```
# STEP 2: Import Required Libraries

# Install gensim if not already installed
!pip install gensim

# gensim → to load pre-trained word embeddings
import gensim.downloader as api

# numpy → to handle numerical arrays
import numpy as np

# matplotlib → to visualize embeddings
import matplotlib.pyplot as plt

# sklearn → for t-SNE dimensionality reduction
from sklearn.manifold import TSNE
```

```
Collecting gensim
  Downloading gensim-4.4.0-cp312-cp312-manylinux_2_24_x86_64.manylinux_2_28_x86_64.whl.metadata (8
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.12/dist-packages (from gens
Requirement already satisfied: scipy>=1.7.0 in /usr/local/lib/python3.12/dist-packages (from gensi
Requirement already satisfied: smart_open>=1.8.1 in /usr/local/lib/python3.12/dist-packages (from
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from smart_open>=
Downloading gensim-4.4.0-cp312-cp312-manylinux_2_24_x86_64.manylinux_2_28_x86_64.whl (27.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 27.9/27.9 MB 45.1 MB/s eta 0:00:00
Installing collected packages: gensim
Successfully installed gensim-4.4.0
```

```
# STEP 3: Load Pre-trained Model

print("Loading model...")

model = api.load("glove-wiki-gigaword-100")  # 100-dimensional vectors

print("Model loaded successfully!")

# Print vocabulary size
print("Vocabulary size:", len(model.key_to_index))

# Display one example vector
print("Vector for word 'computer':")
print(model['computer'])
```

```
Loading model...
[==================================================] 100.0% 128.1/128.1MB downloaded
Model loaded successfully!
Vocabulary size: 400000
Vector for word 'computer':
[-1.6298e-01  3.0141e-01  5.7978e-01  6.6548e-02  4.5835e-01 -1.5329e-01
  4.3258e-01 -8.9215e-01  5.7747e-01  3.6375e-01  5.6524e-01 -5.6281e-01
  3.5659e-01 -3.6096e-01 -9.9662e-02  5.2753e-01  3.8839e-01  9.6185e-01
  1.8841e-01  3.0741e-01 -8.7842e-01 -3.2442e-01  1.1202e+00  7.5126e-02
  4.2661e-01 -6.0651e-01 -1.3893e-01  4.7862e-02 -4.5158e-01  9.3723e-02
  1.7463e-01  1.0962e+00 -1.0044e+00  6.3889e-02  3.8002e-01  2.1109e-01
 -6.6247e-01 -4.0736e-01  8.9442e-01 -6.0974e-01 -1.8577e-01 -1.9913e-01
 -6.9226e-01 -3.1806e-01 -7.8565e-01  2.3831e-01  1.2992e-01  8.7721e-02
  4.3205e-01 -2.2662e-01  3.1549e-01 -3.1748e-01 -2.4632e-03  1.6615e-01
  4.2358e-01 -1.8087e+00 -3.6699e-01  2.3949e-01  2.5458e+00  3.6111e-01
  3.9486e-02  4.8607e-01 -3.6974e-01  5.7282e-02 -4.9317e-01  2.2765e-01
  7.9966e-01  2.1428e-01  6.9811e-01  1.1262e+00 -1.3526e-01  7.1972e-01
 -9.9605e-04 -2.6842e-01 -8.3038e-01  2.1780e-01  3.4355e-01  3.7731e-01
 -4.0251e-01  3.3124e-01  1.2576e+00 -2.7196e-01 -8.6093e-01  9.0053e-02
 -2.4876e+00  4.5200e-01  6.6945e-01 -5.4648e-01 -1.0324e-01 -1.6979e-01
  5.9437e-01  1.1280e+00  7.5755e-01 -5.9160e-02  1.5152e-01 -2.8388e-01
  4.9452e-01 -9.1703e-01  9.1289e-01 -3.0927e-01]
```

```
# STEP 4: Select Words

words = [
    # Animals
    "dog", "cat", "lion", "tiger", "elephant", "wolf",

    # Fruits
    "apple", "banana", "orange", "mango", "grape",

    # Countries
    "india", "china", "france", "germany", "japan",

    # Cities
    "delhi", "mumbai", "paris", "tokyo", "berlin",

    # Technology
    "computer", "laptop", "keyboard", "mouse", "internet",

    # Vehicles
    "car", "bus", "train", "airplane", "bicycle"
]

# Extract vectors
vectors = []

valid_words = []

for word in words:
    if word in model:
        vectors.append(model[word])
        valid_words.append(word)

vectors = np.array(vectors)

print("Total selected words:", len(valid_words))
print("Shape of vectors:", vectors.shape)
```

```
Total selected words: 31
Shape of vectors: (31, 100)
```

```
# STEP 5: Apply t-SNE

tsne = TSNE(n_components=2, random_state=42, perplexity=5)

reduced_vectors = tsne.fit_transform(vectors)

print("t-SNE reduction complete.")
print("New shape:", reduced_vectors.shape)
```

```
t-SNE reduction complete.
New shape: (31, 2)
```

```
# STEP 6: Plot 2D Visualization

plt.figure(figsize=(12, 8))

x = reduced_vectors[:, 0]
y = reduced_vectors[:, 1]

plt.scatter(x, y)

# Annotate words
for i, word in enumerate(valid_words):
```

```
    plt.annotate(word, (x[i], y[i]))

plt.title("t-SNE Visualization of Word Embeddings")
plt.xlabel("Dimension 1")
plt.ylabel("Dimension 2")

plt.grid(True)
plt.show()
```