Module-05

Applications of AI - Natural Language Processing, Text Classification and Information Retrieval, Speech Recognition , Image processing and computer vision, Robotics.

Applications of AI:

Artificial Intelligence (AI) refers to the ability of machines or computer programs to perform tasks that typically require human intelligence. These tasks include learning, reasoning, problem-solving, understanding language, and perception. AI has numerous practical applications across different fields.

Natural Language Processing:

Language is a structured system of communication used by humans, consisting of spoken, written, or signed symbols. It allows people to express thoughts, convey information, and interact with one another. Key components of language include:

- **Phonology**: Sounds of speech
- **Morphology**: Structure of words
- **Syntax**: Rules for sentence formation
- **Semantics**: Meaning of words and sentences
- **Pragmatics**: Contextual use of language

In the context of **Artificial Intelligence (AI)**, **language** typically refers to the system of communication used by humans (natural language) or machines (programming and formal languages) for understanding, generating, and interacting with information.

Natural Language for human communication and AI interaction.

Programming Languages to implement AI models.

Formal Languages for logical reasoning and symbolic AI.

1. **Natural Language**

   Definition: The way humans communicate using spoken or written words (e.g., English, Spanish, Chinese).

   - **Relevance in AI**: AI systems aim to **understand, interpret, generate, and respond to natural language** through **Natural Language Processing (NLP)**.

   Examples:

   - Chatbots and virtual assistants understanding user queries.
   - Machine translation (e.g., Google Translate).
   - Text summarization, sentiment analysis.

2. **Programming Language**

Definition: A formal language used to write software programs that instruct computers to perform specific tasks.

- **Relevance in AI**: AI models and algorithms are built using languages like **Python, R, Java, or C++**.
- **Examples**:
  - Writing machine learning models in Python using TensorFlow or PyTorch.
  - Creating AI-based applications

3. **Formal Language / Logic Language**

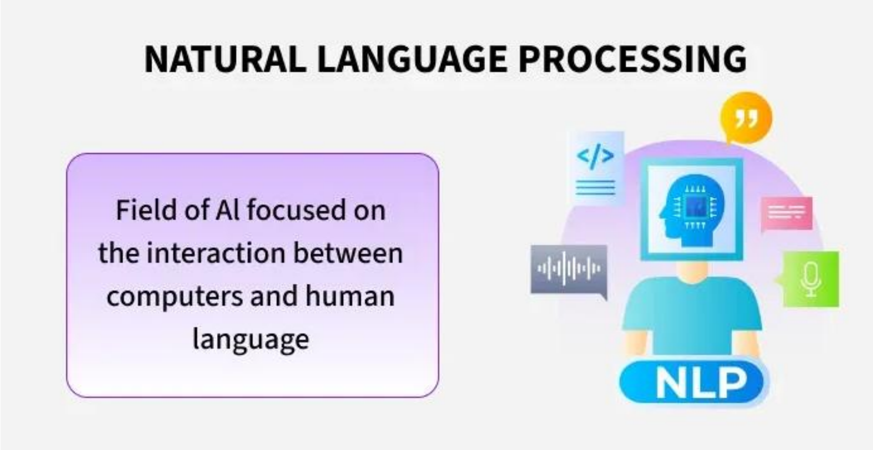
   Definition: A symbolic system used in logic and mathematics to express rules, facts, and relationships.

   Relevance in AI: Used in **expert systems, knowledge representation, and reasoning**.

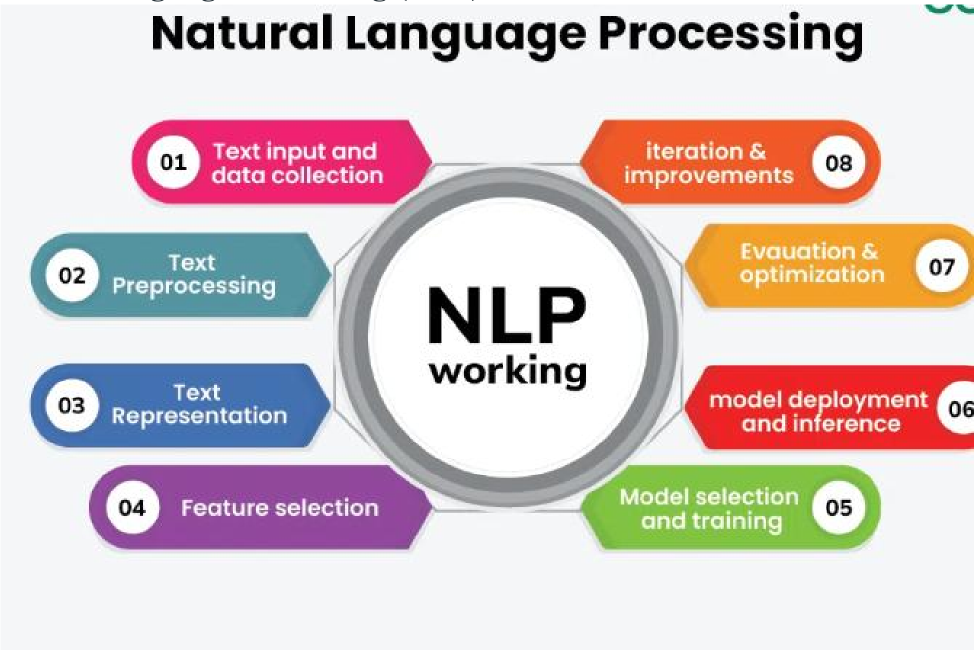
   Examples:
   - Propositional and predicate logic in rule-based AI.
     Description logic in semantic web technologies.

1) **Natural Language Processing (NLP)** :It is a subfield of **Artificial Intelligence (AI)** that focuses on enabling computers to understand, interpret, and generate human language in a way that is both meaningful and useful. NLP bridges the gap between human communication (natural languages) and computer understanding (structured data).

- **Natural Language Processing (NLP)** refers to the computational techniques used to process and analyze large amounts of natural language data.
- It combines **linguistics**, **computer science**, and **machine learning** to allow machines to understand and interact with humans using spoken or written language.
- Natural Language Processing (NLP) is a field that combines computer science, artificial intelligence and language studies.
- It helps computers understand, process and create human language in a way that makes sense and is useful. With the growing amount of text data from social media, websites and other sources, NLP is becoming a key tool to gain insights and automate tasks like analysing text or translating languages.



How Natural Language Processing (NLP) Works



Working in natural language processing (NLP) typically involves using computational techniques to analyze and understand human language. This can include tasks such as language understanding, language generation and language interaction.

1. Text Input and Data Collection

- **Data Collection:** Gathering text data from various sources such as websites, books, social media or proprietary databases.
- **Data Storage:** Storing the collected text data in a structured format, such as a database or a collection of documents.

2. Text Preprocessing

Preprocessing is crucial to clean and prepare the raw text data for analysis. Common preprocessing steps include:

- **Tokenization:** Splitting text into smaller units like words or sentences.

- **Lowercasing:** Converting all text to lowercase to ensure uniformity.
- **Stopword Removal:** Removing common words that do not contribute significant meaning, such as "and," "the," "is."
- **Punctuation Removal:** Removing punctuation marks.
- **Stemming and Lemmatization:** Reducing words to their base or root forms. Stemming cuts off suffixes, while lemmatization considers the context and converts words to their meaningful base form.
- **Text Normalization:** Standardizing text format, including correcting spelling errors, expanding contractions and handling special characters.

3. Text Representation

- **Bag of Words (BoW):** Representing text as a collection of words, ignoring grammar and word order but keeping track of word frequency.
- **Term Frequency-Inverse Document Frequency (TF-IDF)**: A statistic that reflects the importance of a word in a document relative to a collection of documents.
- **Word Embeddings:** Using dense vector representations of words where semantically similar words are closer together in the vector space (e.g., Word2Vec, GloVe).

4. Feature Extraction

Extracting meaningful features from the text data that can be used for various NLP tasks.

- **N-grams**: Capturing sequences of N words to preserve some context and word order.
- **Syntactic Features**: Using parts of speech tags, syntactic dependencies and parse trees.
- **Semantic Features**: Leveraging word embeddings and other representations to capture word meaning and context.

5. Model Selection and Training

Selecting and training a machine learning or deep learning model to perform specific NLP tasks.

- **Supervised Learning**: Using labeled data to train models like Support Vector Machines (SVM), Random Forests or deep learning models like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).
- **Unsupervised Learning**: Applying techniques like clustering or topic modeling (e.g., Latent Dirichlet Allocation) on unlabeled data.
- **Pre-trained Models:** Utilizing pre-trained language models such as BERT, GPT or transformer-based models that have been trained on large corpora.

6. Model Deployment and Inference

Deploying the trained model and using it to make predictions or extract insights from new text data.

- **Text Classification**: Categorizing text into predefined classes (e.g., spam detection, sentiment analysis).
- **Named Entity Recognition (NER)**: Identifying and classifying entities in the text.
- **Machine Translation**: Translating text from one language to another.
- **Question Answering**: Providing answers to questions based on the context provided by text data.

7. Evaluation and Optimization

Evaluating the performance of the NLP algorithm using metrics such as accuracy, precision, recall, F1-score and others.

- **Hyperparameter Tuning**: Adjusting model parameters to improve performance.
- **Error Analysis**: Analyzing errors to understand model weaknesses and improve robustness.

1.1 Language Model:A **Language Model (LM)** in AI is a mathematical model that predicts the probability of a sequence of words. It understands and generates human-like language based on patterns it has learned from large datasets. These models are essential for enabling machines to read, understand, and produce human language.

Types of Language Models

1. **Statistical Language Models**
   These models use probabilities based on word occurrence in a corpus.
   - **N-gram models**: Estimate the probability of a word based on the previous *n-1* words.
     Example: Bigram model (n=2), Trigram model (n=3).
2. **Neural Language Models**
   These use neural networks to learn more complex patterns and contexts.
   - **Recurrent Neural Networks (RNNs)**
   - **Long Short-Term Memory networks (LSTMs)**
   - **Transformers** (e.g., BERT, GPT)

How Language Models Work

Language models are trained on large text corpora. The training process involves:

- **Tokenization**: Breaking text into tokens (words or subwords).
- **Contextual Learning**: Understanding word meaning based on surrounding words.

- **Prediction**: Learning to predict the next word, fill in blanks, or classify sentences.

For example, given the input:

"The cat sat on the ___."

The model might predict: "mat."

Applications of Language Models in AI

1. **Text Generation** – Generating coherent and contextually relevant sentences (e.g., ChatGPT).
2. **Machine Translation** – Translating text between languages (e.g., Google Translate).
3. **Speech Recognition** – Converting spoken language to text.
4. **Text Summarization** – Condensing large texts into shorter versions.
5. **Question Answering** – Answering user queries in natural language.
6. **Sentiment Analysis** – Detecting emotions or opinions in text.

Modern Language Models

- **BERT (Bidirectional Encoder Representations from Transformers)**
  Focuses on understanding the context from both directions.
- **GPT (Generative Pre-trained Transformer)**
  Excels at generating human-like text by predicting next words in a sequence.
- **T5 (Text-To-Text Transfer Transformer)**
  Converts all NLP tasks into a text-to-text format.

1.2 N-gram Character Models

Definition: An **N-gram character model** is a probabilistic language model that predicts the next character in a sequence based on the preceding **N-1** characters.

How it works:

- It breaks down the text into overlapping substrings of length **N**.
- It then calculates the probability of each character given the previous **N-1** characters.

Example (Trigram = 3-gram):

For the word: "hello", the character trigrams are:

"hel", "ell", "llo"

To predict the next character after "he", the model looks at the frequency of sequences starting with "he" in the training corpus.

```
from collections import defaultdict

def generate_char_ngrams(text, n):
    ngrams = defaultdict(int)
    for i in range(len(text) - n + 1):
```

```
        gram = text[i:i+n]
        ngrams[gram] += 1
    return dict(ngrams)

text = "hello hello"
n = 3
char_ngrams = generate_char_ngrams(text, n)
print("Character Trigrams:", char_ngrams)
```

Output:

1.3 Smoothing in N-gram Models

Definition: Smoothing is used to handle the **zero probability problem** in N-gram models — when an N-gram is missing from the training data, it would otherwise get a probability of 0.

Common Techniques:

1. Add-one (Laplace) Smoothing:
   Adds 1 to each count to avoid zero probabilities.

$$P(w_i|w_{i-1}) = \frac{C(w_{i-1}, w_i) + 1}{C(w_{i-1}) + V}$$

   Where:
   - $C(w_{i-1}, w_i)$ is the count of the bigram.
   - $C(w_{i-1})$ is the count of the first word.
   - V is the vocabulary size.
2. Add-k Smoothing (general form of Add-one):
   Adds a small constant k instead of 1.
3. Good-Turing Discounting:
   Adjusts the probabilities based on the frequency of frequencies (i.e., how many bigrams occur once, twice, etc.).

```
from collections import defaultdict
def add_one_smoothing_bigrams(text):
    words = text.lower().split()
    bigram_counts = defaultdict(int)
    unigram_counts = defaultdict(int)
    # Count unigrams and bigrams
    for i in range(len(words) - 1):
        unigram_counts[words[i]] += 1
        bigram_counts[(words[i], words[i+1])] += 1
    unigram_counts[words[-1]] += 1
    # Vocabulary size
    vocab = set(words)
    V = len(vocab)
    # Compute smoothed probabilities
```

```
    print("Smoothed Bigram Probabilities (Add-One):")
    for w1 in vocab:
        for w2 in vocab:
            count_bigram = bigram_counts[(w1, w2)]
            count_unigram = unigram_counts[w1]
            prob = (count_bigram + 1) / (count_unigram + V)
            print(f"P({w2} | {w1}) = {prob:.4f}")
# Example text
text = "i love natural language processing i love machine learning"
add_one_smoothing_bigrams(text)
```

Output:

Smoothed Bigram Probabilities (Add-One):

P(i | i) = 0.0833

P(love | i) = 0.2500

P(natural | i) = 0.0833

1.4 N-gram Word Models Definition:

N-gram word models predict the next word in a sequence based on the previous **N-1** words.

Example (Bigram Model = 2-gram):

Consider this corpus: "I love natural language processing"

Bigram word pairs:

- ("I", "love")
- ("love", "natural")
- ("natural", "language")
- ("language", "processing")

To predict the next word after "natural", the model checks what words followed "natural" in the training data.

```
from collections import defaultdict

def build_bigram_model(text):
    words = text.lower().split()
    unigram_counts = defaultdict(int)
    bigram_counts = defaultdict(int)

    for i in range(len(words) - 1):
        unigram_counts[words[i]] += 1
        bigram_counts[(words[i], words[i+1])] += 1
    unigram_counts[words[-1]] += 1  # last word

    print("Bigram Probabilities (no smoothing):")
    for (w1, w2), count in bigram_counts.items():
```

```
    probability = count / unigram_counts[w1]
    print(f"P({w2} | {w1}) = {probability:.4f}")

# Example usage
text = "i love natural language processing i love machine learning"
build_bigram_model(text)
```

Output:

Bigram Probabilities (no smoothing):

P(love | i) = 1.0000

P(natural | love) = 0.5000

P(language | natural) = 1.0000

P(processing | language) = 1.0000

P(i | processing) = 1.0000

P(machine | love) = 0.5000

P(learning | machine) = 1.0000

For example: P(love | i) = 1.0 means every time i appeared, it was followed by love.

love appeared twice and was followed once by natural and once by machine, so P(natural | love) = 0.5.

1.5 Applications of NLP in AI.

1.Information Retrieval (IR)

NLP improves search engines by interpreting user queries more effectively.

- **Search Engines (Google, Bing):** Understands query intent, synonyms, and context.
- **Semantic Search:** Goes beyond keywords to understand meaning (e.g., "How tall is the Eiffel Tower?" → returns height).
- **Document Classification:** Identifying and indexing documents based on content.

🖿 2. Information Extraction (IE)

NLP identifies and pulls structured information from unstructured text.

- **Named Entity Recognition (NER):** Extracts names, places, dates, etc., from text.
- **Relation Extraction:** Detects relationships between entities (e.g., "Steve Jobs founded Apple").
- **Event Extraction:** Identifies events, times, and participants from text sources.

3. Text Classification

Automatically categorizes text into predefined classes.

- **Spam Detection:** Filters out unwanted emails or messages.

- **Sentiment Analysis:** Determines if text expresses positive, negative, or neutral sentiment.
- **Topic Labeling:** Classifies articles by subject (e.g., sports, politics, technology).

💬 4. Machine Translation

Converts text from one language to another.

- **Google Translate, DeepL:** Use NLP and deep learning to perform real-time, context-aware translation.
- **Multilingual Applications:** Enables global access to content and services.

🗣 5. Speech Recognition & Processing

Converts spoken language into text and vice versa.

- **Voice Assistants (Siri, Alexa, Google Assistant):** Understand and respond to voice commands.
- **Transcription Tools:** Convert speech into written text (e.g., Otter.ai, Google Docs Voice Typing).

6. Chatbots and Conversational Agents

Enable automated dialogue systems to simulate conversation.

- **Customer Support Bots:** Handle FAQs, complaints, and service requests.
- **Virtual Assistants:** Schedule meetings, set reminders, fetch information.
- **Mental Health Chatbots:** Offer conversational support (e.g., Woebot, Wysa).

🕮 7. Text Summarization

Generates concise summaries from long text documents.

- **Extractive Summarization:** Selects key sentences directly from the source.
- **Abstractive Summarization:** Rewrites text in a condensed form with new phrasing.

📚 8. Question Answering (QA)

Provides precise answers to user queries from a large dataset or corpus.

- **Open-Domain QA:** Like Google or ChatGPT answering questions from general knowledge.
- **Closed-Domain QA:** Specific to domains like medicine, law, or finance (e.g., legal document analysis).

✍ 9. Text Generation

Creates new text based on input or learned patterns.

- **Language Models (e.g., GPT, BERT):** Write emails, stories, code, etc.

- **Content Creation Tools:** Generate product descriptions, blog posts, and social media content.

📝 10. Grammar and Spell Checking

Improves the quality of written text.

- **Tools like Grammarly, Microsoft Editor:** Suggest corrections, improve clarity and tone.
- **Autocorrect and Predictive Text:** Used in smartphones and writing tools.

11. Optical Character Recognition (OCR)

Extracts text from images or scanned documents.

- **Digitizing Documents:** Convert printed books, invoices, or forms into editable formats.
- **Handwriting Recognition:** Extracts handwritten notes into digital text.

12. Biomedical NLP

Analyzes scientific texts and medical records.

- **Clinical Notes Mining:** Extract patient data from doctors' notes.
- **Drug Interaction Detection:** NLP models identify interactions and adverse effects from literature.
- **Genomics Text Mining:** Helps in research by extracting biological relationships.

💼 13. Legal and Financial NLP

Automates the processing of dense, domain-specific text.

- **Contract Analysis:** Summarizes clauses and flags risky language.
- **Compliance Monitoring:** Monitors regulations and legal updates.
- **Financial News Analysis:** Detects market sentiment from reports.

📊 14. Social Media and Opinion Mining

Monitors public sentiment and trends.

- **Brand Monitoring:** Analyzes user sentiment about a product or service.
- **Trend Detection:** Identifies viral topics and user engagement.
- **Misinformation Detection:** Identifies fake or misleading posts.

15. Document Management and Retrieval

Facilitates better management of large text corpora.

- **Automatic Tagging:** Assigns relevant tags or keywords to documents.
- **Smart Search in Enterprise Systems:** Helps employees find documents efficiently.

2. Text Classification and Information Retrieval

Text Classification, also known as text categorization or text tagging, is a technique used in machine learning and artificial intelligence to automatically categorize text into predefined classes or categories.

- It involves training a model on a labeled dataset, where each text example is associated with a specific class or category.
- The trained model can then be used to classify new, unseen texts into the appropriate categories.
- Text classification is a fundamental task in Artificial Intelligence (AI) and Natural Language Processing (NLP) that involves categorizing text into predefined groups or labels.
- The goal is to automatically assign tags or categories to textual data, such as emails, reviews, or news articles.This process is widely used in applications like spam detection, sentiment analysis, topic labeling, and language detection.
- AI models—especially machine learning and deep learning algorithms—are trained on labeled datasets to learn patterns and make accurate predictions on unseen text.
- Popular techniques include traditional methods like Naive Bayes and Support Vector Machines (SVM), as well as modern neural network models like Transformers (e.g., BERT).
- Effective text classification helps in organizing, filtering, and gaining insights from large volumes of textual information.

Text Classification in AI (with Real-World Example)

Text classification is an important task in Artificial Intelligence (AI) where text data is automatically categorized into predefined labels or groups.
It is commonly used in Natural Language Processing (NLP) to make sense of large volumes of unstructured text.

Real-World Example:

One common use is **spam detection** in email services. AI models analyze the content of incoming emails and classify them as either "Spam" or "Not Spam." For example, Gmail uses text classification to filter out unwanted promotional or phishing emails and move them to the spam folder.

Other Examples:

- **Sentiment Analysis**: Companies use AI to classify customer reviews as positive, negative, or neutral to understand public opinion about products.
- **News Categorization**: News apps use text classification to group articles into categories like Sports, Politics, Technology, etc.

Write down the Naïve Bayesian classifier program in text classification

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB

# Sample training data (text, label)
texts = [
    "I love this movie, it is fantastic!",
    "This film is terrible and boring.",
    "What a great and exciting movie!",
    "I did not like this movie at all.",
    "An amazing and thrilling experience.",
    "Worst movie I have ever seen."
]
labels = ['positive', 'negative', 'positive', 'negative', 'positive',
'negative']

# Convert text data to numeric features
vectorizer = CountVectorizer()
X_train = vectorizer.fit_transform(texts)

# Train a Naive Bayes classifier
model = MultinomialNB()
model.fit(X_train, labels)

# Test data
test_texts = ["I really enjoyed the movie", "It was a dull and bad movie"]

# Transform test data and predict
X_test = vectorizer.transform(test_texts)
predictions = model.predict(X_test)

for text, label in zip(test_texts, predictions):
    print(f"Text: '{text}' => Classified as: {label}")
```

- Converts text into a numeric format using `CountVectorizer`.

- Trains a simple Naive Bayes model on labeled movie reviews.
- Predicts sentiment (positive or negative) on new reviews.

Output:

Text: 'I really enjoyed the movie' => Classified as: positive

Text: 'It was a dull and bad movie' => Classified as: negative

2.1 Applications of Text Classification in AI

Text classification is a crucial task in Artificial Intelligence (AI) and Natural Language Processing (NLP) where text data is automatically categorized into predefined labels. This technology has numerous real-world applications that impact daily life and business operations significantly.

1. **Spam Detection in Emails**
   One of the most common applications of text classification is in filtering unwanted emails or spam. Email services like Gmail or Outlook use AI models to classify incoming emails as "spam" or "not spam" based on the content and metadata. This helps users avoid phishing attacks and irrelevant advertisements, improving email security and user experience.
2. **Sentiment Analysis**
   Text classification is widely used to analyze opinions expressed in texts such as product reviews, social media posts, or customer feedback. Sentiment analysis models classify text into categories such as positive, negative, or neutral. Businesses leverage this to monitor brand reputation, understand customer satisfaction, and make data-driven marketing decisions.
3. **Topic Labeling and News Categorization**
   News websites and content aggregators use text classification to automatically assign categories like Sports, Politics, Technology, or Entertainment to articles. This helps users easily find content of interest and enables efficient content organization and recommendation systems.
4. **Language Detection**
   AI systems classify text by language, which is essential for translation services, multilingual chatbots, and global content delivery. This classification enables automatic routing of queries to appropriate language models and improves communication in diverse linguistic settings.
5. **Document Organization and Email Routing**
   In large organizations, text classification helps in automatically sorting and routing

documents or emails to the relevant departments or personnel based on their content. For example, customer support tickets can be categorized into billing, technical issues, or general inquiries for faster resolution.

6. **Medical Text Classification**
   In healthcare, text classification aids in organizing clinical notes, patient records, and research papers. It can automatically detect mentions of diseases, symptoms, or treatments, assisting in faster diagnosis and research.
7. **Content Moderation**
   Social media platforms use text classification to detect and filter harmful or inappropriate content such as hate speech, cyberbullying, or violent language, ensuring a safer online environment.
8. **Legal Document Classification**
   Law firms and courts use AI to classify legal documents by type, case relevance, or jurisdiction, which speeds up document review and legal research.
9. **Customer Support Automation**
   Many companies use text classification to automatically categorize customer queries received through chatbots, emails, or helpdesks. By classifying issues into predefined categories like "technical support," "billing," or "product inquiry," the system can either provide instant automated responses or route the query to the right human agent. This reduces response time and improves customer satisfaction.
10. **Fake News Detection**
    With the rise of misinformation online, AI models use text classification to analyze news articles or social media posts and classify them as "real" or "fake." This helps platforms and users identify misleading or false content, contributing to the fight against the spread of misinformation and maintaining content credibility.

3. Information retrieval in AI

Information Retrieval (IR) in AI refers to the process of obtaining relevant information or data from a large collection of resources based on a user's query or need. The goal is to find, rank, and deliver the most useful documents or data snippets that match the user's search intent.

In AI, IR systems often use techniques like natural language processing (NLP), machine learning, and semantic analysis to understand queries better and retrieve more accurate results.

Real-world Example:

Search Engines (e.g., Google)

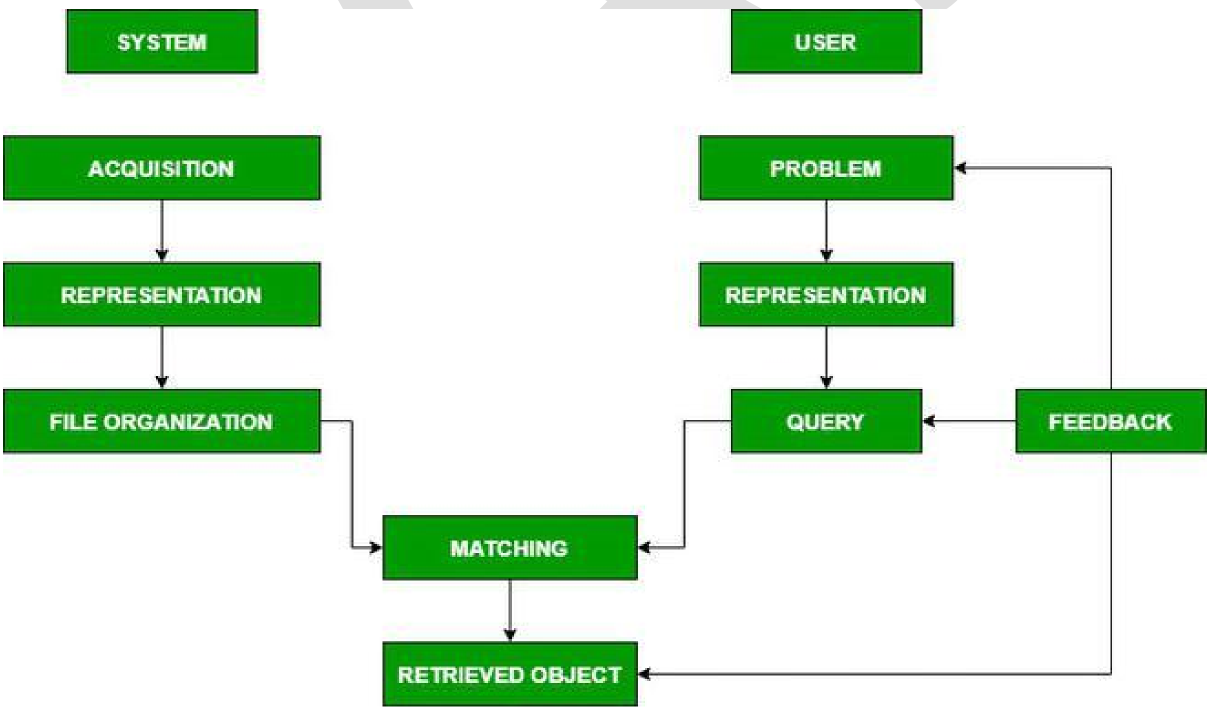
- When you type a question or keywords into Google, it uses information retrieval to scan billions of web pages.
- The AI-powered system analyzes your query, matches it with the most relevant documents, ranks them based on relevance and authority, and then shows you the best results.

For instance, if you search for **"best Italian restaurants near me"**, the IR system understands your intent, looks through numerous sources (websites, reviews, maps), and returns a list of nearby Italian restaurants, often sorted by ratings and distance.

Information Retrieval (IR) helps to find relevant information from large collections of documents. It can be defined as **a software program that deals with the organization, storage, retrieval and evaluation of information from documents**. It is like a smart librarian who doesn't give you direct answers but tells you where to find the right book like this IR system scans them and pulls out the ones that **match your query.**

Components of Information Retrieval/ IR Model

The **Information Retrieval (IR) model** can be broken down into key components that involve both the system and the user. Here's how it works in a simple flow:



User Side (Search Process)

- **Problem Identification:** A student wants to learn about machine learning and types a query into a search engine.

- **Representation:** The user converts their need into a search query using keywords or phrases like instead of asking *"How do machines learn?"* the student types "machine learning basics" into Google and the problem is converted into a query (keywords or phrases).
- **Query:** The user submits the search query into IR system.
- **Feedback:** User can refine or modify the search based on the retrieved results.

System Side (Retrieval Process)

- **Acquisition:** The system collects and stores a large number of documents or data sources. It can includes web pages, books, research papers or any text-based information.
- **Representation:** Each document in the system is analyzed and represented in a structured way using keywords (terms). Example: If the document talks about "machine learning" it is tagged with relevant terms like "AI, deep learning, algorithms, models" to help retrieval.
- **File Organization:** The documents are **indexed and stored** efficiently so the system can quickly find relevant ones. Like organizing a library so books can be found easily based on topics.
- **Matching:** The system **compares the user's search query with stored documents** to find the best matches. It uses **matching functions** that rank documents based on **relevance.**
- **Retrieved Object:** The system returns the most **relevant documents** to the user. These documents are ranked so the most useful ones appear **at the top**.

Information Retrieval (IR) is the process of finding relevant information from a large repository in response to a user's query. It involves indexing, searching, and ranking documents to provide the most useful results. IR systems underpin search engines, digital libraries, and question-answering systems.

4.1 IR Scoring Functions

IR scoring functions measure how relevant a document is to a user's query. They assign scores to documents based on matching terms, frequency, and importance.

Common Scoring Functions:

- **TF-IDF (Term Frequency-Inverse Document Frequency):**
  Measures the importance of a term in a document relative to its occurrence in all documents.
    - **TF**: How often a term appears in a document.

- **IDF**: How rare the term is across all documents.

Formula:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \log\left(\frac{N}{\text{DF}(t)}\right)$$

- **BM25:**
  A probabilistic model that improves over TF-IDF by considering term saturation and document length.

Example:

If you search for "apple", documents with more frequent occurrences of "apple" get higher scores, but if "apple" appears in almost every document, it gets a lower score due to IDF.

```
def simple_score(query, document):
    score = 0
    for word in query.split():
        score += document.lower().split().count(word.lower())
    return score

query = "apple banana"
document = "Apple banana fruit apple"
print(simple_score(query, document))
```

Output: 3

4.2 IR System Evaluation

Evaluation measures how well an IR system retrieves relevant documents.

Key metrics:

- **Precision:** Proportion of retrieved documents that are relevant.

$$\text{Precision} = \frac{\text{Relevant documents retrieved}}{\text{Total documents retrieved}}$$

- **Recall:** Proportion of relevant documents that are retrieved.

$$\text{Recall} = \frac{\text{Relevant documents retrieved}}{\text{Total relevant documents}}$$

- **F1 Score:** Harmonic mean of precision and recall.
- **Mean Average Precision (MAP):** Average precision across multiple queries.

4

```
retrieved = ["D1", "D2", "D3"]
relevant = ["D2", "D3", "D4"]
precision = len(set(retrieved) & set(relevant)) / len(retrieved)
print(f"Precision: {precision:.2f}")
```

Output:

Precision: 0.67

Explanation:

- Retrieved = ["D1", "D2", "D3"]
- Relevant = ["D2", "D3", "D4"]
- Relevant & Retrieved = ["D2", "D3"] → 2 documents
- Precision = 2 / 3 = 0.67 (rounded to two decimals)

4.3 IR Refinements

Refinements improve retrieval quality by enhancing query understanding or document representation:

- **Query Expansion:** Add synonyms or related terms to the query to improve recall. *Example:* Query "car" expanded to include "automobile," "vehicle."
- **Relevance Feedback:** User marks some results as relevant or irrelevant; the system adjusts the query accordingly.
- **Stemming and Lemmatization:** Reducing words to root forms (e.g., "running" → "run") to match different word forms.
- **Stop word Removal:** Remove common words like "the," "is," which carry little meaning.

```
stop_words = {"the", "is", "at", "which", "on"}
query = "What is the capital of France"
refined_query = ' '.join([w for w in query.lower().split() if w not in stop_words])
print(refined_query)
```

Output: what capital of france

4.4 Question Answering (QA)

QA systems are a specialized form of IR that directly answer user questions instead of returning documents.

- They combine IR with **Natural Language Processing (NLP)** to understand questions and extract precise answers.
- Examples: Siri, Alexa, Google Assistant.

Example:

User asks, "Who is the prime minister of India ?"

The QA system searches through knowledge bases or documents and returns: "**Narendra Modi.**"

```
knowledge_base = {
    "capital of India": "New Delhi",
    "capital of USA": "Washington, D.C.",
}

def answer_question(question):
    for key in knowledge_base:
        if key in question.lower():
            return knowledge_base[key]
    return "I don't know."

print(answer_question("What is the capital of India?"))
# Output: New Delhi
```

4.5 **PageRank Algorithm**

PageRank is an algorithm originally developed by Google to rank web pages in search results. It measures the importance of a webpage based on the number and quality of links pointing to it.

- The more important pages link to a page, the higher its PageRank.
- Links from highly ranked pages count more than links from low-ranked pages.

Real-world Example:

Imagine a university website with pages about **Admissions**, **Courses**, and **Faculty**.

- The **Faculty** page links to the **Courses** page.
- The **Admissions** page links to both **Courses** and **Faculty**.
- The **Courses** page links back to **Faculty**

Pages with many incoming links from important pages get a higher rank, meaning they are more "important" in the network.

PageRank Algorithm Overview:

1. Initialize each page's rank equally (e.g., 1/N for N pages).
2. For each page, update its rank based on the ranks of pages linking to it.
3. Use a damping factor (usually 0.85) to simulate the probability of continuing to click links.
4. Repeat until ranks converge (don't change much).

Formula for PageRank of page i:

$$PR(i) = \frac{1-d}{N} + d \sum_{j \in M(i)} \frac{PR(j)}{L(j)}$$

- d = damping factor (e.g., 0.85)
- N = total number of pages
- $M(i)$ = set of pages linking to page i
- $L(j)$ = number of outgoing links from page j

PageRank Algorithm:

Step 01:Assign initial ranks:

Give every page the same starting rank (e.g., 1).

Step 02:Distribute rank through links:

Each page passes its rank equally to all pages it links to.

Step 03:**Calculate new rank:**

A page's new rank is the sum of the ranks it receives from other pages.

Step 04: Apply damping factor:

Adjust ranks by multiplying with a factor (usually 0.85) to simulate "random surfing."

Step 05:Repeat until stable:

Keep updating ranks by repeating steps 2-4 until the ranks don't change much.

```
links = {
    'A': ['B', 'C'],
    'B': ['C'],
    'C': ['A']
}

ranks = {page: 1 for page in links}
d = 0.85

for _ in range(10):  # iterate 10 times
    new_ranks = {}
    for page in links:
        new_ranks[page] = (1 - d) + d * sum(ranks[p] / len(links[p]) for p in
links if page in links[p])
    ranks = new_ranks
```

```
print(ranks)
```

output:

{'A': 1.0185901161510577, 'B': 0.7402389847998907, 'C': 1.2411708990490516}

- Page **C** has the highest rank because both **A** and **B** link to it.
- Page **A** is next because **C** links back to it.
- Page **B** has the lowest rank here.

Applications page link algorithm:

1. **Search Engine Ranking:**
   Google's original search algorithm used PageRank to rank web pages by importance based on incoming links.
2. **Social Network Analysis:**
   To find influential users by analyzing connections and followers.
3. **Recommendation Systems:**
   Ranking items (like products or movies) based on user interactions and references.
4. **Academic Paper Ranking:**
   Identifying important research papers by citation analysis

4.6 HITS Algorithm (Hyperlink-Induced Topic Search)

What is HITS

HITS is another link analysis algorithm. It assigns two scores to each page:

- **Authority score:** How valuable the content of the page is.
- **Hub score:** How good the page is at linking to valuable content.

Good **hubs** point to good **authorities**, and good **authorities** are pointed to by good **hubs**.

Real-world Example:

Suppose you search for **Python programming**.

- Pages that provide good content on Python are **authorities**.
- Pages that list many useful Python resources or tutorials are **hubs**.

HITS Algorithm :

Step 01:Initialize scores:

Give every page an initial **authority** score and **hub** score of 1.

Step 02: Update authority scores:

A page's authority score = sum of the **hub scores** of pages that link to it.

Step 03: Update hub scores:

A page's hub score = sum of the **authority scores** of pages it links to.

Step 04: Normalize scores:

Scale the authority and hub scores so they don't get too big (e.g., divide by total norm).

Step 05: Repeat until stable:

Keep updating and normalizing scores until the numbers stop changing much.

```
links = {
    'A': ['B', 'C'],
    'B': ['C'],
    'C': ['A']
}

auth = {p: 1 for p in links}
hub = {p: 1 for p in links}

for _ in range(10):
    auth = {p: sum(hub[q] for q in links if p in links[q]) for p in links}
    hub = {p: sum(auth[q] for q in links[p]) for p in links}

print("Authorities:", auth)
print("Hubs:", hub)
```

Output:Authorities: {'A': 1, 'B': 1, 'C': 1}

Hubs: {'A': 2, 'B': 1, 'C': 1}

Page **A** is a strong **hub** (links to both B and C).

All pages have equal **authority** here because of the simple graph and limited iterations.

.You can increase iterations for more refined scores.

APPLICATIONS OF HITS ALGORITHMS:

- **Focused Search:**

  Identifying good **authorities** (trusted content) and **hubs** (resource lists) in a specific topic or query.

- **Web Mining:**

  Finding quality pages that either provide information (authorities) or link to quality content (hubs).

- **Community Detection:**

  Understanding roles of pages/users in a network (who provides content vs who points to content).

- **Spam Detection:**
  Differentiating between genuine hubs/authorities and spam my pages linking to many irrelevant pages.

Speech Recognition: Speech recognition is the process by which a computer takes audio signals of spoken language and transcribes them into text.

- This involves several stages: **signal processing**, **feature extraction**, **acoustic modeling**, **language modeling**, and **decoding**.

Speech Recognition Works

- Speech Recognition, also known as **Automatic Speech Recognition (ASR)** or **Speech-to-Text (STT)**, is a field of Artificial Intelligence (AI) that allows computers to **convert spoken language into written text**.
- It enables machines to **"listen"**, interpret, and respond to human speech, forming a crucial bridge between **human communication and machine understanding**.

A. Audio Input

The system receives spoken input via a microphone. This input is a **continuous analog signal**, which needs to be digitized.

B. **Pre-processing**

- **Noise reduction**: Filters out background sounds.
- **Normalization**: Adjusts volume levels.
- **Framing**: Breaks the audio into small overlapping segments (typically 10–25ms).

C. Feature Extraction

Transforms raw audio into a form usable by machine learning models.

- Common features include:
  - **MFCC (Mel-Frequency Cepstral Coefficients)**
  - **Spectrograms**
  - **Log Mel spectrograms**

These capture the frequency and intensity of the sound.

D. Acoustic Model

- Learns the relationship between audio features and phonemes (basic units of sound).
- Often based on **deep neural networks (DNNs)**, **Convolutional Neural Networks (CNNs)**, or **Recurrent Neural Networks (RNNs)** like **LSTM** and **transformers**.

E. Language Model

- Predicts the likelihood of word sequences.

- Helps disambiguate similar-sounding phrases (e.g., "recognize speech" vs. "wreck a nice beach").
- Can be based on:
  - **N-grams**
  - **Statistical models**
  - **Neural networks (e.g., GPT-like transformers)**

F. Decoder

Combines the acoustic and language models to find the most probable text for a given audio signal.

Types of Speech Recognition

Type	Description
Speaker-dependent	Trained for a specific speaker
Speaker-independent	Works for any speaker
Isolated word recognition	Recognizes one word at a time
Continuous speech recognition	Recognizes fluent, connected speech
Multilingual recognition	Supports multiple languages

Components of Speech Recognition

a. **Acoustic Model (AM):**

- The **Acoustic Model** maps **audio signals** to **phonetic units** (like phonemes – the smallest units of sound).
- It learns the statistical relationship between **spoken audio features** and the **linguistic units**.
- It is trained using large datasets containing **audio recordings** and their **transcriptions**.
- **Technology used**: Hidden Markov Models (HMM), Deep Neural Networks (DNN), CNNs, RNNs.

Example: When a user says "dog", the acoustic model processes the waveform and identifies the sounds /d/, /o/, and /g/.

b. Language Model (LM):

- The **Language Model** predicts the **most probable sequence of words**.

- It ensures that the output of the recognizer is **grammatically and contextually correct**.
- It reduces errors by choosing valid combinations of words.
- **Types**:
    - **Statistical Models** (e.g., N-gram models)
    - **Neural Language Models** (e.g., RNNs, Transformers)

Example: In noisy audio, the system might confuse "recognize speech" with "wreck a nice beach". The LM helps choose the more meaningful sentence based on context.

Steps to Build a Speech Recognition System

Step 1: Data Collection

- Collect thousands of hours of voice recordings and their corresponding text transcripts.
- Example: Datasets like LibriSpeech, Common Voice.

Step 2: Pre-processing

- Clean the audio, remove noise, and convert it to a standard format (e.g., mono, 16kHz).
- Segment long recordings into smaller clips.

Step 3: Feature Extraction

- Extract features such as MFCCs (Mel Frequency Cepstral Coefficients) from the audio.
- These features represent the unique patterns of sound for each word.

Step 4: Train the Acoustic Model

- Use the extracted features and phoneme labels to train a model to predict phonemes or audio units.
- Algorithms: HMM, DNN, or CNN-based models.

Step 5: Train the Language Model

- Use large text corpora to understand word usage and predict correct word sequences.
- Helps reduce ambiguity and improves word prediction accuracy.

Step 6: Decoding

- Combine the Acoustic Model and Language Model to convert audio into the most likely word sequence.
- Use decoding algorithms like the Viterbi algorithm or beam search.

Step 7: Post-processing

- Add punctuation, capitalization, and correct minor recognition errors.
- Improve readability and output format.

Applications Speech Recognition.

Speech recognition is a technology that enables a computer or device to identify and process human speech into text. It has a wide range of applications across various fields. The key applications are as follows:

1. **Virtual Assistants:**
   - Speech recognition is used in virtual assistants like **Google Assistant, Siri, and Alexa**.
   - Users can perform tasks such as setting reminders, sending messages, or searching the web through voice commands.
2. **Healthcare:**
   - Doctors use speech recognition tools to **dictate patient notes**, diagnoses, and prescriptions.
   - It improves documentation efficiency and reduces the time spent on typing.
3. **Customer Service:**
   - Many companies use **automated voice response systems** in customer support.
   - These systems can understand and respond to basic customer queries without human intervention.
4. **Accessibility:**
   - Speech recognition assists people with **disabilities** by enabling them to control devices, write documents, or navigate software through voice.
   - It supports **hands-free computing** for people with physical limitations.
5. **Transcription Services:**
   - It is widely used to convert **spoken content into written text**, such as meeting minutes, lectures, and interviews.
   - Tools like **Otter.ai or Google Live Transcribe** are popular in this area.

6. **Language Learning:**
    - Language learning apps use speech recognition to help users **improve pronunciation** and **speaking skills**.
    - It offers instant feedback based on the accuracy of spoken words.
7. **Smart Homes:**
    - Voice-controlled smart devices (lights, thermostats, TVs) use speech recognition to offer **convenience and automation**.
    - Commands like "Turn off the lights" or "Play music" are understood and executed.
8. **Automotive Industry:**
    - Speech recognition enables **hands-free calling**, **GPS navigation**, and **music control** in cars.
    - It enhances driver safety by reducing manual interactions.
9. **Education:**
    - Used in **online learning platforms** to support voice input for answering questions and interacting with educational content.
    - Helps students with learning disabilities participate more effectively.
10. **Security and Authentication:**
    - Some systems use **voice biometrics** to identify or authenticate users based on their voice patterns.
    - It adds an extra layer of security in banking and mobile applications.

5.Image processing

Image processing in AI refers to the use of artificial intelligence techniques—particularly machine learning (ML) and deep learning (DL)—to interpret, analyze, enhance, and transform images. It is a core field in computer vision and has wide-ranging applications across industries like healthcare, automotive, surveillance, retail, and more.

Concepts in Image Processing with AI

1. **Image Classification**
   AI models identify the main subject or category in an image (e.g., cat, dog, car).
    - Example: ResNet, VGG, EfficientNet.

2. **Object Detection**

   The model detects and localizes multiple objects within an image using bounding boxes.

   - Example: YOLO (You Only Look Once), SSD (Single Shot Detector), Faster R-CNN.

3. **Image Segmentation**

   Divides an image into meaningful segments (pixels grouped by class).

   - Types:
     - *Semantic Segmentation*: Classifies each pixel (e.g., "road", "sky").
     - *Instance Segmentation*: Separates different instances of the same class (e.g., two people).
   - Example: U-Net, Mask R-CNN.

4. **Image Generation**

   AI creates new images from data using models like:

   - GANs (Generative Adversarial Networks)
   - VAEs (Variational Autoencoders)
   - Diffusion models (e.g., DALL·E, Stable Diffusion)

5. **Image Enhancement and Restoration**

   AI improves image quality (e.g., denoising, super-resolution, deblurring).

   - Example: ESRGAN (Enhanced Super-Resolution GAN)

6. **Facial Recognition & Emotion Detection**

   AI recognizes and matches faces or even interprets expressions and emotions.

7. **Optical Character Recognition (OCR)**

   Converts text in images (e.g., documents, street signs) into machine-readable text.

   - Tools: Tesseract, Google Vision API.

How AI Processes Images

1. **Pre-processing**
   - Resize, normalize, grayscale/color adjust
   - Data augmentation (rotate, flip, crop, etc.)
2. **Model Training**
   - Typically uses Convolutional Neural Networks (CNNs) or Transformer-based models
   - Trained on labeled datasets like ImageNet, COCO, MNIST
3. **Feature Extraction**
   - Filters detect edges, shapes, textures, etc.
   - Layers build hierarchical understanding from pixels to patterns
4. **Inference**
   - AI applies learned patterns to new, unseen images to classify or analyze them

Popular Libraries and Frameworks

- **OpenCV**: Traditional image processing
- **TensorFlow/Keras** and **PyTorch**: Deep learning with CNNs, transformers
- **scikit-image**: Python image processing for simpler tasks
- **Detectron2, MMDetection**: Advanced object detection

Early Image Processing in AI:

Early image processing in AI refers to the period before deep learning (pre-2010s), when techniques were largely **rule-based**, **algorithmic**, and involved **handcrafted features**. This era laid the foundation for modern computer vision and AI-driven image processing.

Characteristics of Early Image Processing

1. **Rule-Based Systems**
   - Used manually coded algorithms based on mathematical models.
   - No learning from data—developers specified "if-then" logic for tasks.
   - Example: Edge detection using the **Sobel** or **Canny** filter.
2. **Handcrafted Features**
   - Focused on identifying low-level features: edges, corners, textures, blobs.
   - Popular techniques:
     - **SIFT** (Scale-Invariant Feature Transform)
     - **SURF** (Speeded-Up Robust Features)
     - **HOG** (Histogram of Oriented Gradients)
3. **Classical Machine Learning (Pre-Deep Learning)**
   - Feature vectors were extracted from images and fed into classifiers like:
     - **SVM (Support Vector Machines)**
     - **k-NN (k-Nearest Neighbors)**
     - **Decision Trees**
   - Example: Object classification using HOG + SVM.
4. **Segmentation and Thresholding**
   - Used for separating objects from the background.
   - Techniques:
     - **Otsu's Thresholding**
     - **Watershed Algorithm**
     - **Region Growing**
5. **Morphological Operations**
   - Useful for shape and structure analysis.
   - Techniques include **erosion**, **dilation**, **opening**, and **closing**.

6. **Template Matching**
   - Sliding a small image (template) across a larger one to find matches.
   - Used in OCR and simple object detection.

1. Applications of image processing in Artificial Intelligence (AI)

2. **Facial Recognition:**
   AI systems use image processing to identify and verify individuals by analyzing facial features. It is widely used in security, access control, and social media tagging.
3. **Medical Imaging:**
   Image processing assists AI in interpreting medical images like X-rays, MRIs, and CT scans to detect diseases such as cancer, tumors, or fractures, improving diagnostic accuracy.
4. **Autonomous Vehicles:**
   AI-driven self-driving cars use image processing to interpret surroundings by recognizing lanes, traffic signs, pedestrians, and obstacles, enabling safe navigation.
5. **Object Detection and Classification:**
   AI models use image processing to detect and classify objects in images or videos, useful in surveillance, robotics, and retail (e.g., inventory management).
6. **Augmented Reality (AR) and Virtual Reality (VR):**
   Image processing allows AI to overlay digital content onto real-world scenes in AR or create immersive environments in VR for gaming, training, or education.
7. **Optical Character Recognition (OCR):**
   AI uses image processing to convert scanned documents or handwritten text into machine-readable text, enhancing document digitization and automation.
8. **Image Enhancement and Restoration:**
   AI applies image processing to improve image quality by removing noise, correcting colors, and restoring damaged images for better analysis.
9. **Gesture Recognition:**
   Image processing enables AI to recognize human gestures from images or video, useful in human-computer interaction, gaming, and sign language interpretation.
10. **Satellite and Aerial Image Analysis:**
    AI analyzes satellite images for applications in agriculture, environmental monitoring, urban planning, and disaster management.

11. **Security and Surveillance:**
    Image processing helps AI monitor video feeds for unusual activities, intrusions, or crowd management, enhancing public safety.

6.Computer vision

Computer Vision is a branch of Artificial Intelligence that enables machines to interpret, analyze, and understand images and videos from the real world.

- It mimics human vision by processing visual data to make decisions or take actions.
- Using techniques like image processing, pattern recognition, and machine learning, computer vision helps computers "see" and extract meaningful information.

Real-World Example: Self-Driving Cars

Self-driving cars are a prominent example of computer vision in action. These cars are equipped with cameras and sensors that continuously capture the surrounding environment. Computer vision algorithms process this visual data to:

- Detect and recognize traffic signs, traffic lights, and road markings
- Identify pedestrians, cyclists, and other vehicles
- Understand the vehicle's position on the road
- Detect obstacles and hazards in real time

By analysing this information, the AI system can make decisions such as when to stop, accelerate, or steer, enabling the car to navigate safely without human intervention.

Computer Vision Work

Computer vision enables machines to understand and interpret visual information from images or videos. The process involves several key steps, combining image processing, machine learning, and AI algorithms:

1. **Image Acquisition**

- The first step is capturing an image or video using cameras, sensors, or other devices.
- The raw visual data is collected for further processing.

2. **Pre-processing**

- The raw images are often noisy or inconsistent, so pre-processing improves their quality.
- Common pre-processing tasks include:
  - **Noise reduction:** Removing unwanted distortions.
  - **Normalization:** Adjusting brightness and contrast.
  - **Resizing:** Changing image size for consistent input.
  - **Filtering:** Enhancing edges or important features

3. Feature Extraction

- The system identifies important characteristics or patterns in the image, such as edges, corners, textures, or shapes.
- Traditional methods include techniques like:
  - **Edge detection** (e.g., Canny edge detector)
  - **SIFT** (Scale-Invariant Feature Transform)
  - **HOG** (Histogram of Oriented Gradients)
- Modern computer vision often uses deep learning to automatically learn features from data.

4.Object Detection and Recognition

- The system locates and classifies objects within the image.
- Techniques include:
  - **Classification:** Assigning a label to an entire image (e.g., "dog" or "car").
  - **Detection:** Finding where objects are in the image (bounding boxes).
  - **Segmentation:** Dividing the image into meaningful parts (pixels belonging to objects).
- Deep learning models like **Convolutional Neural Networks (CNNs)** are widely used for this task.

5. **Post-Processing**
   - The output from detection and recognition is refined.
   - Examples include removing false positives, combining overlapping detections, or tracking objects over time in videos.
6. **Decision Making and Action**
   - Finally, the interpreted visual data is used to make decisions or trigger actions.
   - For example, in self-driving cars, the detected objects and road conditions guide navigation commands.

6.1 Applications/Tasks Associated with Computer Vision in AI

Computer vision involves various tasks that enable machines to interpret and understand visual data effectively. The main tasks include:

1. **Image Classification:**
   This task involves categorizing an entire image into a specific class or label. For example, determining whether an image contains a cat, dog, or car.
2. **Object Detection:**
   Beyond classification, object detection locates and identifies multiple objects within an image by drawing bounding boxes around them, such as detecting all the cars and pedestrians in a street scene.
3. **Image Segmentation:**
   Image segmentation divides an image into meaningful regions or segments, assigning a label to each pixel. This helps in detailed analysis like separating a person from the background or segmenting different organs in medical images.
4. **Face Recognition:**
   This task identifies or verifies individuals by analyzing their facial features. It is widely used in security, authentication systems, and social media.
5. **Optical Character Recognition (OCR):**
   OCR converts text in images—such as scanned documents or street signs—into machine-readable and editable text.
6. **Pose Estimation:**
   Pose estimation determines the position and orientation of objects or humans in images or videos, useful in gesture recognition and sports analytics.
7. **Image Enhancement:**
   Techniques under this task improve the quality of images by reducing noise, correcting colors, and sharpening features to make analysis easier.
8. **3D Reconstruction:**
   From 2D images or videos, this task creates 3D models of objects or environments, enabling applications in virtual reality and robotics.

Difference between Image Processing and Computer Vision

Aspect	Image Processing	Computer Vision
Definition	Image processing involves enhancing or manipulating images.	Computer vision focuses on enabling machines to understand and interpret images.
Primary Goal	To improve image quality or extract low-level features.	To gain high-level understanding and make decisions based on visual data.

Input and Output	Input and output are both images (e.g., a filtered or resized image).	Input is an image; output is meaningful information (e.g., object labels, positions).
Techniques Used	Filtering, enhancement, transformation, compression.	Object detection, classification, segmentation, recognition.
Level of Analysis	Low-level (pixel-based operations).	High-level (semantic understanding).
Example Tasks	Noise removal, image sharpening, contrast adjustment.	Face recognition, self-driving vision, medical diagnosis.
Involves AI/ML	Not necessarily. Many techniques are rule-based.	Heavily involves AI and machine learning, especially deep learning.
Applications	Medical imaging, photography, satellite image enhancement.	Autonomous vehicles, surveillance, augmented reality.

7. ROBOTICS: **Robotics in AI** refers to the integration of **Artificial Intelligence (AI)** technologies into **robotic systems** to enable them to perform tasks autonomously or semi-autonomously.

Robotics in AI is the branch of artificial intelligence that deals with the design, development, and operation of robots that can perceive their environment, reason, make decisions, and take actions to achieve specific goals, often without human intervention.

Key Components:

1. **Perception** – Using sensors (like cameras, microphones, or LIDAR) to gather data about the environment.
2. **Processing/Reasoning** – Applying AI algorithms to interpret the sensory data and make decisions.
3. **Action/Control** – Directing actuators or motors to perform physical tasks (e.g., moving, grasping).
4. **Learning** – Using machine learning to improve performance over time based on experience.

Examples:

- Self-driving cars (autonomous navigation and decision-making)
- Industrial robots (smart assembly lines)
- Service robots (like delivery or cleaning robots)
- Humanoid robots (social interaction, personal assistance)

Robotic Hardware in AI

Robotic hardware refers to the **physical components** that make up a robot. In the context of **AI-powered robotics**, this hardware is essential for interacting with the real world and enabling intelligent behaviour through sensors, actuators, and processing units. Here's a detailed breakdown:

1. Sensors

Sensors are responsible for **perception**—gathering data from the robot's environment and its own internal state.

a. Vision Sensors

- **Cameras (2D and 3D)**: Capture images or video for object recognition, navigation, and tracking.
- **Depth Sensors (e.g., LiDAR, stereo cameras)**: Measure distance to objects, used in obstacle detection and mapping.

b. Proximity and Distance Sensors

- **Ultrasonic Sensors**: Emit sound waves to detect nearby objects.
- **Infrared (IR) Sensors**: Detect heat or measure short-range distance.

c. Tactile and Force Sensors

- Detect physical contact, pressure, or force. Useful in robotic arms and grippers.

d. Gyroscopes and Accelerometers

- Measure orientation, tilt, and acceleration. Essential for balance and movement control (e.g., in drones or bipedal robots).

e. Environmental Sensors

- **Temperature, humidity, gas detectors**, etc., for robots operating in specific environments (e.g., agriculture or hazardous zones).

2. Actuators

Actuators are devices that **convert electrical signals into mechanical movement**.

a. Electric Motors

- Most common actuators in robots.
- **DC motors**: Used for simple movement.
- **Servo motors**: Provide precise control over angle and position.
- **Stepper motors**: Ideal for accurate positioning.

b. Hydraulic and Pneumatic Actuators

- Use fluid or air pressure for powerful movements.
- Often used in heavy-duty industrial robots.

c. Linear Actuators

- Provide straight-line motion (as opposed to rotational).

d. Piezoelectric Actuators

- Used for micro-movements in precision robotics, such as in medical or microelectronics fields.

3. Manipulators and End Effectors

These are **mechanical limbs or tools** that interact with the environment.

a. Robotic Arms

- Have multiple joints and degrees of freedom.
- Mimic human arm functionality for tasks like assembly, welding, or surgery.

b. Grippers

- Simple end-effectors for picking and placing objects.
- Types include two-finger, suction cups, or magnetic grippers.

c. Specialized Tools

- Tools designed for specific applications (e.g., scalpels in surgical robots or drills in construction robots).

4. Power Supply

Provides the energy required for the robot to operate.

a. Batteries

- Lithium-ion batteries are common in mobile robots and drones.

b. Power Cords

- Used in stationary robots for continuous operation.

c. Solar Panels

- Occasionally used in outdoor or autonomous exploratory robots.

5. Control Systems and Embedded Electronics

These are the **"brains"** of the robotic hardware, often integrated with AI systems.

a. Microcontrollers and Microprocessors

- Handle real-time control of motors and sensors.
- Examples: Arduino, Raspberry Pi, STM32.

b. Computational Units (CPUs, GPUs, TPUs)

- Used for AI processing, such as computer vision and neural networks.
- Common in advanced robots like autonomous vehicles or humanoids.

c. Communication Interfaces

- Enable internal and external data transfer.
- Wi-Fi, Bluetooth, CAN bus, USB, Ethernet, etc.

7.1Robotic movement

Robotic movement in Artificial Intelligence refers to the ability of robots to perceive their environment and move within it intelligently and autonomously. This movement is enabled through a combination of mechanical design, sensor systems, and AI algorithms that allow robots to plan, decide, and execute physical actions.

1. Components Involved in Robotic Movement

a) Actuators and Motors:

These are mechanical components (like servos, stepper motors, or hydraulic actuators) that enable the robot to move. They convert electrical signals into physical motion.

b) Sensors:

Robots use sensors (such as cameras, LiDAR, sonar, gyroscopes, and encoders) to perceive their surroundings and monitor their own position, orientation, and movement.

c) Controllers:

Microcontrollers or embedded systems interpret sensor data and send commands to actuators. These serve as the brain of the robot's physical control.

2. Role of AI in Robotic Movement

a) Motion Planning:

AI algorithms plan a robot's movement by determining the optimal path from one point to another, avoiding obstacles. Common algorithms include:

- A* (A-star)

- Dijkstra's algorithm
- RRT (Rapidly-exploring Random Trees)

b) Perception and Object Recognition:

AI allows robots to identify objects and understand spatial relationships using computer vision and machine learning, enhancing the accuracy of movement.

c) Localization and Mapping (SLAM):

Simultaneous Localization and Mapping enables robots to build a map of an unknown environment while tracking their location. SLAM is crucial for autonomous movement.

d) Feedback and Adaptation:

AI systems use feedback loops to correct movement errors in real time. Reinforcement learning helps robots improve movement through trial and error.

3. Types of Robotic Movement

a) Wheeled Movement: Common in mobile robots, this involves coordinated control of wheel motors for direction and speed.

b) Legged Movement: Found in humanoid or quadruped robots (like Boston Dynamics' Spot), requiring complex balance and gait algorithms.

c) Arm Movement (Manipulation): Used in robotic arms for tasks like picking, placing, and assembling. It involves kinematics, inverse kinematics, and trajectory planning.

7.2 Robotic Perception in Artificial Intelligence

Robotic perception refers to the robot's ability to acquire, process, and interpret sensory information from the environment to make informed decisions and perform tasks autonomously. It acts as the "eyes, ears, and skin" of a robot, enabling it to interact meaningfully with the physical world.

- Perception in AI-powered robots allows them to sense their surroundings, recognize objects, determine locations, and understand environmental context.
- Without perception, robots would be blind and unable to operate intelligently or safely in dynamic environments.

Types of Sensors in Robotic Perception

Robotic perception relies on different types of sensors:

a) Vision Sensors (Cameras):

Used for capturing images and video. Combined with computer vision and deep learning for object detection, tracking, facial recognition, etc.

b) Depth Sensors (LiDAR, Stereo Cameras, ToF Sensors):

Help determine distance and depth, essential for 3D perception and navigation.

c) Proximity Sensors (Ultrasonic, Infrared):

Used to detect nearby obstacles and support collision avoidance.

d) Tactile Sensors:

Mimic the sense of touch. Used in robotic hands to feel texture, pressure, and force.

e) Inertial Measurement Units (IMUs):

Provide orientation, acceleration, and rotational data, helping in balance and movement tracking.

f) Microphones and Audio Sensors:

Allow sound recognition, voice command processing, and audio source localization.

3. Role of AI in Robotic Perception

AI techniques enable the processing and interpretation of sensor data into meaningful information:

a) Computer Vision:

Deep learning models (e.g., CNNs) interpret images for tasks like object recognition, gesture detection, and scene understanding.

b) Sensor Fusion:

Combines data from multiple sensors (e.g., camera + LiDAR) to provide accurate and robust environmental understanding.

c) Machine Learning:

Used for pattern recognition, such as identifying objects, predicting human actions, or classifying terrain.

d) Natural Language Processing (NLP):

Enables robots to understand and respond to voice commands or spoken questions.

7.3 Robotics Applications:

1. Industrial Automation

a) Manufacturing & Assembly Lines:

AI-powered robotic arms perform repetitive tasks like welding, painting, assembling, and packaging with high precision and speed.

b) Predictive Maintenance:

Robots use AI to analyze equipment conditions and predict failures before they occur, reducing downtime.

2. Healthcare and Medical Robotics

a) Surgical Robots:

AI-assisted robots (e.g., da Vinci Surgical System) help in minimally invasive surgeries with enhanced precision and control.

b) Rehabilitation and Elderly Care:

Robots support patient recovery through physical therapy or assist elderly individuals with daily tasks.

c) Diagnosis and Monitoring:

AI robots monitor vital signs, deliver medications, and assist in diagnostic imaging using real-time data.

3. Service and Domestic Robots

a) Cleaning Robots:

Robots like robotic vacuum cleaners (e.g., Roomba) use AI to navigate homes and clean efficiently.

b) Personal Assistants:

AI-powered humanoid robots help with tasks like scheduling, answering questions, and controlling smart home devices.

4. Autonomous Vehicles and Drones

a) Self-driving Cars:
AI enables robotic vehicles to detect obstacles, recognize traffic signs, and make driving decisions.

b) Delivery Drones:
Used by companies like Amazon and Zipline to deliver goods and medical supplies using AI for navigation and route optimization.

5. Defense and Security

a) Bomb Disposal Robots:
AI robots are deployed in hazardous zones to defuse explosives and handle dangerous materials.

b) Surveillance and Patrol:
Autonomous robots equipped with cameras and AI monitor sensitive areas, detect threats, and raise alerts.

6. Agriculture

a) Precision Farming:
AI robots perform tasks like planting, watering, spraying pesticides, and monitoring crop health using data from sensors and cameras.

b) Harvesting Robots:
Use AI to identify ripe fruits/vegetables and pick them without damaging the crop.

7. Space Exploration

a) Mars Rovers (e.g., Perseverance):
AI robots explore planetary surfaces, collect samples, and send data back to Earth.

b) Autonomous Navigation in Space:
Robots assist astronauts and operate in space environments where human presence is limited.

8. Education and Research

a) Teaching Aids:
AI robots are used as interactive tools to assist students in learning, especially in STEM subjects.

b) Research Assistants:
Robots are employed in labs to conduct repetitive experiments, data collection, and analysis.

MODULE-05 APPLICATIONS OF AI

5 MARKS QUESTION

1.Define language. Explain language models in detail.

2.Explain Applications of NLP in AI.

3.Write a short note on text classification

4.Define information retrieval. Explain Application of Information retrieval.

5.Define Speech recognition. Explain Application of Speech recognition.

6.Define image processing. Explain Application of Image processing.

7.Define computer vision. Explain Application of Image processing.

8. Short note on task associated with computer vision

9. Define Robotics.Explain Applicatons of Robotics.

10.Explain Robotic Hardwares in detail

10 MARKS QUESTION

1.what is Natural Language processing. Explain in detail

a) N-gram Character models

b) smoothing N-gram Models.

c)N-gram word models.

2.Explain Application text classification in AI.

3.what is an information retrieval. Write a short note
a)IR Scoring functions.

b)IR system Evaluation.

c)IR refinements.

d)Question Answering.

4.Explain a)Page Rank algorithm.

b) The HITS algorithm.

5. What is Speech recognition. Explain Acoustic Model and language model & Building a Speech recognizer.

6. what image processing.Explain early image processing.

7.How does Computer vision Work in detail.

8.Difference between Image Processing and Computer vision.

9.Explain Robotic movement in detail.

10.Discuss Robotic perceptions in detail.

**************************************END*******************************