

▼ Step2:import libraries

```
!pip install gensim
import gensim.downloader as api    # To load pretrained word embeddings
import numpy as np                 # For numerical vector handling
import matplotlib.pyplot as plt   # For visualization
from sklearn.manifold import TSNE # For dimensionality reduction

Requirement already satisfied: gensim in /usr/local/lib/python3.12/dist-packages (4.4.0)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.12/dist-packages (from gensim) (2.0.2)
Requirement already satisfied: scipy>=1.7.0 in /usr/local/lib/python3.12/dist-packages (from gensim) (1.16.3)
Requirement already satisfied: smart_open>=1.8.1 in /usr/local/lib/python3.12/dist-packages (from gensim) (7.5.0)
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from smart_open>=1.8.1->gensim) (2.1.1)
```

▼ Step3:load embedding model

Double-click (or enter) to edit

```
print("Loading Word2Vec model...")
model = api.load("word2vec-google-news-300")

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

# Example vector
example_word = "computer"
print("\nExample word:", example_word)
print("Vector length:", len(model[example_word]))
print("First 10 values of vector:\n", model[example_word][:10])

Loading Word2Vec model...
Vocabulary Size: 3000000

Example word: computer
Vector length: 300
First 10 values of vector:
 [ 0.10742188 -0.20117188  0.12304688  0.21191406 -0.09130859  0.21679688
 -0.13183594  0.08300781  0.20214844  0.04785156]
```

▼ STEP 4:Select Words

```
word_list = [
    # Animals
    "dog", "cat", "lion", "tiger", "elephant", "wolf", "fox", "horse",

    # Cities
    "delhi", "mumbai", "london", "paris", "tokyo", "newyork", "hyderabad",

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

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

    # Vehicles
    "car", "bus", "train", "bike", "truck", "airplane"
]

# Extract vectors
vectors = np.array([model[word] for word in word_list])
```

▼ step5: apply t-SNE

```
tsne = TSNE(n_components=2, random_state=42, perplexity=5)
reduced_vectors = tsne.fit_transform(vectors)
```

```
print("Reduced Shape:", reduced_vectors.shape)
Reduced Shape: (34, 2)
```

▼ STEP6: Plot Visualization

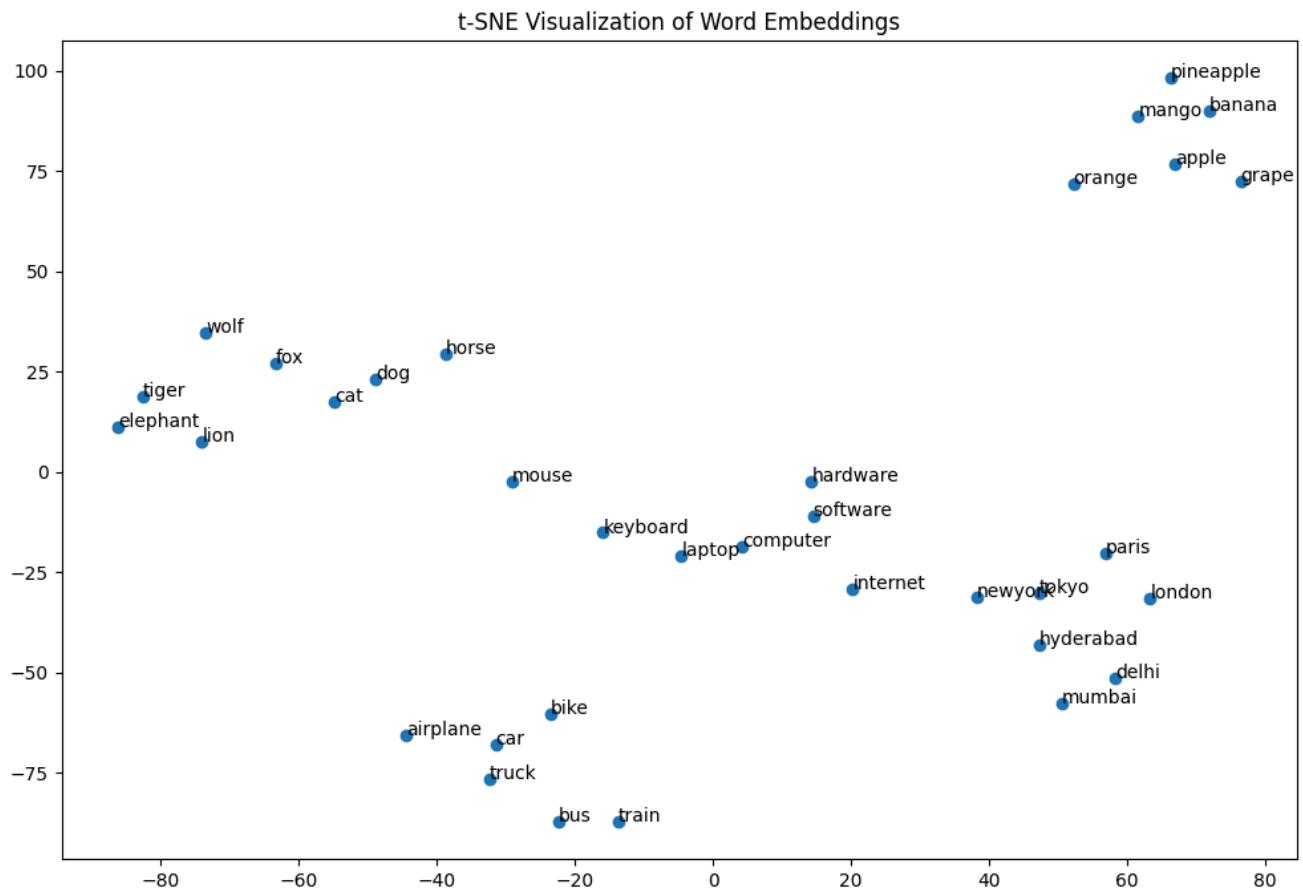
```
plt.figure(figsize=(12,8))

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

plt.scatter(x, y)

# Annotate each word
for i, word in enumerate(word_list):
    plt.annotate(word, (x[i], y[i]))

plt.title("t-SNE Visualization of Word Embeddings")
plt.show()
```



▼ STEP 7 — Interpretation (8–10 Sentences)

The visualization shows that semantically similar words form clusters. Animal names such as dog, cat, lion, and tiger appear close together indicating shared contextual meaning. Similarly, technology related words like computer, laptop, and keyboard are grouped in another region of the plot. City names such as Delhi, London, and Tokyo appear in proximity suggesting geographical similarity. Fruit names also form a separate cluster showing semantic association. Vehicle related words like car, bus, and airplane appear grouped together. Some words may appear slightly misplaced because embeddings are trained using contextual data rather than strict categories. Words with multiple meanings sometimes appear near unexpected groups. Overall, the visualization demonstrates that embedding models capture semantic relationships effectively.

STEP 8 — Lab Report Content

- ◆ Objective

The objective of this experiment is to visualize high-dimensional word embeddings using t-SNE and analyze semantic relationships between words.

- ◆ Embedding Model Description

The Google News Word2Vec pretrained model was used. It contains 300-dimensional vectors trained on large text corpora and captures semantic meaning of words.

- ◆ Word List Used

Animals, Cities, Technology terms, Fruits, and Vehicles were selected to observe semantic grouping.

- ◆ Visualization Result

The t-SNE algorithm reduced 300-dimension vectors into 2-dimension space and scatter plot visualization was generated.

- ◆ Cluster Interpretation

Words belonging to similar categories formed clusters indicating semantic similarity. Technology, animals, fruits, cities, and vehicles grouped separately.

- ◆ Conclusion

t-SNE is an effective dimensionality reduction technique that helps understand word embeddings visually. It reveals semantic relationships between words and helps evaluate embedding quality.