

```
!pip install gensim

Collecting gensim
  Downloading gensim-4.4.0-cp312-cp312-
manylinux_2_24_x86_64.manylinux_2_28_x86_64.whl.metadata (8.4 kB)
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)
  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 42.6 MB/s eta
0:00:00

Successfully installed gensim-4.4.0

# Download the model
!wget -c https://github.com/mmmihaltz/word2vec-GoogleNews-
vectors/raw/master/GoogleNews-vectors-negative300.bin.gz

--2026-02-11 06:23:47-- https://github.com/mmmihaltz/word2vec-
GoogleNews-vectors/raw/master/GoogleNews-vectors-negative300.bin.gz
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://media.githubusercontent.com/media/mmmihaltz/word2vec-
GoogleNews-vectors/master/GoogleNews-vectors-negative300.bin.gz
[following]
--2026-02-11 06:23:47--
https://media.githubusercontent.com/media/mmmihaltz/word2vec-
GoogleNews-vectors/master/GoogleNews-vectors-negative300.bin.gz
Resolving media.githubusercontent.com (media.githubusercontent.com)... 
185.199.110.133, 185.199.109.133, 185.199.111.133, ...
Connecting to media.githubusercontent.com
(media.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1647046227 (1.5G) [application/octet-stream]
Saving to: 'GoogleNews-vectors-negative300.bin.gz'

GoogleNews-vectors- 100%[=====] 1.53G 40.1MB/s in
38s

2026-02-11 06:24:26 (40.8 MB/s) - 'GoogleNews-vectors-
negative300.bin.gz' saved [1647046227/1647046227]
```

```

# gensim is used to load pre-trained Word2Vec model
from gensim.models import KeyedVectors

# numpy for numerical operations
import numpy as np

# pandas for tabular display
import pandas as pd

# matplotlib for visualization
import matplotlib.pyplot as plt

# PCA for dimensionality reduction
from sklearn.decomposition import PCA

print("Loading Google News Word2Vec model...")

model = KeyedVectors.load_word2vec_format(
    "GoogleNews-vectors-negative300.bin.gz",
    binary=True
)

print("Model Loaded Successfully!")
print("Vocabulary Size:", len(model.key_to_index))

Loading Google News Word2Vec model...
Model Loaded Successfully!
Vocabulary Size: 3000000

word = "king"

vector = model[word]

print("Vector for word:", word)
print(vector)
print("Vector dimension:", len(vector))

Vector for word: king
[ 1.25976562e-01  2.97851562e-02   8.60595703e-03   1.39648438e-01
 -2.56347656e-02 -3.61328125e-02   1.11816406e-01  -1.98242188e-01
  5.12695312e-02  3.63281250e-01  -2.42187500e-01  -3.02734375e-01
 -1.77734375e-01 -2.49023438e-02  -1.67968750e-01  -1.69921875e-01
  3.46679688e-02  5.21850586e-03   4.63867188e-02   1.28906250e-01
  1.36718750e-01  1.12792969e-01   5.95703125e-02   1.36718750e-01
  1.01074219e-01 -1.76757812e-01  -2.51953125e-01   5.98144531e-02
  3.41796875e-01 -3.11279297e-02   1.04492188e-01   6.17675781e-02
  1.24511719e-01  4.00390625e-01  -3.22265625e-01   8.39843750e-02
  3.90625000e-02  5.85937500e-03   7.03125000e-02   1.72851562e-01
  1.38671875e-01 -2.31445312e-01   2.83203125e-01   1.42578125e-01
  3.41796875e-01 -2.39257812e-02  -1.09863281e-01   3.32031250e-02
 -5.46875000e-02  1.53198242e-02  -1.62109375e-01   1.58203125e-01

```

-2.59765625e-01	2.01416016e-02	-1.63085938e-01	1.35803223e-03
-1.44531250e-01	-5.68847656e-02	4.29687500e-02	-2.46582031e-02
1.85546875e-01	4.47265625e-01	9.58251953e-03	1.31835938e-01
9.86328125e-02	-1.85546875e-01	-1.00097656e-01	-1.33789062e-01
-1.25000000e-01	2.83203125e-01	1.23046875e-01	5.32226562e-02
-1.77734375e-01	8.59375000e-02	-2.18505859e-02	2.05078125e-02
-1.39648438e-01	2.51464844e-02	1.38671875e-01	-1.05468750e-01
1.38671875e-01	8.88671875e-02	-7.51953125e-02	-2.13623047e-02
1.72851562e-01	4.63867188e-02	-2.65625000e-01	8.91113281e-03
1.49414062e-01	3.78417969e-02	2.38281250e-01	-1.24511719e-01
-2.17773438e-01	-1.81640625e-01	2.97851562e-02	5.71289062e-02
-2.89306641e-02	1.24511719e-02	9.66796875e-02	-2.31445312e-01
5.81054688e-02	6.68945312e-02	7.08007812e-02	-3.08593750e-01
-2.14843750e-01	1.45507812e-01	-4.27734375e-01	-9.39941406e-03
1.54296875e-01	-7.66601562e-02	2.89062500e-01	2.77343750e-01
-4.86373901e-04	-1.36718750e-01	3.24218750e-01	-2.46093750e-01
-3.03649902e-03	-2.11914062e-01	1.25000000e-01	2.69531250e-01
2.04101562e-01	8.25195312e-02	-2.01171875e-01	-1.60156250e-01
-3.78417969e-02	-1.20117188e-01	1.15234375e-01	-4.10156250e-02
-3.95507812e-02	-8.98437500e-02	6.34765625e-03	2.03125000e-01
1.86523438e-01	2.73437500e-01	6.29882812e-02	1.41601562e-01
-9.81445312e-02	1.38671875e-01	1.82617188e-01	1.73828125e-01
1.73828125e-01	-2.37304688e-01	1.78710938e-01	6.34765625e-02
2.36328125e-01	-2.08984375e-01	8.74023438e-02	-1.66015625e-01
-7.91015625e-02	2.43164062e-01	-8.88671875e-02	1.26953125e-01
-2.16796875e-01	-1.73828125e-01	-3.59375000e-01	-8.25195312e-02
-6.49414062e-02	5.07812500e-02	1.35742188e-01	-7.47070312e-02
-1.64062500e-01	1.15356445e-02	4.45312500e-01	-2.15820312e-01
-1.11328125e-01	-1.92382812e-01	1.70898438e-01	-1.25000000e-01
2.65502930e-03	1.92382812e-01	-1.74804688e-01	1.39648438e-01
2.92968750e-01	1.13281250e-01	5.95703125e-02	-6.39648438e-02
9.96093750e-02	-2.72216797e-02	1.96533203e-02	4.27246094e-02
-2.46093750e-01	6.39648438e-02	-2.25585938e-01	-1.68945312e-01
2.89916992e-03	8.20312500e-02	3.41796875e-01	4.32128906e-02
1.32812500e-01	1.42578125e-01	7.61718750e-02	5.98144531e-02
-1.19140625e-01	2.74658203e-03	-6.29882812e-02	-2.72216797e-02
-4.82177734e-03	-8.20312500e-02	-2.49023438e-02	-4.00390625e-01
-1.06933594e-01	4.24804688e-02	7.76367188e-02	-1.16699219e-01
7.37304688e-02	-9.22851562e-02	1.07910156e-01	1.58203125e-01
4.24804688e-02	1.26953125e-01	3.61328125e-02	2.67578125e-01
-1.01074219e-01	-3.02734375e-01	-5.76171875e-02	5.05371094e-02
5.26428223e-04	-2.07031250e-01	-1.38671875e-01	-8.97216797e-03
-2.78320312e-02	-1.41601562e-01	2.07031250e-01	-1.58203125e-01
1.27929688e-01	1.49414062e-01	-2.24609375e-02	-8.44726562e-02
1.22558594e-01	2.15820312e-01	-2.13867188e-01	-3.12500000e-01
-3.73046875e-01	4.08935547e-03	1.07421875e-01	1.06933594e-01
7.32421875e-02	8.97216797e-03	-3.88183594e-02	-1.29882812e-01
1.49414062e-01	-2.14843750e-01	-1.83868408e-03	9.91210938e-02
1.57226562e-01	-1.14257812e-01	-2.05078125e-01	9.91210938e-02

```
3.69140625e-01 -1.97265625e-01 3.54003906e-02 1.09375000e-01
1.31835938e-01 1.66992188e-01 2.35351562e-01 1.04980469e-01
-4.96093750e-01 -1.64062500e-01 -1.56250000e-01 -5.22460938e-02
1.03027344e-01 2.43164062e-01 -1.88476562e-01 5.07812500e-02
-9.37500000e-02 -6.68945312e-02 2.27050781e-02 7.61718750e-02
2.89062500e-01 3.10546875e-01 -5.37109375e-02 2.28515625e-01
2.51464844e-02 6.78710938e-02 -1.21093750e-01 -2.15820312e-01
-2.73437500e-01 -3.07617188e-02 -3.37890625e-01 1.53320312e-01
2.33398438e-01 -2.08007812e-01 3.73046875e-01 8.20312500e-02
2.51953125e-01 -7.61718750e-02 -4.66308594e-02 -2.23388672e-02
2.99072266e-02 -5.93261719e-02 -4.66918945e-03 -2.44140625e-01
-2.09960938e-01 -2.87109375e-01 -4.54101562e-02 -1.77734375e-01
-2.79296875e-01 -8.59375000e-02 9.13085938e-02 2.51953125e-01]
```

Vector dimension: 300

```
word_pairs = [
    ("doctor", "nurse"),
    ("cat", "dog"),
    ("car", "bus"),
    ("king", "queen"),
    ("apple", "banana"),
    ("teacher", "student"),
    ("paris", "france"),
    ("man", "woman"),
    ("computer", "laptop"),
    ("school", "college")
]

for w1, w2 in word_pairs:
    similarity = model.similarity(w1, w2)
    print(f"Similarity between {w1} and {w2}: {similarity:.4f}")

Similarity between doctor and nurse: 0.6320
Similarity between cat and dog: 0.7609
Similarity between car and bus: 0.4693
Similarity between king and queen: 0.6511
Similarity between apple and banana: 0.5318
Similarity between teacher and student: 0.6301
Similarity between paris and france: 0.5551
Similarity between man and woman: 0.7664
Similarity between computer and laptop: 0.6640
Similarity between school and college: 0.6082

words_to_check = ["king", "university", "money", "doctor", "india"]

for word in words_to_check:
    print(f"\nTop 5 words similar to '{word}':")
    for similar_word, score in model.most_similar(word, topn=5):
        print(f"{similar_word} ({score:.4f})")
```

```
Top 5 words similar to 'king':
kings (0.7138)
queen (0.6511)
monarch (0.6413)
crown_prince (0.6204)
prince (0.6160)

Top 5 words similar to 'university':
universities (0.7004)
faculty (0.6781)
university (0.6758)
undergraduate (0.6587)
univeristy (0.6585)

Top 5 words similar to 'money':
monies (0.7165)
funds (0.7055)
moneys (0.6289)
dollars (0.6289)
cash (0.6151)

Top 5 words similar to 'doctor':
physician (0.7806)
doctors (0.7477)
gynecologist (0.6948)
surgeon (0.6793)
dentist (0.6785)

Top 5 words similar to 'india':
indian (0.6967)
usa (0.6836)
pakistan (0.6815)
chennai (0.6676)
america (0.6589)

print("king - man + woman =")
print(model.most_similar(positive=["king", "woman"], negative=["man"], topn=1))

print("\nparis - france + india =")
print(model.most_similar(positive=["paris", "india"], negative=["france"], topn=1))

print("\nteacher - school + hospital =")
print(model.most_similar(positive=["teacher", "hospital"], negative=["school"], topn=1))

king - man + woman =
[('queen', 0.7118193507194519)]
```

```

paris - france + india =
[('chennai', 0.5442505478858948)]

teacher - school + hospital =
[('Hospital', 0.6331106424331665)]

words = [
    "king", "queen", "man", "woman",
    "doctor", "nurse",
    "paris", "france", "india", "delhi",
    "school", "college", "university",
    "apple", "banana", "fruit",
    "car", "bus", "vehicle"
]

word_vectors = np.array([model[word] for word in words])

pca = PCA(n_components=2)
reduced = pca.fit_transform(word_vectors)

plt.figure(figsize=(10,8))

for i, word in enumerate(words):
    x, y = reduced[i]
    plt.scatter(x, y)
    plt.text(x+0.01, y+0.01, word)

plt.title("Word Embeddings Visualization (GoogleNews Word2Vec)")
plt.show()

```

Word Embeddings Visualization (GoogleNews Word2Vec)

