**Artificial Intelligence (AI) - Lecture Notes**

**1. What is Artificial Intelligence (AI)?**

* AI is technology that helps computers or systems perform tasks that normally need human intelligence.
* Examples:
  + Unlocking phones with Face ID (Computer Vision)
  + Siri or Alexa (Speech Recognition & NLP)
  + ChatGPT or Gemini (Large Language Models)
  + Amazon, Netflix, YouTube (Recommendation Systems)
  + Google Maps, Uber (Traffic Prediction, ETA estimation)
  + GitHub Copilot (AI-assisted Coding)

**2. AI in Daily Life**

We use AI every day in multiple ways; shopping, navigation, entertainment, communication, and even coding.

**3. AI Fundamentals**

AI is a broad field. It includes:

* **Machine Learning (ML)**
* **Deep Learning (DL)**
* **Generative AI (GenAI)**
* **Other areas** like rule-based systems, robotics, and fuzzy logic.

**4. Machine Learning (ML)**

* ML is a subset of AI.
* ML algorithms **learn from data** rather than being explicitly programmed.
* ML became popular with the rise of data from the internet.
* Example: Gmail spam detection, Credit card fraud detection, Delivery time prediction.

**Key Points**

* **All Machine Learning is AI**, but **not all AI is Machine Learning**.
* Rule-based systems and classical robotics are AI, but not ML.

**5. How Machine Learning Works**

ML has **two main steps**:

1. **Training**: Learning patterns from past data.
   * Example: A bank studies old loan data to see who got approved or rejected.
2. **Inference (Prediction)**: Using the trained model to make new predictions.
   * Example: Predicting whether a new applicant will get a loan or not.

**Formula**

* Model learns relationship: **Y = f(X)**
  + X = Input (data)
  + Y = Output (prediction)

**6. Machine Learning vs Traditional Programming**

| **Traditional Programming** | **Machine Learning** |
| --- | --- |
| Logic is written by humans | Logic is learned from data |
| Input → Logic → Output | Input + Output → Algorithm learns logic (Model) → Predict new Output |

**7. Types of Machine Learning**

There are **three main types**:

1. **Supervised Learning**
2. **Unsupervised Learning**
3. **Reinforcement Learning**



**8. Supervised Learning**

* Model learns from **labeled data** (data with input and correct output).
* Examples:
  + Spam or not spam (Email classification)
  + Loan approved or not approved

**Types of Supervised Learning Problems**

**(a) Classification**

* Output has **fixed categories**.
* Examples:
  + Spam / Not Spam
  + Loan: Yes / No
  + Cat / Dog image
* Types:
  + **Binary Classification**: Two possible outputs (Yes/No)
  + **Multi-class Classification**: More than two outputs (e.g. sentiment analysis: positive, negative, neutral)

**(b) Regression**

* Output is **numerical** (continuous value).
* Examples:
  + Predicting house price
  + Predicting delivery time
  + Predicting stock price
* Tries to find relationship between:
  + **Independent Variable (X)** – Input
  + **Dependent Variable (Y)** – Output
  + **Equation:** Y = f(X)

**9. Common Algorithms**

* **For Classification:**  
  Logistic Regression, KNN, SVM, Random Forest, XGBoost
* **For Regression:**  
  Linear Regression, Decision Trees, Random Forest Regressor, etc.

**Reinforcement Learning**

* The goal of reinforcement learning is to **maximize rewards**.
* Rewards are given when the model makes **correct predictions**.
* It does not make a single prediction, but a **set of predictions**.
* The agent interacts with the **environment** and takes **actions**.
* If the action is correct, it gets a **reward** (positive value).
* If the action is wrong, it gets a **penalty** (negative value).

**Example:**

* In a game like *Snake and Ladders*, if the player meets a snake → penalty.
* If the player meets a ladder → reward.
* The goal is to reach the endpoint (maximize rewards).

**Applications:**

* Used in **games** like Chess, Go.
* Used in **self-driving cars**.
* Used in **robotics** (for movement, walking, picking objects).

**Algorithms in Reinforcement Learning:**

* Q-Learning
* Deep Q Networks (DQN)
* Policy Gradient Methods
* Proximal Policy Optimization (PPO)

**Deep Learning**

* Deep learning is a **subset of machine learning**.
* It studies **neural networks**.
* Machine learning algorithms that work on tables are called **statistical ML**.
* Statistical ML works well on **structured data** (rows and columns).
* Deep learning works better on **unstructured data** like images, videos, audio, text, etc.

**Why deep learning for unstructured data?**

* In unstructured data, it is hard to define features manually.
* For example, in an image, it’s hard to tell the computer what an eye or mouth looks like.
* Deep learning models can **extract features automatically** from raw data using **neural networks**.

**Neural Networks (Basic Idea)**

* Inspired by the **human brain**.
* Contains many **neurons** connected in layers:
  + **Input Layer**: gets data
  + **Hidden Layers**: process data
  + **Output Layer**: gives results
* Real networks can have millions or billions of connections.

**Example (Marks Calculation):**

* Inputs: Midterm marks, End term marks, Class tests
* Each has a **weight** (importance)
* Output: Final CGPA
* If results are wrong, weights are adjusted.
* Neural networks do the same: adjust **weights** and **biases** to improve accuracy.

**Training Neural Networks**

Two main steps:

1. **Forward Propagation**
   * Input moves forward through the network.
   * Each neuron performs calculations:  
     Weighted sum → Activation function → Output
   * Produces a **prediction**.
2. **Backward Propagation**
   * The prediction is compared to the **expected result**.
   * Calculates **loss** (error) using a **loss function**.
   * Adjusts **weights and biases** to reduce errors.
   * This repeats for all data until the network becomes accurate.

**Activation Functions Examples:**

* Sigmoid: 1 / (1 + e^-x)
* ReLU: Rectified Linear Unit

**Training repeats** many times, improving accuracy each time.

**Tools for Neural Networks**

* **Programming Languages:** Python, R
* **Libraries:**
  + NumPy, Pandas: data processing
  + Seaborn, Matplotlib : data visualization
  + Scikit-learn, XGBoost: ML algorithms
* **Deep Learning Frameworks:**
  + **TensorFlow** (by Google)
  + **PyTorch** (by Meta): easier for beginners
* **Kaggle**: for datasets
* **GPU or Cloud Machines** for training large models

**Neural Network Architectures**

1. **Feedforward Neural Network (FNN)**
   * Data moves in one direction (no loops).
   * Used for simple predictions like medical diagnosis, loan approval.
2. **Recurrent Neural Network (RNN)**
   * Has memory, remembers previous information.
   * Good for **sequential data** (like text or time series).
   * Used for **language translation**, **speech recognition**, **stock prediction**.
   * Limitation: weak long-term memory.
3. **LSTM (Long Short-Term Memory)**
   * Advanced form of RNN.
   * Can remember long-term dependencies better.
4. **Convolutional Neural Network (CNN)**
   * Used for **image data**.
   * More complex architecture.
   * Will be explained in depth next.

**Convolutional Neural Networks (CNNs)**

**1. How Computers Read Images**

* Computers cannot see like humans.
* They read **images and videos as grids of pixels**.
* A black and white image can be shown as a grid of numbers (for example, 6x6).
* Each pixel has a value:
  + **1** for black (or where something is present)
  + **0** for white (or empty space)

**2. Example**

* A 6x6 image means **36 pixels**.
* So, the network needs **36 input values** to process one image.
* Real images are much larger (for example, 1000x1000 = 10⁶ pixels).
* Each neuron in the hidden layer must do that many calculations.
* This becomes **too heavy and slow** for normal neural networks.

**3. Colored Images**

* Color images use **three channels**:
  + Red (R)
  + Green (G)
  + Blue (B)
* So instead of one grid, we have **three grids** (R, G, and B).
* This means **3 × 10⁶ inputs** for a single large image.
* For **videos**, each frame is one large image, so data becomes even larger.
* That is why **normal neural networks are not practical** for image or video tasks.

**Why We Use CNNs**

* CNNs process data **more efficiently**.
* They do not take the whole image at once.
* They **analyze small patches** or small parts of the image.
* They look for **patterns**, **edges**, **corners**, and **shapes** in each patch.
* This reduces the input size and makes training **faster and more optimized**.

**CNN Architecture**

A CNN usually has multiple layers:

1. **Convolution Layer**: applies filters (kernels) to small patches of the image.
2. **Pooling Layer**: reduces the size of data by keeping important features.
3. **Fully Connected Layer**: combines all extracted features and gives output (like identifying the object).

**Applications of CNNs:**

* Image classification (like “cat” or “dog”)
* Object detection (finding where things are in an image)
* Video analysis

**Transformers**

* Transformers are another type of **neural network architecture**.
* They are used for **sequential data** (like text).
* They replaced old RNNs in many applications.
* Transformers work using a system called **Attention Mechanism**.
* Attention helps the model find which parts of data are more important.

**Example:**  
Sentence – “The dog chased the cat because it was scared.”  
The model uses attention to figure out that “it” refers to “dog”.

**Transformers are the base of GPT (Generative Pretrained Transformer)** models.  
They can read, understand, and generate text.

**Generative AI (GenAI)**

* A part of AI that **creates new content** such as text, images, audio, video, and code.
* Examples of GenAI tools:
  + **Text:** ChatGPT (by OpenAI, funded by Microsoft), Claude (by Anthropic, funded by Amazon), Gemini (by Google), Llama (by Meta)
  + **Images:** DALL·E, Midjourney, Stable Diffusion
  + **Audio:** Eleven Labs, Bark, MusicGen
  + **Videos:** Sora, Runway, Veed
  + **Code:** GitHub Copilot, Code Llama, CodeWhisperer (by Amazon)

**Natural Language Processing (NLP)**

* NLP = **Natural Language Processing**
* It teaches machines to **understand, interpret, and generate human language**.
* Used in chatbots, translators, and virtual assistants.

**Large Language Models (LLMs)**

* LLM = **Large Language Model**
* They are used to solve NLP tasks.
* They are called “large” because:
  + They are trained on **huge amounts of data** (internet text, books, articles).
  + They have **billions or trillions of parameters** (weights and biases).

**Examples of LLMs:**  
GPT-4, Claude, Llama

**Training of LLMs:**

* Uses methods like **Reinforcement Learning with Human Feedback (RLHF)** to make answers accurate, safe, and non-toxic.

**Computer Vision**

* Computer Vision is another field of AI.
* It teaches computers to **see and understand images or videos**.
* CNNs are mainly used here.

**Applications:**

* Face recognition
* Self-driving cars (to detect objects and roads)

**Summary of the Lecture**

* Started with **AI**
* Then **Machine Learning**
* Then **Deep Learning** and **Neural Networks**
* Learned about **CNNs**, **Transformers**, **Generative AI**, **NLP**, **LLMs**, and **Computer Vision**
* Each part works together to make modern AI tools like ChatGPT, Copilot, and Sora possible.