Experiment 4:

Aim: Use Word Embedding's to improve prompts for Generative AI model.

(i) Retrieve similar words using word embedding's.

(ii) Use the similar words to enrich a GenAI prompt.

(iii) Use the AI model to generate responses for the original and enriched prompts.

(iv) Compare the outputs in terms of detail and relevance.

Step 1: Installs necessary libraries**,** imports modules**, and** sets up Google Colab environment **to work with** word embedding's **using Genism.**

```
%pip install numpy
%pip install scipy
%pip install gensim
import os
import gensim.downloader as api
from gensim.models import KeyedVectors
from google.colab import drive
```

Step 2: Connect your Google Drive to Google Colab**,** allowing you to access, save, and load files from your Drive directly in the Colab environment.

```
drive.mount('/content/drive')
```

Step 3: Check if Word2Vec model is already saved in Google Drive**,** load it if found**, or** downloads and save it if not available**.**

```
model_path = "/content/drive/My Drive/word2vec-google-news-300.model"

if os.path.exists(model_path):
  print("Model found in Google Drive..Loading")
  word_vectors = KeyedVectors.load(model_path)
  print("Loading Completed")
else:
  print("Model not found. Downloading Word2Vec model...")
  word_vectors = api.load("word2vec-google-news-300")
  print("Saving model to Google Drive for future use...")
  word_vectors.save(model_path)
  print("Model saved successfully")
  print("\nModel Loaded Successfully\n")
```

Step 4: Retrieve words that are most similar **to "king"** based on the pre-trained Word2Vec model**. It measures similarity using** cosine similarity **between word vectors.**

```
print(word_vectors.most_similar("king"))
```

Output: *[('kings', 0.7138045430183411), ('queen', 0.6510956883430481), ('monarch', 0.6413194537162781), ('crown_prince', 0.6204220056533813), ('prince', 0.6159993410110474), ('sultan', 0.5864824056625366), ('ruler', 0.5797567367553711), ('princes', 0.5646552443504333), ('Prince_Paras', 0.5432944297790527), ('throne', 0.5422105193138123)]*

Step 5: Take user input to create a structured prompt for an AI system.

1. The original prompt (a question or instruction for AI).
2. Key terms (words that help refine the AI response).

```
original_prompt = input("Enter the original prompt: ")

# Get key terms from user (comma-separated)
key_terms_input = input("Enter key terms (comma-separated): ")
key_terms = [term.strip() for term in key_terms_input.split(",")]
```

Output:

***Enter the original prompt**: Importance of engineering*

***Enter key terms (comma-separated):** Job, career*

Step 6: Enrich the given user prompt **by finding** semantically similar words **for key terms using** pre-trained word embedding's (Word2Vec)**.**

```
similar_terms = []

for term in key_terms:
    if term in  word_vectors.key_to_index:
        similar_terms.extend({word for word, _ in word_vectors.most_similar(term, topn=2)})

if similar_terms:
    enriched_prompt = f"{original_prompt} Consider aspects like: {', '.join(similar_terms)}."
else:
    enriched_prompt = original_prompt

print("Original Prompt:", original_prompt)
print("Enriched Prompt:", enriched_prompt)
```

Output:

Original Prompt: *Importance of engineering.*

Enriched Prompt: *Importance of engineering Consider aspects like: Employment, job, career.*

Step 7: Prompt the user **to enter the** API key (API key of Google AI used here)**, stores it in an** environment variable**, and assigns it to a Python variable for later use.**

Note: Create the Google API using the link, **https://aistudio.google.com/**

```
import getpass
import os

GOOGLE_API_KEY= os.environ["GOOGLE_API_KEY"] = getpass.getpass("Enter your Google AI API key: ")
```

Step 8: Install the necessary LangChain libraries and set up **ChatGoogleGenerativeAI** to interact with **Google's Gemini-2.0 Flash model** for text generation**. Import the required module, initialize the model with key parameters like** temperature (0.3 for balanced responses), max tokens (512 for response length), timeout (30 seconds to prevent hanging requests), and max retries (2 for handling failures)**.**
The api_key ensures secure access to Google's AI services, enabling efficient and controlled AI-powered text generation.

```
%pip install langchain-google-genai
%pip install langchain-core
%pip install langchain-community
%pip install -qU langchain-google-genai
%pip install --upgrade langchain

from langchain_google_genai import ChatGoogleGenerativeAI

llm = ChatGoogleGenerativeAI(
    model="gemini-2.0-flash-exp",
    temperature=0.3,
    api_key=GOOGLE_API_KEY,
    max_tokens=512,
    timeout=30,
    max_retries=2,
)
```

Step 8: Invoking the AI Model for a Response

```
llm.invoke("Hi")
```

The command llm.invoke("Hi") sends the input "Hi" to the Google Gemini AI model via LangChain and returns the model's response. Since llm is an instance of ChatGoogleGenerativeAI, this method invokes the AI model, processes the input, and generates a response based on the configured parameters (e.g., temperature, max tokens, timeout). The output will typically be a greeting or a conversational response like

"Hi! How can I assist you today?"

Step 9: Send the original prompt to the Google Gemini AI model (Large Language Model)

```
print(llm.invoke(original_prompt).content)
```

This command **sends the** original_prompt to the Google Gemini AI model (llm), **retrieves the response**, extracts the **text content**, and prints it.

Output: Engineering is incredibly important to modern society, playing a vital role in shaping our world and improving our lives in countless ways.

Here's a breakdown of its importance:

1. **Solving Problems and Improving Life:**

* **Addressing Global Challenges:** Engineers are at the forefront of tackling some of the world's most pressing issues, such as climate change, food security, water scarcity, and disease. They develop innovative solutions for renewable energy, sustainable agriculture, water purification, and medical technologies.

* **Improving Quality of Life:** Engineering advancements have significantly improved our quality of life. From transportation and communication to healthcare and entertainment, engineers design and build the systems and technologies that make our lives easier, safer, and more enjoyable.

* **Infrastructure Development:** Engineers design, construct, and maintain the infrastructure that supports our society, including roads, bridges, buildings, power grids, and water systems. This infrastructure is essential for economic growth, transportation, and public safety.

Step 10: Send the enriched prompt to the Google Gemini AI model.

```
print(llm.invoke(enriched_prompt).content)
```

This command **sends an enriched version of the prompt** (enriched_prompt) to the **Google Gemini AI model**, retrieves its response, extracts the **text content**, and prints it.

Output: ## The Importance of Engineering: A Foundation for Progress

Engineering is a vital field that underpins much of modern society. Its importance spans numerous aspects, impacting everything from our daily lives to global progress. Let's explore its significance, focusing on employment, job, career, and overall impact:

****1. Employment:****

* **High Demand:** Engineering fields consistently experience high demand globally. This is driven by technological advancements, infrastructure development, and the need for innovative solutions to complex problems.
* **Diverse Opportunities:** Engineering encompasses a wide range of disciplines (civil, mechanical, electrical, chemical, computer, aerospace, etc.), each offering specialized employment opportunities.
* **Recession-Resistant:** While no field is completely immune to economic downturns, engineering tends to be more resilient than many others. The need for infrastructure maintenance, technological upgrades, and problem-solving persists even during challenging economic times.