

FGAI LAB3

SAANVI KULKARNI 391

```
!pip install gensim scikit-learn matplotlib
from gensim.models import Word2Vec
from nltk.tokenize import word_tokenize
import nltk
nltk.download('punkt')
nltk.download('punkt_tab') # Added to download the missing resource
```

```
# Sample corpus
corpus = [
    "In the context of Generative AI, inferencing refers to the process of using a  

    trained model to make predictions or generate outputs based on new input data",
    "Prompt engineering involves several techniques, including prompt selection,  

    prompt formatting, and prompt tuning",
    "Prompt selection involves choosing the right words and phrasing to effectively  

    communicate the desired task to the model",
    "A prompt is the input given to a language model, and it can be thought of as a  

    set of instructions or a question that the model is being asked to answer"
]

# Tokenize sentences
tokenized_corpus = [word_tokenize(sentence.lower()) for sentence in corpus]
print(tokenized_corpus)

# Train Word2Vec model
model = Word2Vec(sentences=tokenized_corpus, vector_size=100, window=5,
min_count=1, workers=4)

# Save the model
model.save("word2vec.model")
model = Word2Vec.load("word2vec.model")
# Get vectors for a subset of words
words = list(model.wv.index_to_key)[:10] # Select the first 10 words
print(words)
word_vectors = [model.wv[word] for word in words]
print(word_vectors)
from sklearn.decomposition import PCA

# Apply PCA for dimensionality reduction
pca = PCA(n_components=2)
pca_result = pca.fit_transform(word_vectors)
import matplotlib.pyplot as plt

# Plot the words in 2D space
plt.figure(figsize=(10, 5))
plt.scatter(pca_result[:, 0], pca_result[:, 1])
# Annotate the points with the words
for i, word in enumerate(words):
```

```
plt.annotate(word, xy=(pca_result[i, 0], pca_result[i, 1]))

plt.title("2D Visualization of Word Embeddings")
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.grid(True)
plt.show()
```

