**DS Lab**

**Exp-2**

**Name: Piyush Tilokani**

**Division: D20A**

**Roll no: 63**

**Aim:** To build a Cognitive text based application to understand context for a Customer

service application/ Insurance/ Healthcare Application/ Smarter Cities/ Government

etc.

**Theory:**

**Cognitive Computing**

Cognitive computing is the process of replicating human thought patterns within a computerized system. These systems are self-learning, leveraging techniques like data mining, pattern recognition, and natural language processing to imitate the workings of the human brain. By analyzing vast amounts of unstructured data—such as text, images, or videos—cognitive systems can generate insights and make informed decisions or predictions.

**Context Understanding**

Understanding context is essential for cognitive applications. It enables the system to comprehend not only the literal meaning of words but also the context in which they are used. This involves:

User Intent: Identifying what the user aims to accomplish.

Entity Recognition: Detecting important entities such as names, dates, locations, or domain-specific terms.

Sentiment Analysis: Assessing the emotional tone of the conversation.

Conversation Flow: Monitoring the progression of the dialogue to ensure coherence and relevance throughout the interaction.

**Key Components for Building Cognitive Applications**

**Natural Language Processing (NLP):**

Tokenization: Dividing text into individual words or phrases.

Part-of-Speech Tagging: Identifying the grammatical roles of words within a sentence.

Named Entity Recognition (NER): Extracting specific entities such as names, locations, and dates.

Sentiment Analysis: Assessing the emotional tone of the text, whether positive, negative, or neutral.

Text Classification: Assigning text to predefined categories, such as distinguishing between spam and non-spam.

**Machine Learning:**

Supervised Learning: Training models with labeled data to predict outcomes.

Unsupervised Learning: Finding patterns in data without predefined labels.

Reinforcement Learning: Models learn to make decisions by receiving feedback (rewards or penalties).

**Domain-Specific Knowledge:**

Customer Service: Understanding common customer inquiries and troubleshooting processes.

Insurance: Recognizing insurance terminology and processing claims.

Healthcare: Interpreting medical terminology and patient records.

Smarter Cities: Integrating data from various urban systems for efficient city management.

Government: Processing and responding to public inquiries, legal documents, and policy information.

**Code:**

from transformers import pipeline

import spacy

from google.colab import files

# Load spaCy model

nlp = spacy.load('en\_core\_web\_sm')

# Initialize pipelines

summarizer = pipeline('summarization')

qa\_pipeline = pipeline('question-answering')

# Upload text file

uploaded = files.upload()

# Read and analyze the lesson plan

file\_name = list(uploaded.keys())[0]

with open(file\_name, 'r') as file:

lesson\_text = file.read()

# Process the text with spaCy

doc = nlp(lesson\_text)

key\_concepts = [chunk.text for chunk in doc.noun\_chunks if chunk.root.dep\_ == "nsubj"]

# Summarize the lesson

summary = summarizer(lesson\_text, max\_length=50, min\_length=25, do\_sample=False)

summary\_text = summary[0]['summary\_text']

print("Key Concepts:", key\_concepts)

print("Summary:", summary\_text)

# Function to answer questions based on the text

def answer\_question(question):

result = qa\_pipeline(question=question, context=lesson\_text)

return result['answer']

# Example usage

while True:

user\_question = input("Ask a question related to the text (or type 'exit' to quit): ")

if user\_question.lower() == 'exit':

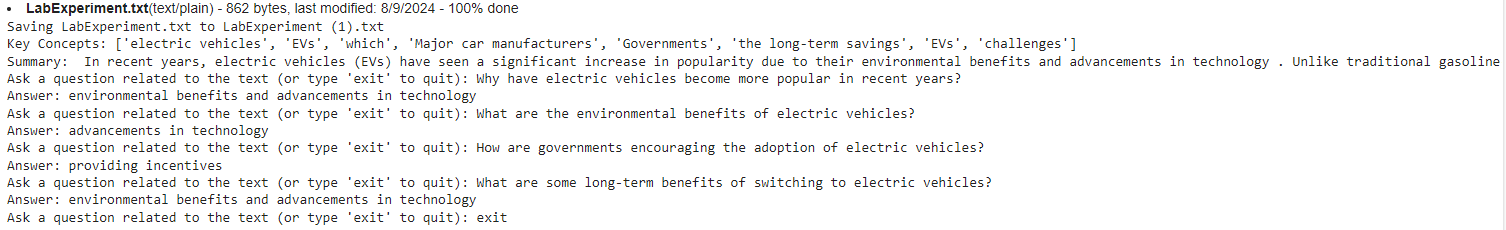
break

answer = answer\_question(user\_question)

print("Answer:", answer)

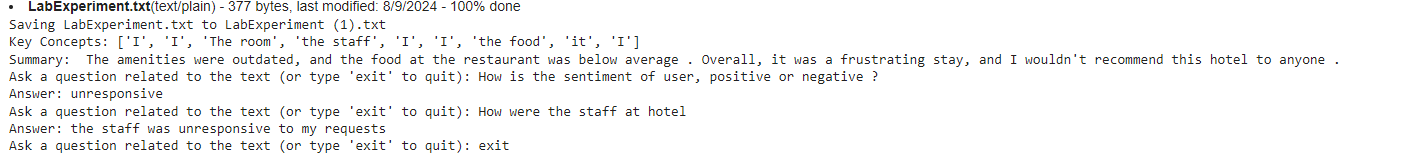
**Output 1 :**

Input 1 was a paragraph about Electric Vehicles. The model understood the context and answered all the questions correctly.



**Output 2 :**

Input 2 was a paragraph to check the sentiment analysis of model. The output shows that the model works perfectly fine.



**Conclusion:** As a result, cognitive text-based applications signify a significant advancement toward creating more intuitive and effective interactions between humans and machines.