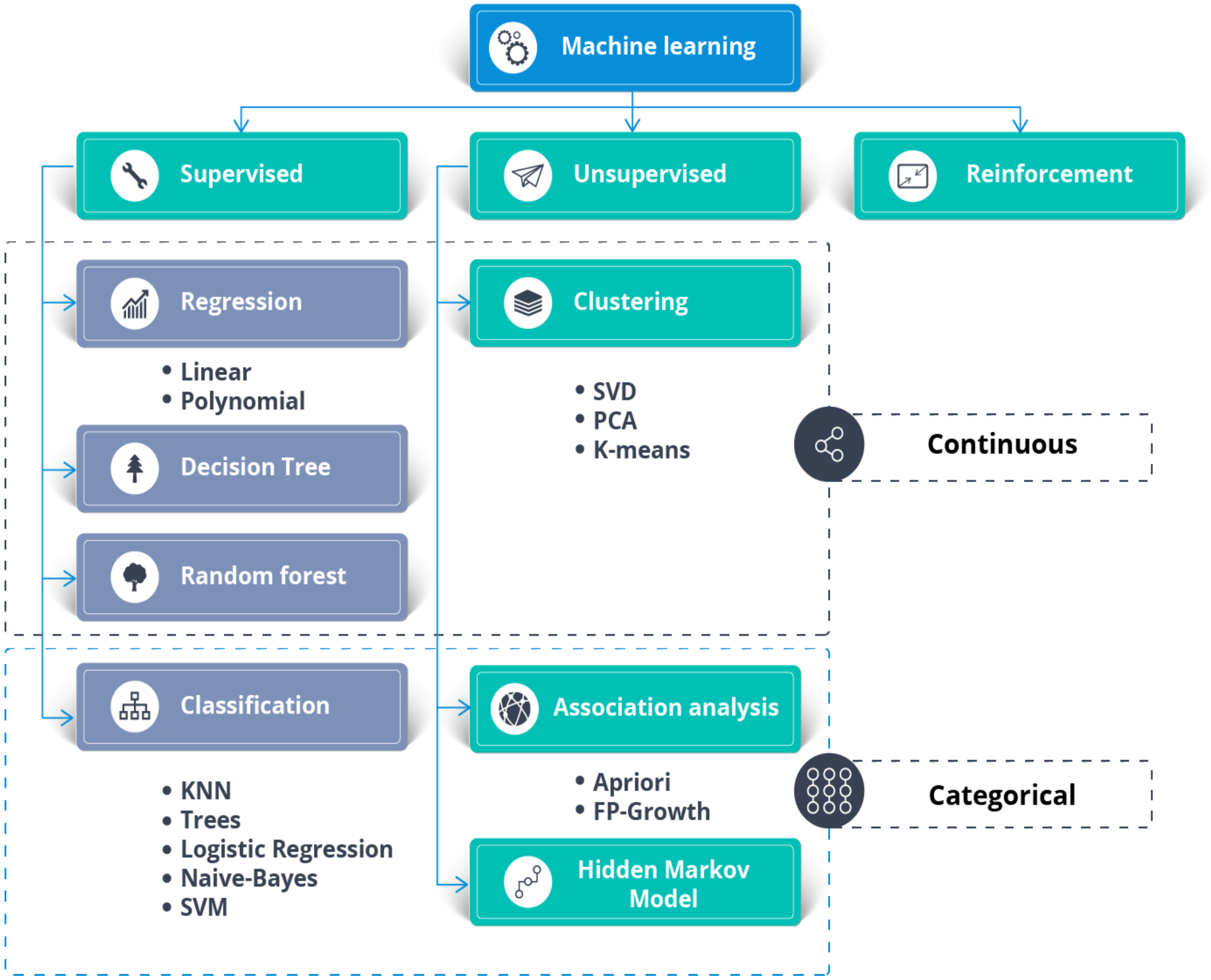
Machine Learning based systems

- Machine Learning (ML) is a subset of artificial intelligence (AI) that enables computers to learn from and make predictions or decisions based on data. Rather than being explicitly programmed to perform a task, a machine learning model uses algorithms to find patterns within data and improve its performance over time as it is exposed to more data.

Key Aspects of Machine Learning

- **Algorithms**: ML relies on various algorithms, such as decision trees, neural networks, and support vector machines, to identify patterns or relationships within data.
- **Training and Testing**: ML models are trained on historical data to learn patterns, then tested on new data to validate their accuracy.
- **Types of Learning**:
    - **Supervised Learning**: The model learns from labeled data, meaning each input has an associated correct output (e.g., image classification).
    - **Unsupervised Learning**: The model works with unlabeled data, identifying patterns or groupings on its own (e.g., clustering).
    - **Reinforcement Learning**: The model learns by trial and error, receiving feedback from its actions and adjusting based on rewards or penalties (e.g., game playing).
- **Applications**: ML is widely used in fields such as healthcare, finance, e-commerce, and many more, powering applications like recommendation systems, fraud detection, image recognition, and natural language processing.

Branches of Machine Learning



- Supervised Machine learning
- Unsupervised Machine learning
- Reinforcement Learning

Supervised ML

Supervised learning, also known as supervised machine learning, is a subcategory of machine learning and artificial intelligence. It is defined by its use of labeled datasets to train algorithms that to classify data or predict outcomes accurately

- Is like learning with a teacher
- Training dataset is like a teacher
- Training dataset is used to train the machine

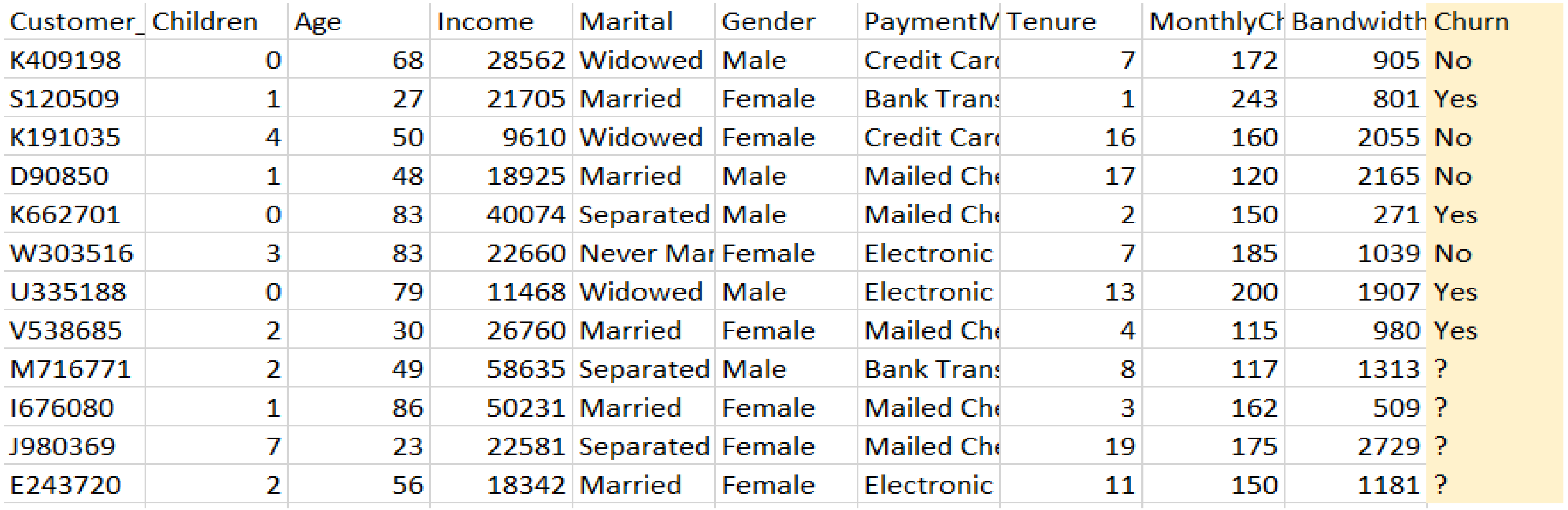
Example:
Classification: Machine is trained to classify something into some class.

- Classifying whether a patient has disease or not
- Classifying whether an email is spam or not
- You get a bunch of photos with information about what is on them and then you train a model to recognize new photos.

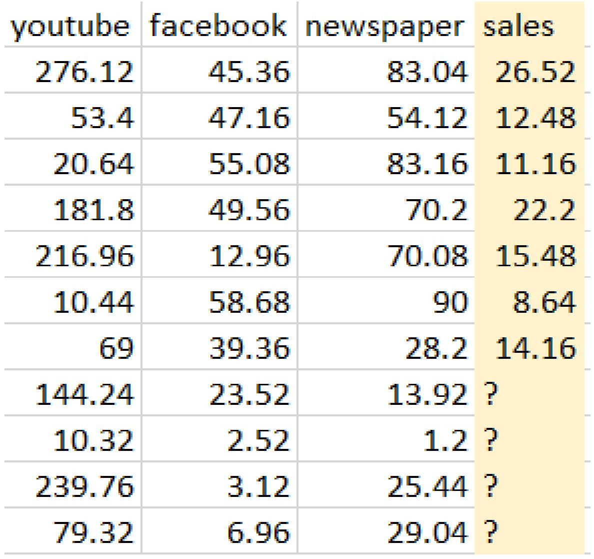
Regression: Machine is trained to predict some value like price, weight or height.

- Predicting house/property price
- Predicting stock market price

Supervised ML



Customer_	Children	Age	Income	Marital	Gender	PaymentM	Tenure	MonthlyCh	Bandwidth	Churn
K409198	0	68	28562	Widowed	Male	Credit Car	7	172	905	No
S120509	1	27	21705	Married	Female	Bank Trans	1	243	801	Yes
K191035	4	50	9610	Widowed	Female	Credit Car	16	160	2055	No
D90850	1	48	18925	Married	Male	Mailed Che	17	120	2165	No
K662701	0	83	40074	Separated	Male	Mailed Che	2	150	271	Yes
W303516	3	83	22660	Never Mar	Female	Electronic	7	185	1039	No
U335188	0	79	11468	Widowed	Male	Electronic	13	200	1907	Yes
V538685	2	30	26760	Married	Female	Mailed Che	4	115	980	Yes
M716771	2	49	58635	Separated	Male	Bank Trans	8	117	1313	?
I676080	1	86	50231	Married	Female	Mailed Che	3	162	509	?
J980369	7	23	22581	Separated	Female	Mailed Che	19	175	2729	?
E243720	2	56	18342	Married	Female	Electronic	11	150	1181	?



youtube	facebook	newspaper	sales
276.12	45.36	83.04	26.52
53.4	47.16	54.12	12.48
20.64	55.08	83.16	11.16
181.8	49.56	70.2	22.2
216.96	12.96	70.08	15.48
10.44	58.68	90	8.64
69	39.36	28.2	14.16
144.24	23.52	13.92	?
10.32	2.52	1.2	?
239.76	3.12	25.44	?
79.32	6.96	29.04	?

Un-Supervised ML

As the name suggests, unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data.

Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to **find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format**.

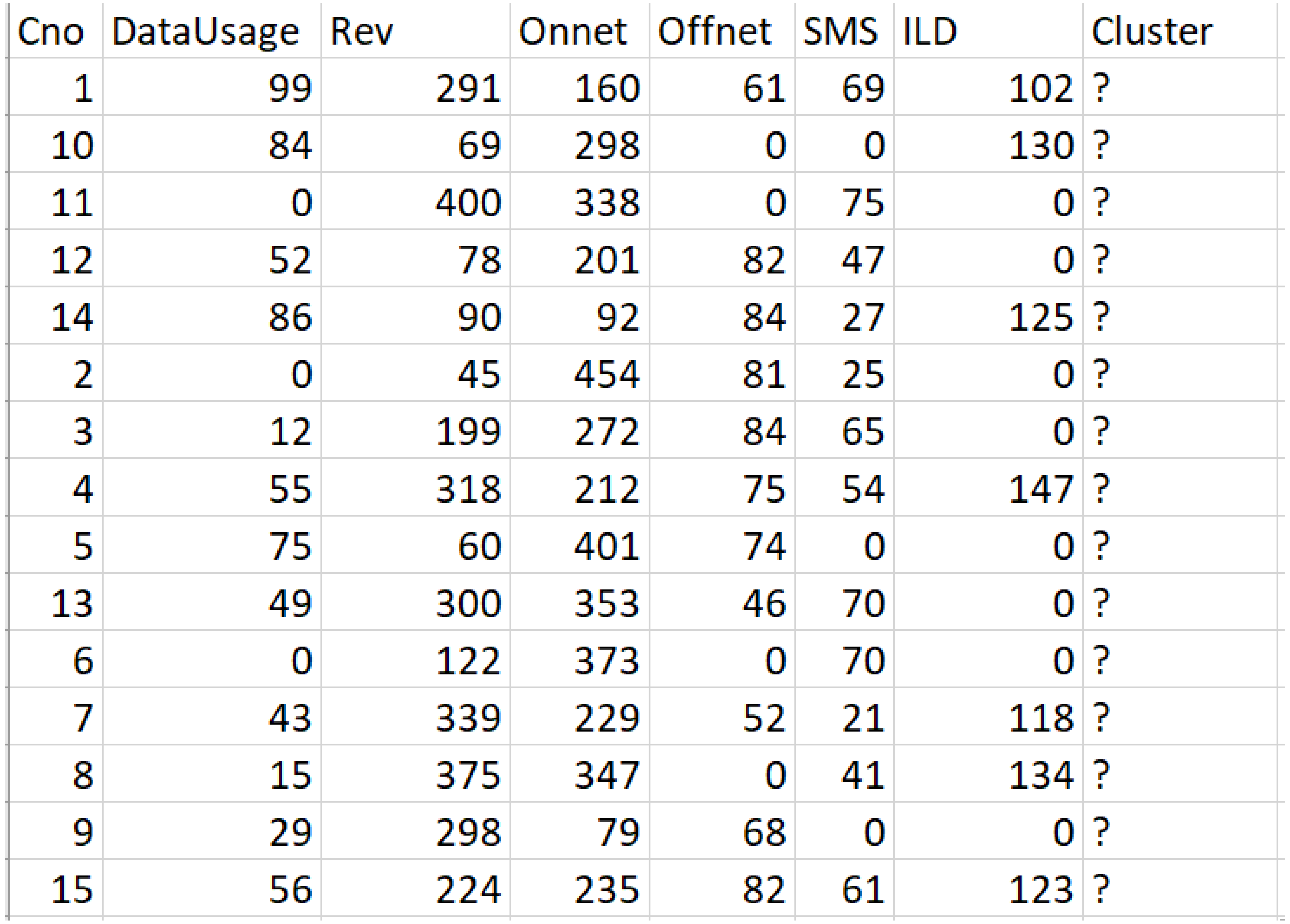
Example:
Clustering: A clustering problem is where you want to discover the inherent groupings in the data

- Clustering observed earthquake epicenters to identify dangerous zones; such as grouping customers by purchasing behavior
- You have a bunch of photos of 6 people but **without information about who is on which one** and you want to **divide** this dataset into 6 piles, each with the photos of one individual.

Association: An association rule learning problem is where you want to discover rules that describe large portions of your data

- Such as people that buy X also tend to buy Y

Un-Supervised ML



Cno	DataUsage	Rev	Onnet	Offnet	SMS	ILD	Cluster
1	99	291	160	61	69	102	?
10	84	69	298	0	0	130	?
11	0	400	338	0	75	0	?
12	52	78	201	82	47	0	?
14	86	90	92	84	27	125	?
2	0	45	454	81	25	0	?
3	12	199	272	84	65	0	?
4	55	318	212	75	54	147	?
5	75	60	401	74	0	0	?
13	49	300	353	46	70	0	?
6	0	122	373	0	70	0	?
7	43	339	229	52	21	118	?
8	15	375	347	0	41	134	?
9	29	298	79	68	0	0	?
15	56	224	235	82	61	123	?

Reinforcement ML

Reinforcement learning (RL) is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward.

Reinforcement learning is the training of machine learning models to make a sequence of decisions

Example:

- **Autonomous driving**
- **Personalizing your Netflix recommendations -** Reinforcement learning can help by introducing exploration which lets the model learn about new interests over time.
- **Optimizing inventory levels for Retail Store** - The algorithm ingests data including sales data, operating costs, number and type of merchandise, and the dynamic time frame for when the merchandise must be sold by.
- **Trading on the financial markets -** The advantage of reinforcement learning in this setting is the ability to learn to make predictions that account for whatever effects the algorithm’s actions have had on the state of the market. This feedback loop allows the algorithm to auto-tune over time, continually making it more powerful and adaptable. The reward function is based on the profit or loss made in each trade.
- **Robotics –** Robots perform various behaviours such as changing walking heights, fast walking, walking sideways and turning in the real world. It was also robust to changes in the robot itself (e.g. partially damaged motors) and the environment (e.g. changes in ground friction and being pushed from different directions)

ML Based systems heavily rely on Data.

Need of Data for ML Systems

- Data is fundamental to Machine Learning (ML) systems because it enables these systems to learn patterns, make predictions, and continuously improve over time.
- Data are required to :-
  - Training the ML Model
  - Evaluating and Validating Performance
  - Improving Accuracy and Robustness
  - Reducing Bias and Ensuring Fairness
  - Enabling Continuous Learning and Adaptation
  - Facilitating Feature Engineering
  - Supporting Real-World Applications

Define Data and its categories

- Data refers to raw facts, figures, and information that are collected, recorded, and stored for processing and analysis.
- It can take various forms, including numbers, text, images, audio, and video, and is often unorganized and unprocessed, lacking meaningful context until analyzed or processed.
- Data serves as the foundation for generating insights, knowledge, and informed decision-making in virtually every field, from science and technology to business and healthcare.

Categories of Data by type of data

- ·Numeric Data
  - Discrete (Integer)
  - Continuous (Float/Ratio Scale)
- Categorical Data
  - Nominal
  - Ordinal
- Date/Time Data
- Boolean Data
- Text Data (Unstructured)
- Image Data (Unstructured)
- Speech/Audio Data (Unstructured)
- Video Data (Unstructured)

Categories of Data by nature of data

- Structured data
    - Data that is highly organized and easily searchable in relational databases or spreadsheets. It typically follows a predefined model or schema, such as rows and columns.
    - Data in relational databases (e.g., SQL databases).
    - Spreadsheets (e.g., Excel files).
    - Data with a well-defined format like CSV.
- Unstructured data
    - Data that lacks a predefined format or organization, making it more challenging to process and analyse. It doesn't fit neatly into relational tables.
    - Text data (emails, documents, social media posts).
    - Images, videos, and audio files.
    - Logs, sensor data, and other types of raw, unformatted data.
- Semi-Structured Data
    - Data that doesn't follow a strict structure but has some organization or metadata that makes it easier to process than unstructured data. It might not fit perfectly into relational tables but still has tags or markers.
    - XML or JSON files.
    - NoSQL databases (e.g., MongoDB).
    - HTML and other mark-up language data.

Focusing on Generative AI

Generative AI

- *Definition:*
  - Generative AI refers to AI systems that create new content or data by learning patterns from large datasets. This content can range from text, images, and audio to videos or even synthetic data.
- *Foundation Models in Generative AI:*
  - **Uni-Modal Models:** Focus on generating one type of content, such as text or images.
  - **Multi-Modal Models:** Capable of handling and generating multiple types of content (e.g., text-to-image models).

Contd…

Generative AI

Types of Foundation Models:

- **Language Models (Text-Based):**
  - **Key Models:** GPT, BERT, T5, LLaMA.
  - **Applications:** Text generation, summarization, question-answering, chatbots.
  - **Example:** GPT-3 generates human-like text.
- **Image-Based Models:**
  - **Key Models:** Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), DALL·E.
  - **Applications:** Image synthesis, style transfer, deepfake generation.
  - **Example:** DALL·E generates realistic images from textual descriptions.
- **Multimodal Models:**
  - **Key Models:** CLIP (Contrastive Language-Image Pretraining), Flamingo, BLIP.
  - **Applications:** Cross-domain generation, such as text-to-image (e.g., generating an image from a text prompt).
  - **Example:** CLIP aligns text and images, allowing models like DALL·E to generate images from text.

Why NLP is Important for Generative AI

- **NLP (Natural Language Processing)** is crucial because:
  - **Foundation for Language Models:** Language models (e.g., GPT, BERT) rely heavily on NLP techniques to generate coherent and meaningful text.
  - **Multi-Domain Relevance:** Language models often form the backbone of multi-modal systems that combine text with other media (e.g., text-to-image).
  - **Understanding Prompts:** A key use case in Generative AI (especially multi-modal systems) is understanding and generating from natural language prompts (e.g., text-to-image generation).
- **NLP Use Cases :**
  - **Text Generation**
  - **Summarization**
  - **Topic Modeling**
  - **Machine Translation**
  - **Chatbot/Conversational AI**
  - **Text Completion/Autocompletion**
  - **Poetry/Story Generation**
  - **Audio/Text-to-Speech Generation**

Introduction to Computer Vision

- Computer Vision is a field of artificial intelligence (AI) focused on enabling computers to interpret and make decisions based on visual data, such as images and videos.
- It aims to replicate the human visual system, allowing machines to “see” and understand the world in a meaningful way.
- This technology powers applications ranging from facial recognition and object detection to medical imaging and autonomous driving.
- As a prerequisite for learning **Generative AI** (GenAI), understanding Computer Vision is essential because GenAI models, such as DALL-E and Stable Diffusion, often generate or manipulate visual content. A foundation in Computer Vision helps in understanding how these models process, generate, and evaluate images.

Key Concepts in Computer Vision

- **Image Representation**:
  - Images are represented as grids of pixels, with each pixel having a color value (for grayscale or RGB). For computers, images are essentially large arrays of numerical data.
  - Understanding image data structure and pixel manipulation forms the basis for many Computer Vision tasks.
- **Basic Image Processing**:
  - Image processing techniques, like filtering, edge detection, and thresholding, are essential for enhancing or extracting features from images.
  - Common libraries for image processing include **OpenCV** and **PIL** in Python, which offer various functions to manipulate images.
- **Feature Detection and Extraction**:
  - Features are key points or patterns in an image that represent significant areas or objects, such as edges, textures, or shapes.
  - Techniques like **SIFT** (Scale-Invariant Feature Transform), **HOG** (Histogram of Oriented Gradients), and **ORB** (Oriented FAST and Rotated BRIEF) help in detecting and extracting these features.

Contd…

Key Concepts in Computer Vision

- **Object Detection and Classification**:
  - Object detection involves locating and identifying objects within an image. Popular algorithms include **YOLO** (You Only Look Once) and **Faster R-CNN**.
  - Image classification involves categorizing entire images into predefined classes, using models like **Convolutional Neural Networks (CNNs)**, **ResNet**, and **VGG**.
- **Image Segmentation**:
  - Image segmentation divides an image into different segments or regions, often corresponding to objects or areas of interest.
  - There are two primary types of segmentation: **Semantic Segmentation** (assigns a class label to each pixel) and **Instance Segmentation** (differentiates separate objects of the same class).

Contd…

Key Concepts in Computer Vision

- **Convolutional Neural Networks (CNNs)**:
  - CNNs are specialized neural networks for processing visual data, designed to capture spatial hierarchies in images.
  - Key components of CNNs include **convolutions**, **pooling layers**, and **fully connected layers**.
  - CNNs are the backbone of many Computer Vision tasks, enabling models to detect and recognize complex patterns.
- **Generative Models in Computer Vision**:
  - Generative models, such as **Generative Adversarial Networks (GANs)** and **Variational Autoencoders (VAEs)**, are central to Generative AI, allowing models to generate new images from learned patterns.
  - GANs consist of a **generator** and a **discriminator** that work together to produce realistic images, often used for image synthesis, super-resolution, and style transfer.

Contd…

CV Use cases

- **Face Recognition**: Recognizing faces in security and social media applications**.**
- **Autonomous Vehicles:** Detecting objects, lanes, and traffic signs to navigate roads.
- **Healthcare:** Analyzing medical images for diagnostics, such as detecting tumors in X-rays or MRIs.
- **Retail:** Visual search and inventory tracking using cameras and image recognition.
- **Agriculture:** Monitoring crop health and detecting pests through aerial imagery and machine vision.
- **Image Generation**
- **Image Style Transfer**
- **Super-Resolution (Enhancing Image Quality)**
- **Deepfake Creation**
- **Inpainting (Image Restoration)**
- **3D Model Generation from Images**
- **Video Synthesis**

Why Computer Vision is Essential for Generative AI

•**Understanding Image Data**: Computer Vision techniques help in processing and analyzing image data, which is the basis for training and evaluating generative models in GenAI.
•**Enhancing Realism**: Knowledge of image quality, resolution, and feature extraction is essential for generating high-quality, realistic visuals in generative models.
•**Evaluating Generated Images**: Skills in object detection and classification help in assessing the accuracy and quality of generated images, essential for refining GenAI models.
•**Combining Vision with Language**: Generative AI models often combine visual and textual understanding (e.g., text-to-image models like DALL-E), making Computer Vision knowledge necessary to manage these multimodal inputs effectively.

Getting Started with Computer Vision

- To start learning Computer Vision, consider familiarizing yourself with:
    - Basic image manipulation and processing using **OpenCV** or **Pillow**.
    - Learning **Convolutional Neural Networks (CNNs)** for image classification tasks.
    - Implementing feature extraction techniques and exploring datasets like **ImageNet** and **CIFAR-10**.
    - Experimenting with **GANs** and **VAEs** to understand image generation.

Overall, understanding Computer Vision will provide a solid foundation for exploring Generative AI, especially for creating, manipulating, and interpreting visual content.

Introduction to Deep Learning

- Deep Learning is a subset of machine learning that focuses on algorithms inspired by the structure and function of the brain, known as **artificial neural networks**.
- It enables computers to learn complex patterns and representations in large datasets, achieving breakthroughs in tasks that require a high level of abstraction, such as image and speech recognition, natural language processing, and generative modeling.
- Deep learning has been instrumental in the rapid advancement of artificial intelligence (AI), as it can process unstructured data like images, audio, and text more effectively than traditional machine learning algorithms.
- It is particularly relevant for **Generative AI** (GenAI) applications, where models generate new content such as text, images, or audio.

List of algorithms (Definitely not a comprehensive list)

- **Artificial Neural Networks (ANN)**
  - Feedforward Neural Network (FNN)
  - Multi-Layer Perceptron (MLP)
  - Radial Basis Function Network (RBF)
  - Extreme Learning Machine (ELM)
  - Self-Organizing Map (SOM)
  - Hopfield Network

- **Convolutional Neural Networks (CNN)**
  - LeNet
  - AlexNet
  - VGGNet
  - GoogLeNet (Inception)
  - ResNet
  - DenseNet
  - MobileNet
  - EfficientNet
  - SqueezeNet
  - Xception
  - ShuffleNet

- **Recurrent Neural Networks (RNN)**
  - Vanilla RNN
  - Long Short-Term Memory (LSTM)
  - Gated Recurrent Unit (GRU)
  - Bidirectional RNN
  - Bidirectional LSTM
  - Sequence-to-Sequence (Seq2Seq)
  - Encoder-Decoder RNN
  - Attention Mechanism
  - Transformers

Contd…

List of algorithms (Definitely not a comprehensive list)

- **Variational Autoencoders (VAE)**
  - Vanilla VAE
  - Conditional VAE (CVAE)
  - Disentangled VAE
  - β-VAE (Beta VAE)
  - Adversarial VAE (AVAE)
  - Hierarchical VAE

- **Generative Adversarial Networks (GAN)**
  - Vanilla GAN
  - Deep Convolutional GAN (DCGAN)
  - Conditional GAN (cGAN)
  - Wasserstein GAN (WGAN)
  - Wasserstein GAN with Gradient Penalty (WGAN-GP)
  - Least Squares GAN (LSGAN)
  - InfoGAN
  - CycleGAN
  - StyleGAN
  - BigGAN
  - Progressive Growing GAN
  - StarGAN
  - SAGAN (Self-Attention GAN)
  - Pix2Pix GAN
  - AttnGAN
  - ProGAN (Progressive GAN)

- **Transformers**
  - Vanilla Transformer
  - BERT (Bidirectional Encoder Representations from Transformers)
  - GPT (Generative Pre-trained Transformer)
  - T5 (Text-To-Text Transfer Transformer)
  - BART (Bidirectional and Auto-Regressive Transformers)
  - XLNet
  - RoBERTa
  - DistilBERT
  - ALBERT
  - Vision Transformer (ViT)

Contd…

List of algorithms (Definitely not a comprehensive list)

- **Autoencoders**
  - Vanilla Autoencoder
  - Denoising Autoencoder
  - Sparse Autoencoder
  - Contractive Autoencoder
  - Undercomplete Autoencoder
  - Stacked Autoencoder
  - Convolutional Autoencoder
- **Spiking Neural Networks (SNN)**
  - Leaky Integrate-and-Fire Model
  - Hodgkin-Huxley Model
  - Izhikevich Model
  - Spike-Timing Dependent Plasticity (STDP)
- **Graph Neural Networks (GNN)**
  - Graph Convolutional Network (GCN)
  - Graph Attention Network (GAT)
  - Graph Isomorphism Network (GIN)
  - GraphSAGE
  - Relational Graph Convolutional Network (R-GCN)
  - Spatial-Temporal Graph Neural Network (ST-GNN)
  - ChebNet

Contd…

List of algorithms (Definitely not a comprehensive list)

- **Self-Supervised Learning (SSL) Models**
  - Contrastive Predictive Coding (CPC)
  - SimCLR (Simple Framework for Contrastive Learning)
  - BYOL (Bootstrap Your Own Latent)
  - MoCo (Momentum Contrast)
  - SwAV (Swapping Assignments between Views)
  - SimSiam (Simple Siamese Network)

- **Memory-Augmented Neural Networks (MANN)**
  - Neural Turing Machine (NTM)
  - Differentiable Neural Computer (DNC)
  - Memory Networks (MemNets)
  - Transformer-XL
  - Recurrent Memory Transformer

- **Attention Mechanisms and Variants**
  - Scaled Dot-Product Attention
  - Multi-Head Attention
  - Self-Attention
  - Cross-Attention
  - Relative Position Encoding
  - Luong Attention
  - Bahdanau Attention (Additive Attention)

Contd…

List of algorithms (Definitely not a comprehensive list)

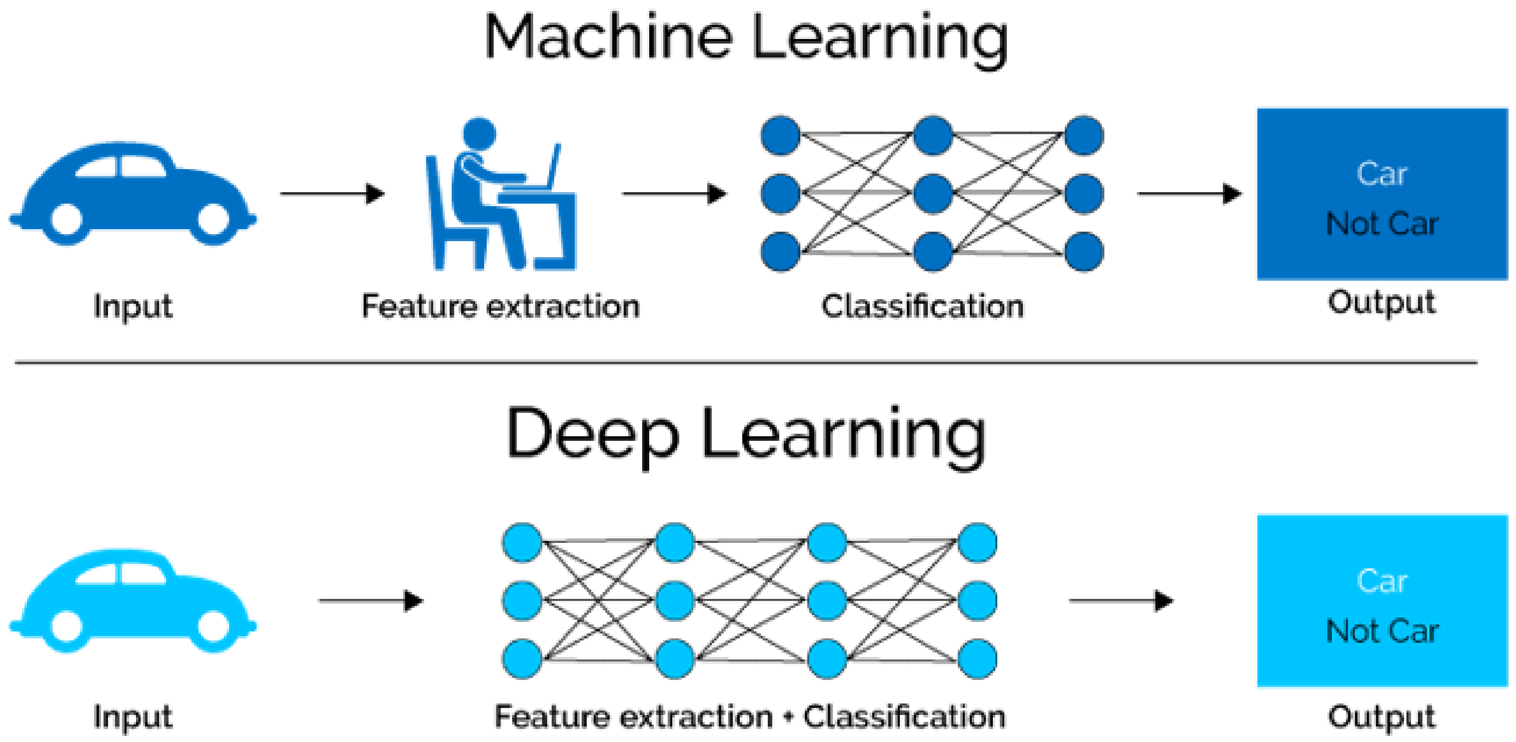
- **Energy-Based Models (EBM)**
  - Boltzmann Machine
  - Restricted Boltzmann Machine (RBM)
  - Deep Belief Network (DBN)
  - Hopfield Network
  - Contrastive Divergence Algorithm
- **Capsule Networks**
  - Dynamic Routing Capsule Networks
  - Matrix Capsules
  - Inverted Dot-Product Attention Capsule Networks
- **Reinforcement Learning Architectures**
  - Deep Q-Network (DQN)
  - Deep Deterministic Policy Gradient (DDPG)
  - Proximal Policy Optimization (PPO)
  - Soft Actor-Critic (SAC)
  - Twin Delayed DDPG (TD3)
  - Trust Region Policy Optimization (TRPO)
  - Advantage Actor-Critic (A2C)
  - Asynchronous Advantage Actor-Critic (A3C)

List of algorithms (Definitely not a comprehensive list)

- **Recurrent Variants for Long-Sequence Processing**
  - Transformer-XL
  - Reformer
  - Longformer
  - Linformer
  - BigBird
  - Performer

- **Advanced Generative Models**
  - Diffusion Models
  - Autoregressive Models (PixelCNN, PixelRNN)
  - Flow-Based Models (RealNVP, Glow)
  - Normalizing Flows (Invertible Neural Networks)

ML VS DL

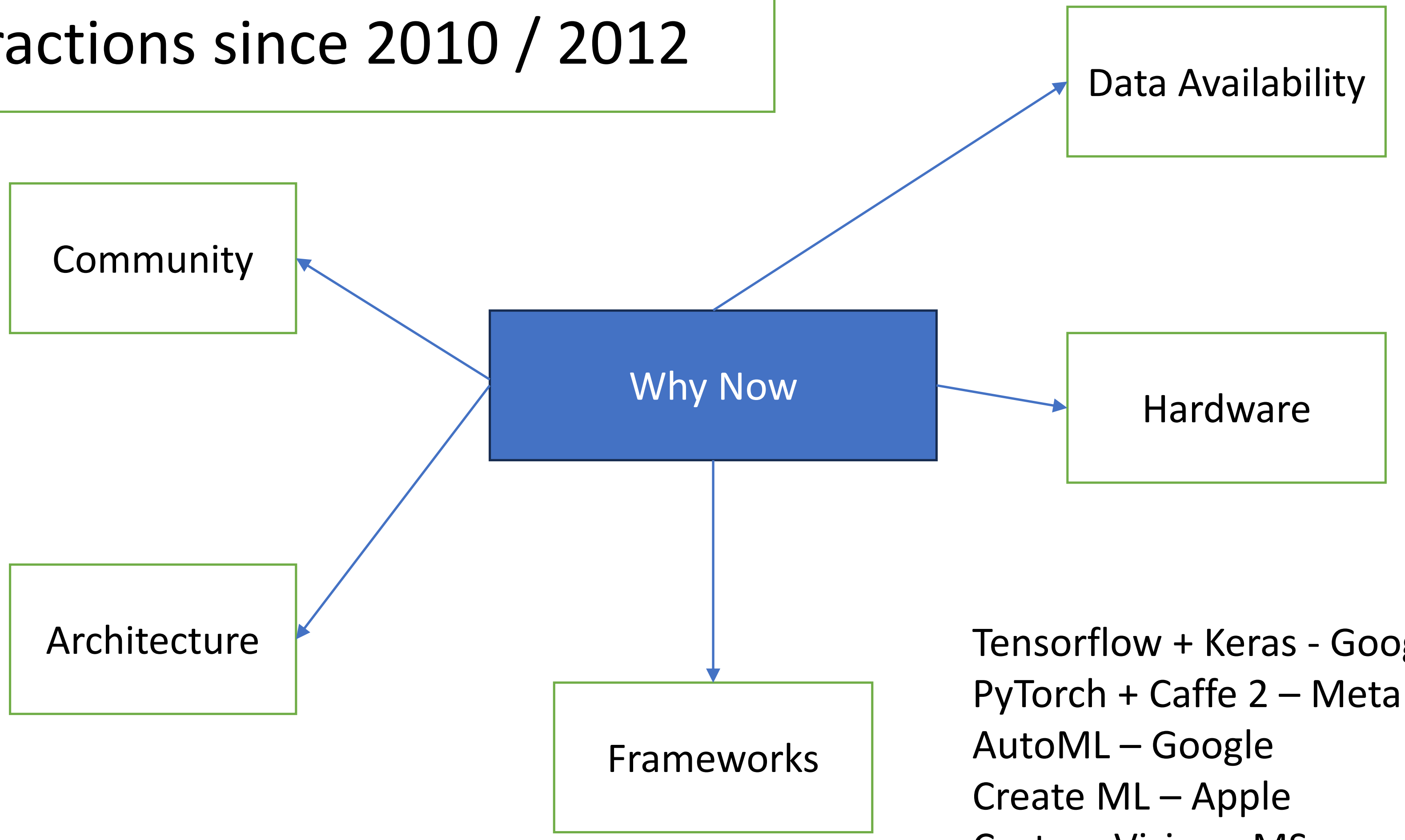

Machine Learning
Input
Feature extraction
Classification
Car
Not Car
Output
Deep Learning
Input
Feature extraction + Classification
Car
Not Car
Output

History of Neural Network

- Chapter 1 – After world war 2 (1960s)
  - Frank Rosenblatt – Perceptron ('the embryo of an electronic computer that expects will be able to walk, talk, see, write, reproduce itself, and be conscious of its existence')
- Chapter -2 – The First AI Winter (1969)
  - Marvin Minsky – Perceptron cannot learn a XOR(Nonlinear) function
- Chapter -3 – The Rise (1978-86)
  - Geoffrey Hinton- Learning representations using Backpropagating errors. (Universal Approximation Theorem)
  - 1989 - Yann LeCun (Student of Geoffrey Hinton, Father of CNN)
- Chapter -3 – The Second AI Winter (Early 1990s)
  - Data availability
  - Computing Power
  - Weight initialization technique was at random
  - SVM, Random Forest overtook DL
- Chapter -4 The Rise again (2006-2010)
  - Deep believe network by Geoffrey Hinton- The initial weights were initialized by another NN called as Unsupervised Pre-training
  - Because of this we could add more hidden layers and our network became deep. From there Neural Networks were called as Deep Learning.
- Chapter -5 The Final Rise (2012)
  - Geoffrey Hinton again in Imagenet Competition – DL on GPU, reduced the error to 16% from 28% (from state-of-the-art ML)
- Chapter -6 The Silver linings (2012 +)
  - Google Deep mind
  - GAN- by Ian Goodfellow

Why DL is Hot now?

- Started in 1960
- Gained tractions since 2010 / 2012



Wikipedia 1M
MS COCO
Youtube 8M
Google Audio ds

Strong Community
Kaggle
Researchers
Passionate People

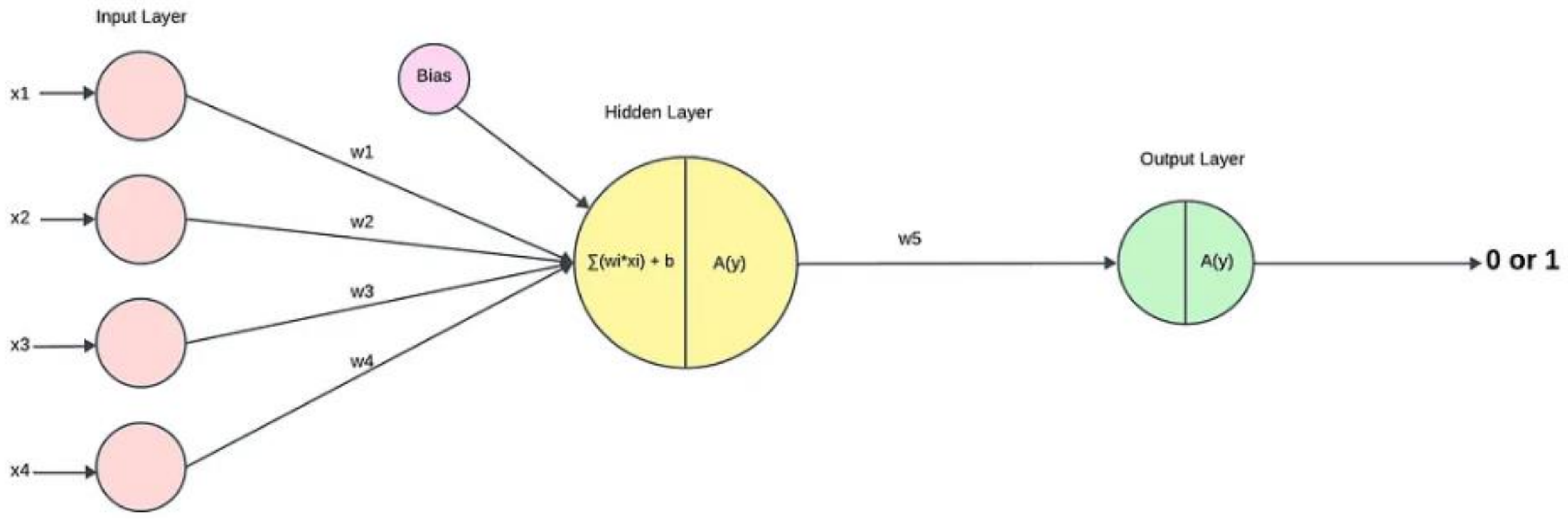
Moore's Law
Parallel Processing – NVIDIA CUDA
ASICs – TPU (Google), EDGE TPU , NPUs

Transfer Learning – Readymade Architectures (RESNET, BERT, UNET, PIX2PIX, YOLO, WaveNET)

Tensorflow + Keras - Google
PyTorch + Caffe 2 – Meta
AutoML – Google
Create ML – Apple
Custom Vision - MS

The Building block - Neuron

A **neuron** in a neural network is a fundamental building block that processes information and helps the network learn to recognize patterns. It is inspired by biological neurons in the human brain and works by receiving input data, applying mathematical transformations, and passing the result to other neurons in the network.



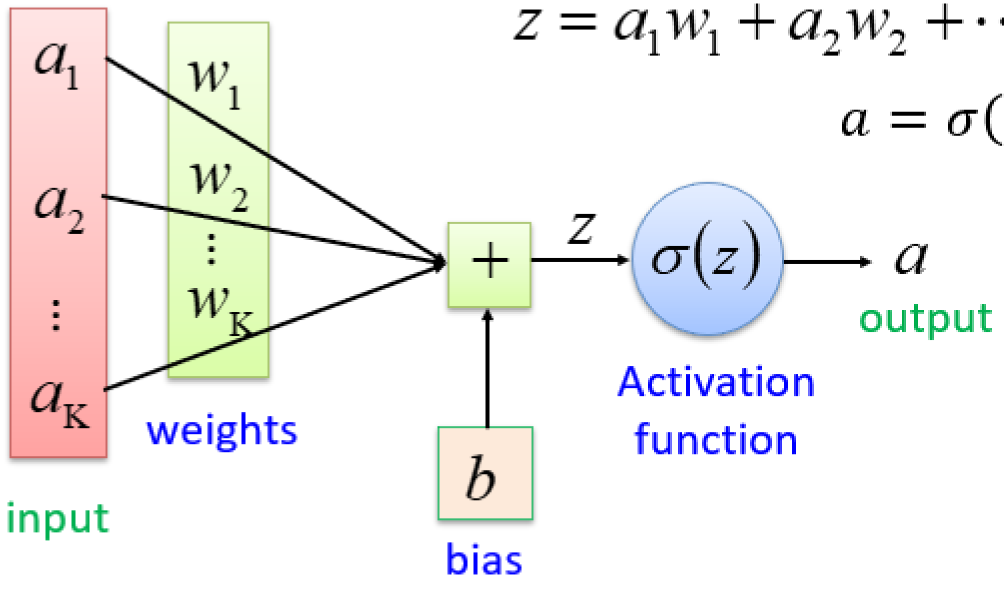
*single neuron is also called as perceptron

The computational unit of NN - Neuron

- NNs consist of hidden layers with neurons (i.e., computational units)
- A single neuron maps a set of inputs into an output number, or $f: R^K \rightarrow R$

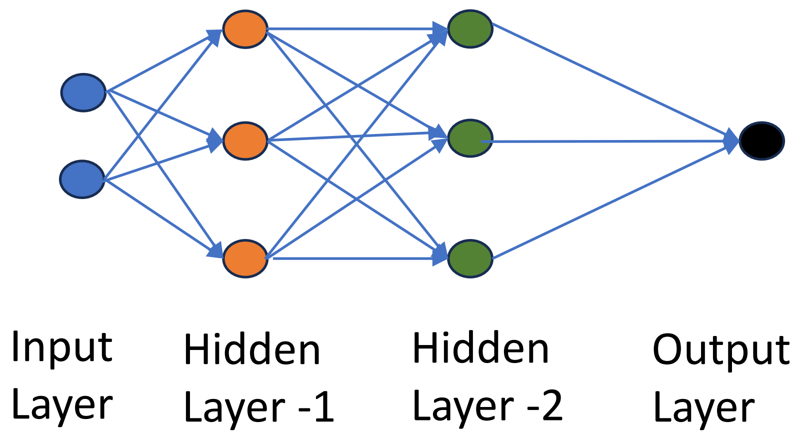
$$z = a_1 w_1 + a_2 w_2 + \cdots + a_K w_K + b$$

$$a = \sigma(z)$$



Neural Network

- A **neural network** is a collection of interconnected neurons organized in layers, allowing it to learn complex patterns in data. Neural networks typically consist of:
- **Input Layer**: Receives the raw data features (e.g., pixels in an image, words in a sentence) and feeds them to the network.
- **Hidden Layers**: One or more layers of neurons that transform inputs through weighted connections, biases, and activation functions. These layers learn intermediate representations and patterns in the data.
- **Output Layer**: Produces the final result, like a classification label or predicted value, based on the learned patterns.
- During training, the network adjusts the weights and biases in each neuron to minimize errors, enabling it to make accurate predictions or decisions on new data.



Artificial Neural Network Example

- Buckle up – Its Hands On

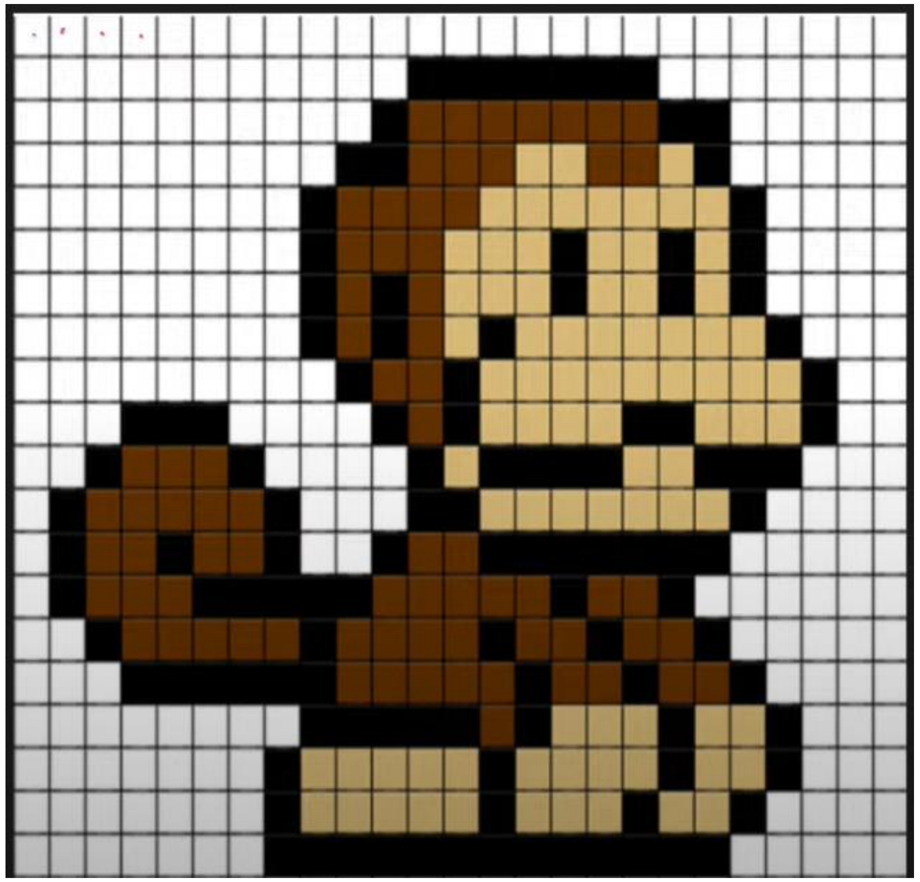
CNN

CNN Introduction

- 1998- Yann Lecun in AT&T Labs
- CNN is a class of NN which takes an input known as grid like topology.
- 1D Time Series and 2D Images qualify for this kind on input.
- Grid like topology in 1D TS – data are been captured at regular interval of time. For 2D image all the pixel values are arranged in matrix.

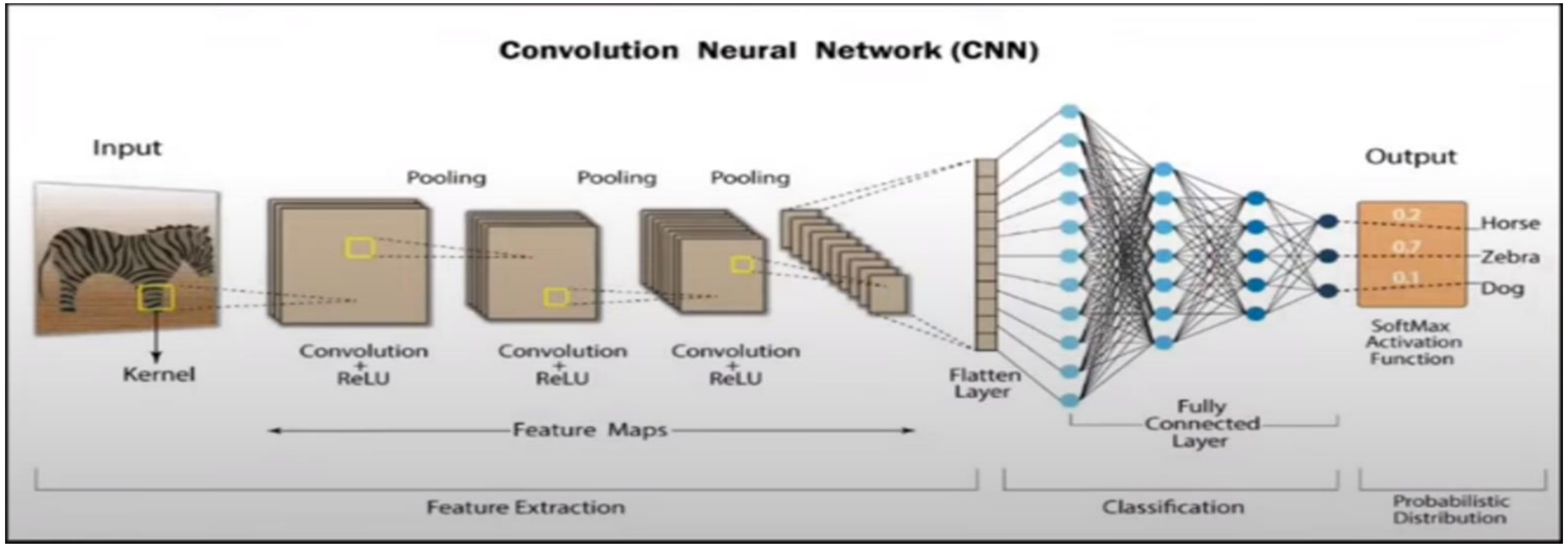
Why not ANN

- CNN > ANN When comes to image data
- Why not ANN
  - High Computational cost
  - Overfitting
  - Loss of important info like spatial arrangement of pixels

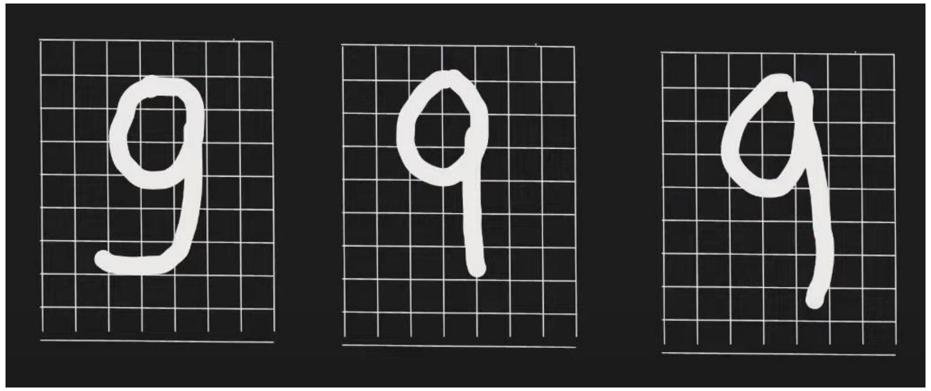


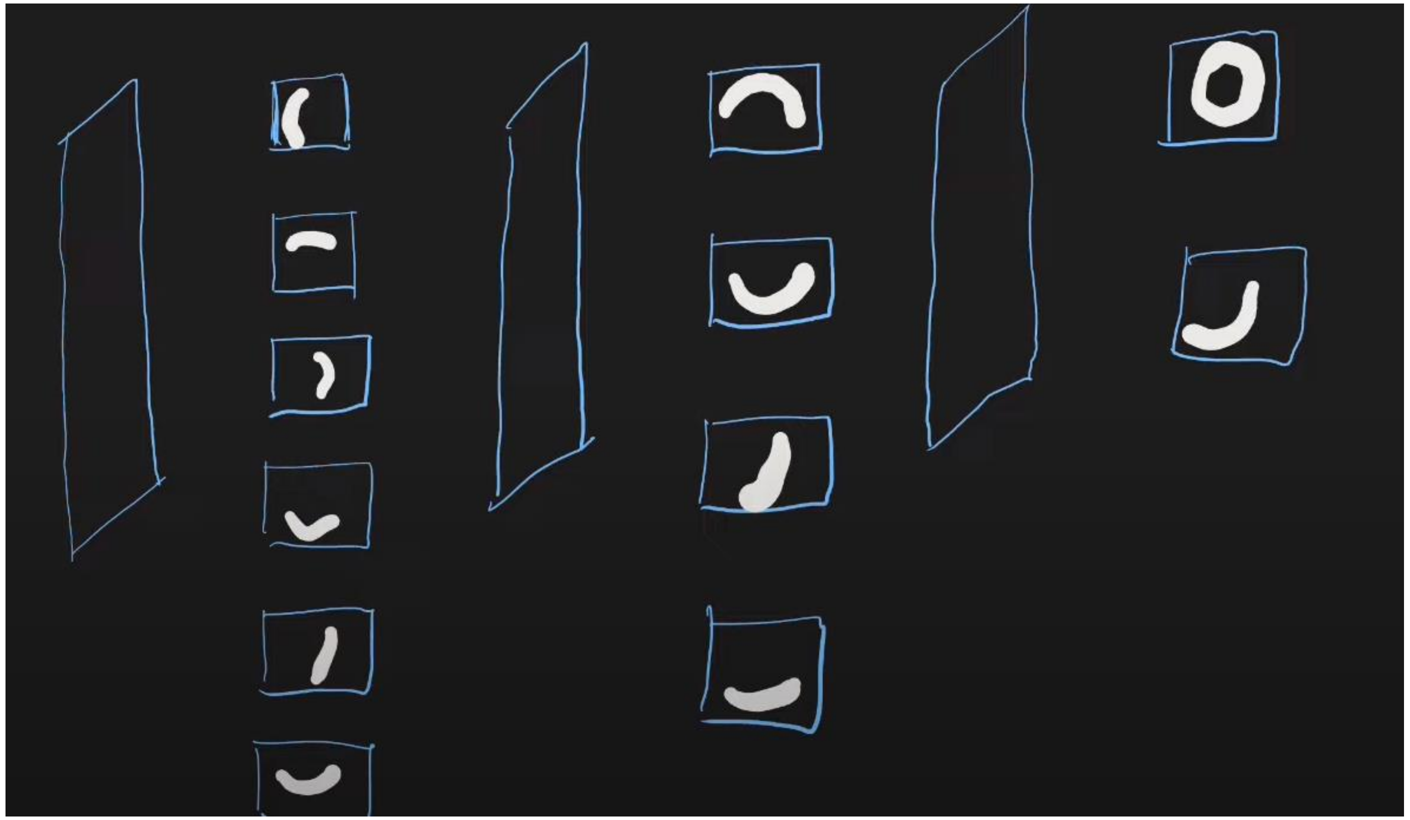
ANN VS CNN

- Introduction of Convolution layer capable of doing convolution operation and matrix operation.
- Convolution layer
- Pooling Layer
- Fully Connected Layer



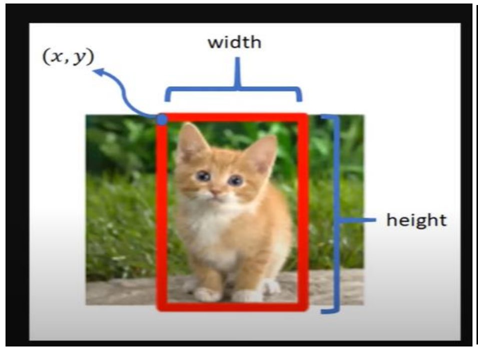
CNN Intuition





- Convolutions layers are filters they extract features

CNN Applications



Object localization

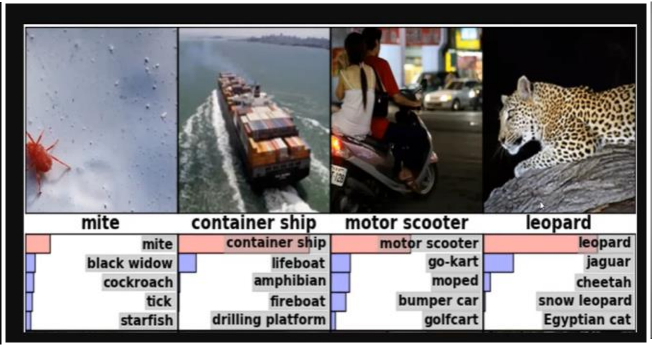
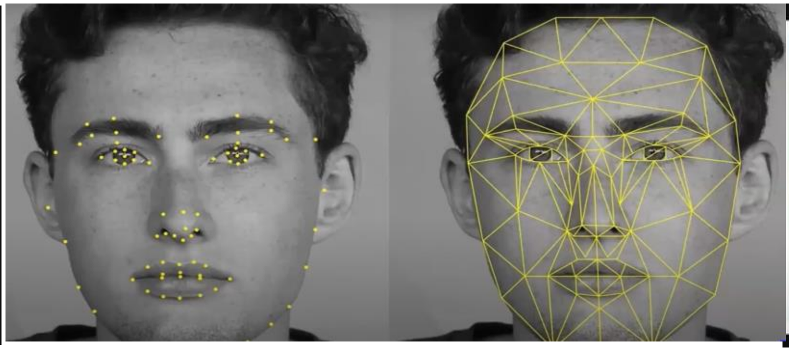


Image classification



Facial recognition

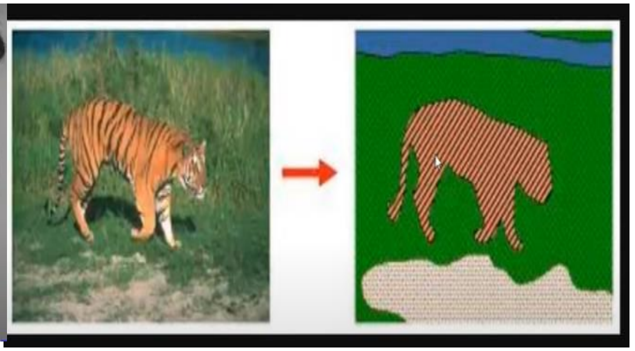
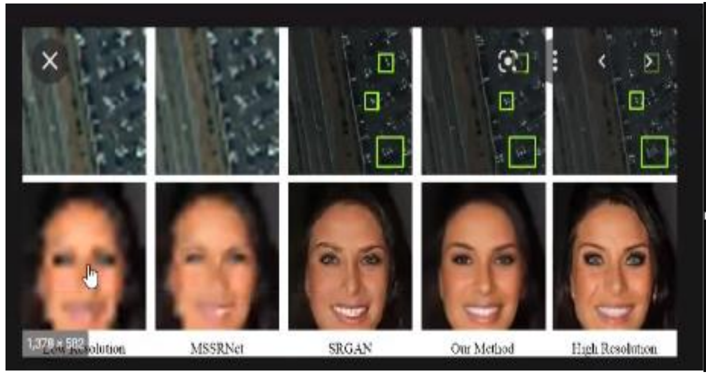
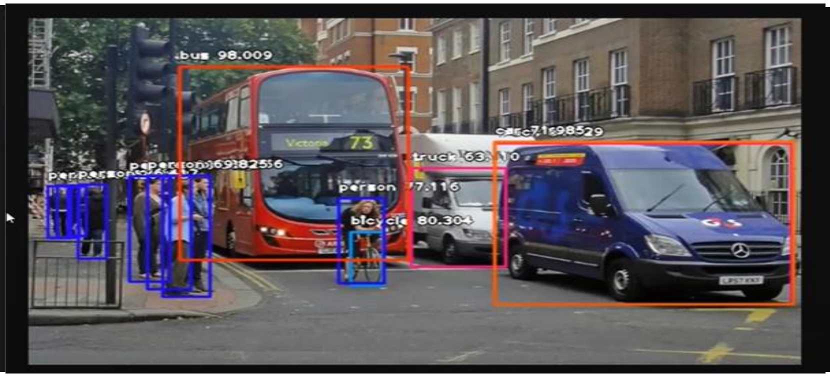


Image segmentation



Super resolution



Object detection

Posture detection

Image coloration

Convolution calculation

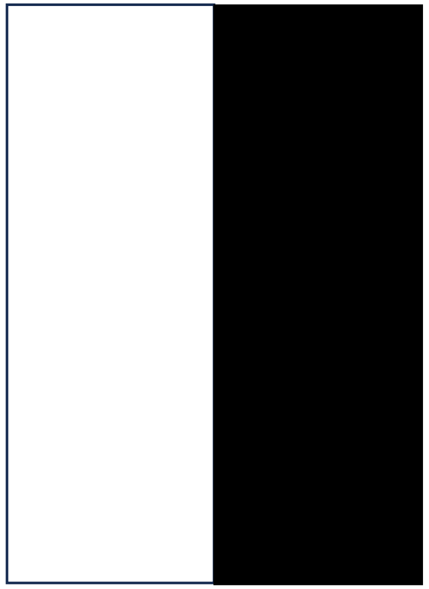
- Grey Scale image

180	210	225	235	245	235	230	220	210	190
200	210	225	235	240	245	240	230	220	210
195	220	230	235	240	240	235	230	220	205
190	185	205	220	230	235	230	225	215	200
180	175	195	210	220	225	220	215	205	190
170	165	185	200	210	215	210	205	195	180
160	155	175	190	200	205	200	195	185	170
150	145	165	180	190	195	190	185	175	160
140	135	155	170	180	185	180	175	165	150
140	135	155	170	180	185	180	175	165	150
140	135	155	170	180	185	180	175	165	150
140	135	155	170	180	185	180	175	165	150

Normalized_value = (Original_value - Min_value) / (Max_value - Min_value)

0.43	0.70	0.83	0.91	1.00	0.91	0.87	0.78	0.70	0.52
0.61	0.70	0.83	0.91	0.96	1.00	0.96	0.87	0.78	0.70
0.57	0.78	0.87	0.91	0.96	0.96	0.91	0.87	0.78	0.65
0.52	0.48	0.65	0.78	0.87	0.91	0.87	0.83	0.74	0.61
0.43	0.39	0.57	0.70	0.78	0.83	0.78	0.74	0.65	0.52
0.35	0.30	0.48	0.61	0.70	0.74	0.70	0.65	0.57	0.43
0.26	0.22	0.39	0.52	0.61	0.65	0.61	0.57	0.48	0.35
0.17	0.13	0.30	0.43	0.52	0.57	0.52	0.48	0.39	0.26
0.09	0.04	0.22	0.35	0.43	0.48	0.43	0.39	0.30	0.17
0.09	0.04	0.22	0.35	0.43	0.48	0.43	0.39	0.30	0.17
0.09	0.04	0.22	0.35	0.43	0.48	0.43	0.39	0.30	0.17
0.09	0.04	0.22	0.35	0.43	0.48	0.43	0.39	0.30	0.17

Simple example -V1 Layer Vertical Edge detector



0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1

1	0	-1
2	0	-2
1	0	-1

Convolution

Apply Filter with stride jump of 1

0	**-4**	**-4**	**0**
0	**-4**	**-4**	**0**
0	**-4**	**-4**	**0**
0	**-4**	**-4**	**0**

Min Max Scaler

N-f+1

255	**0**	**0**	**255**
255	**0**	**0**	**255**
255	**0**	**0**	**255**
255	**0**	**0**	**255**

Feature map

V1 Layer Horizontal Edge detector

1	2	1
0	0	0
-1	-2	-1

Convolution

Layers of Convolution- General Guidelines

- **1st Layer:** it often detects edges, lines, and basic image patterns. Think of it as finding building blocks.
- **2nd Layer:** Combines information from the first layer to detect more complex shapes and textures. For example, it might identify corners, blobs, or specific patterns within edges.
- **3rd Layer:** Starts building higher-level features by combining outputs from the second layer. It might detect simple object components like eyes, noses, or wheels in certain contexts.
- **4th Layer (and beyond):** Further refine and combine features. Deeper layers become more abstract and specific to the task at hand. They might learn combinations of textures, shapes, and spatial relationships that uniquely identify objects, even with variations in size, position, or lighting.

Convolution calculation

- Coloured image

180	210	225	235	245	235	230	220	210	190
200									
195									
190									
180									
170									
160									
150									
140									
140									
140									
140									

180	210	225	235	245	235	230	220	210	190
200									
195									
190									
180									
170									
160									
150									
140									
140									
140									
140									

180	210	225	235	245	235	230	220	210	190
200	210	225	235	240	245	240	230	220	210
195	220	230	235	240	240	235	230	220	205
190	185	205	220	230	235	230	225	215	200
180	175	195	210	220	225	220	215	205	190
170	165	185	200	210	215	210	205	195	180
160	155	175	190	200	205	200	195	185	170
150	145	165	180	190	195	190	185	175	160
140	135	155	170	180	185	180	175	165	150
140	135	155	170	180	185	180	175	165	150
140	135	155	170	180	185	180	175	165	150
140	135	155	170	180	185	180	175	165	150

0.43	0.70	0.83	0.91	1.00	0.91	0.87	0.78	0.70	0.52
0.61									
0.57									
0.52									
0.43									
0.35									
0.26									
0.17									
0.09									
0.09									
0.09									
0.09									

0.43	0.70	0.83	0.91	1.00	0.91	0.87	0.78	0.70	0.52
0.61									
0.57									
0.52									
0.43									
0.35									
0.26									
0.17									
0.09									
0.09									
0.09									
0.09									

0.43	0.70	0.83	0.91	1.00	0.91	0.87	0.78	0.70	0.52
0.61	0.70	0.83	0.91	0.96	1.00	0.96	0.87	0.78	0.70
0.57	0.78	0.87	0.91	0.96	0.96	0.91	0.87	0.78	0.65
0.52	0.48	0.65	0.78	0.87	0.91	0.87	0.83	0.74	0.61
0.43	0.39	0.57	0.70	0.78	0.83	0.78	0.74	0.65	0.52
0.35	0.30	0.48	0.61	0.70	0.74	0.70	0.65	0.57	0.43
0.26	0.22	0.39	0.52	0.61	0.65	0.61	0.57	0.48	0.35
0.17	0.13	0.30	0.43	0.52	0.57	0.52	0.48	0.39	0.26
0.09	0.04	0.22	0.35	0.43	0.48	0.43	0.39	0.30	0.17
0.09	0.04	0.22	0.35	0.43	0.48	0.43	0.39	0.30	0.17
0.09	0.04	0.22	0.35	0.43	0.48	0.43	0.39	0.30	0.17
0.09	0.04	0.22	0.35	0.43	0.48	0.43	0.39	0.30	0.17

255	0	0	255
255			
255			
255			

255	0	0	255
255			
255			
255			

255	0	0	255
255	0	0	255
255	0	0	255
255	0	0	255

Padding

- Reduced number of Pixels
- Low importance on the edge pixels as each of them are a part of only one convolution operation whereas, the central pixels are a part of multiple convolution operation
- Keras – Valid and same (0 padding by default)

0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1

Zero Padding

0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	0
0	0	0	0	1	1	1	0
0	0	0	0	1	1	1	0
0	0	0	0	1	1	1	0
0	0	0	0	1	1	1	0
0	0	0	0	1	1	1	0
0	0	0	0	0	0	0	0

$N + 2P - f + 1$

0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1

Neighbour Padding

0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1

Strides

- Slicing the convolution on the actual image pixels.
- By default stride = 1. However, it can be changed
- With increased stride (strided convolution) information loss despite of padding.
- Stride = 2

$$\left[\frac{n+2p-f}{2}+1\right]$$

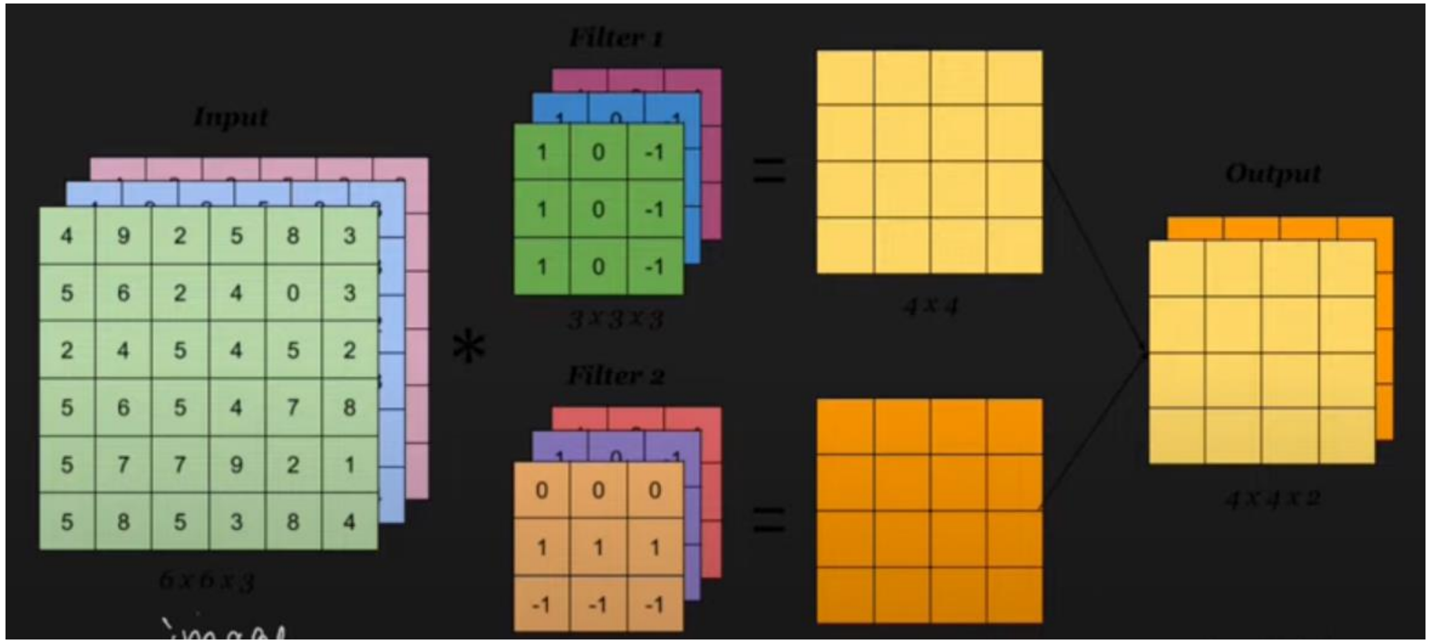
- When required: -
  - When required only high level features
  - Faster training time.
  - Lesser required compute

Types of Pooling

- Max Pooling
- Min Pooling
- Average Pooling
- L_2 Pooling
- Global Pooling

Pooling

- Two Problems
  - Memory Size
  - Translation Variance



Memory Size Problem

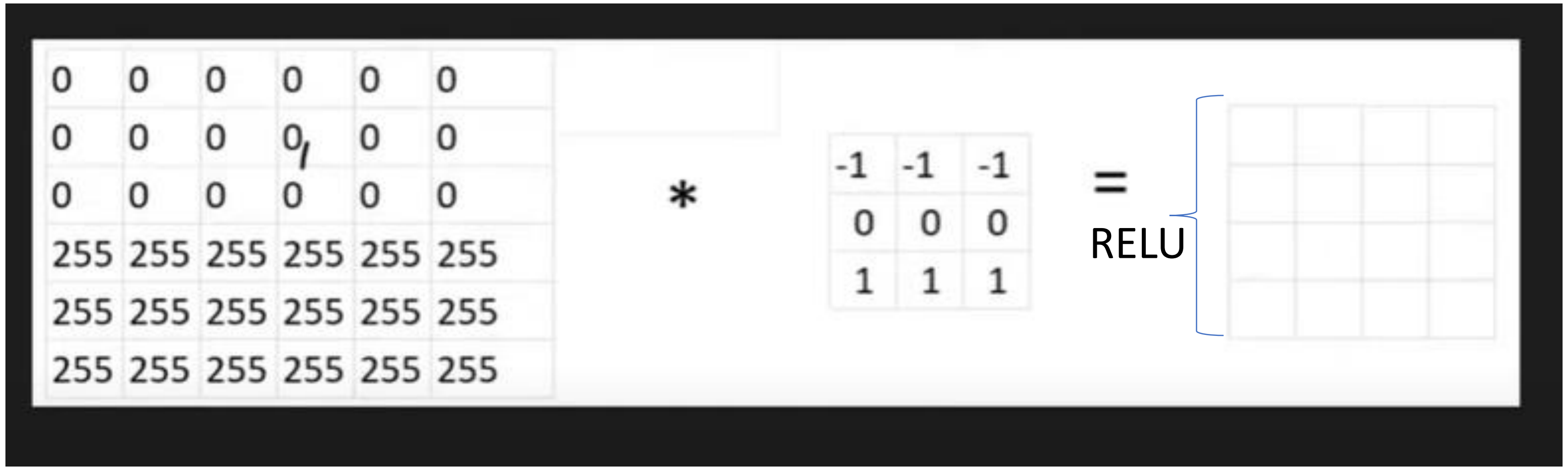
Image	Filters / Convolutions	Feature Map	Features
228 X 228 X 3	3 X 3	100	226 X 226 X 100 X 32(bit) = 19MB

100 images to train we need 1.5 GB Memory

- Memory Size Problem

- Features are location dependent

Down sampling- Pooling



0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
255	255	255	255	255	255
255	255	255	255	255	255
255	255	255	255	255	255

*

-1	-1	-1
0	0	0
1	1	1

=

RELU

Size = (2,2)
Stride = 2
Type = Max

Max Pooling

3	1	1	3
2	5	0	2
1	4	2	1
4	7	2	4

Size = (2,2)
Stride = 2
Type = Max

5	3
7	4

Feature Map

New Feature Map

- Reduces size to handle Memory issue
- Handles Translation Variance also by emphasizing the major details neglecting minor details
- Example - https://deeplizard.com/resource/pavq7noze3

Flattened Layer

- **Transforms data:** The flattening layer takes the multidimensional output of the convolutional/pooling layers (typically shaped like a 3D array for images) and converts it into a single, long 1D vector. This makes the data compatible with the fully connected layers, which require a flat input format.
- **Preserves information:** While flattening transforms the shape, it aims to retain the essential information extracted from the input data by the previous layers. This information, represented as activations across different channels and locations, becomes embedded in the flattened vector.
- **Intuition:** Imagine an image fed into a CNN. Convolutional layers extract features like edges, corners, and textures, creating feature maps. Pooling layers down sample these maps, further summarizing the extracted information. But they still retain a spatial structure. The flattening layer takes this spatial map and "unrolls" it, treating each element (activation) as a single piece of information regardless of its location. This single vector captures the essence of the extracted features, ready for the fully connected layers to analyze and make predictions.

Popular CNN Architectures and transfer learning

- **VGG family (**Visual Geometry Group (VGG))
  - VGG16
  - VGG19
- **ResNet**
- **Inception (Google)**
- **Xception (Google)**
- **DenseNet**