Natural Language Processing

Gabe Baksa and Brendan Granzo

ChatGPT was used as a reference and to help format this report

Part 1: Word2Vec on Custom Corpus

Step 1: Define Initial Corpus

The initial corpus consisted of a small set of sentences for training:

"This is a sample sentence for Word2Vec."

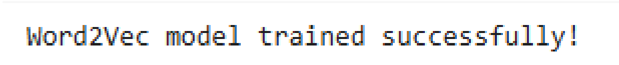
"Word2Vec generates word embeddings."

"This is the final sample sentence for testing."

Step 2: Train the Word2Vec Model

The Word2Vec model was trained using the above corpus with the following parameters:

- **Vector Size**: 50
- **Window Size**: 3
- **Skip-Gram**: Disabled (used CBOW instead).
- **Epochs**: 10

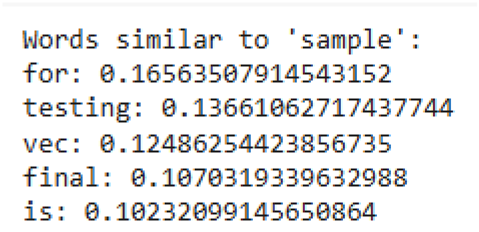


```
Word2Vec model trained successfully!
```

The training completed successfully, and the model was ready for testing.

Step 3: Top 5 Results for a Word

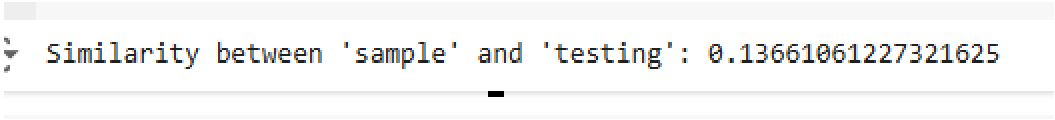
The word **"sample"** was selected, and its top 5 most similar words were:



```
Words similar to 'sample':
for: 0.16563507914543152
testing: 0.13661062717437744
vec: 0.12486254423856735
final: 0.1070319339632988
is: 0.10232099145650864
```

Step 4: Similarity Between Two Words

The similarity between the words **"sample"** and **"testing"** was calculated:

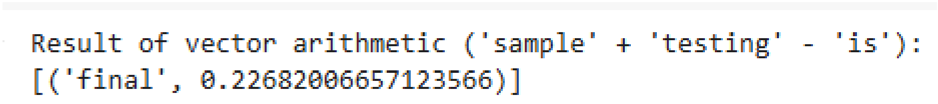


```
Similarity between 'sample' and 'testing': 0.13661061227321625
```

Step 5: Vector Arithmetic

A vector arithmetic operation was performed:

sample + testing - is = final



```
Result of vector arithmetic ('sample' + 'testing' - 'is'):
[('final', 0.22682006657123566)]
```

- **Result**: **final** with a similarity score of `0.22`.

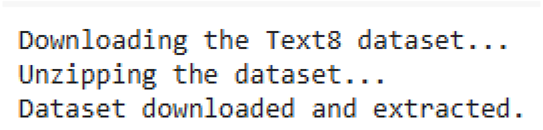
Observations

- The similarity scores reflect relationships within the small corpus but are relatively low due to limited training data.
- Vector arithmetic correctly identified **"final"**, which is contextually relevant to the operation.

Part 2: Text8 Dataset and Word2Vec

Step 1: Download the Text8 Dataset

The **Text8 dataset** was downloaded and extracted successfully. This dataset contains a large corpus of English words for training models.

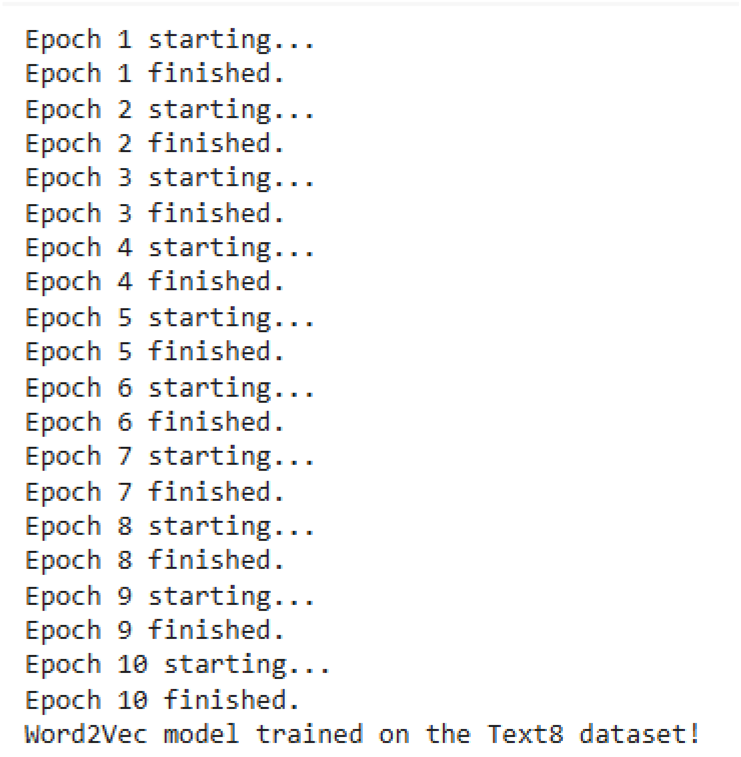


```
Downloading the Text8 dataset...
Unzipping the dataset...
Dataset downloaded and extracted.
```

Step 2: Train the Word2Vec Model

The Word2Vec model was trained on the Text8 dataset using the following parameters:

- **Vector Size**: `100`
- **Window Size**: `5`
- **Skip-Gram**: Enabled
- **Epochs**: `10`



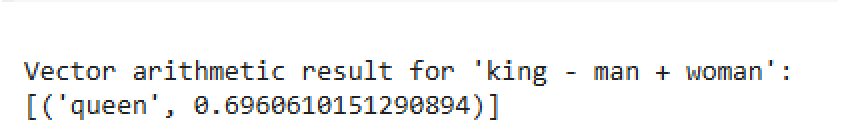
```
Epoch 1 starting...
Epoch 1 finished.
Epoch 2 starting...
Epoch 2 finished.
Epoch 3 starting...
Epoch 3 finished.
Epoch 4 starting...
Epoch 4 finished.
Epoch 5 starting...
Epoch 5 finished.
Epoch 6 starting...
Epoch 6 finished.
Epoch 7 starting...
Epoch 7 finished.
Epoch 8 starting...
Epoch 8 finished.
Epoch 9 starting...
Epoch 9 finished.
Epoch 10 starting...
Epoch 10 finished.
Word2Vec model trained on the Text8 dataset!
```

The training completed successfully, creating a high-quality word embedding model.

Step 3: Vector Arithmetic

The model was tested with a vector arithmetic operation:

king - man + woman = ?

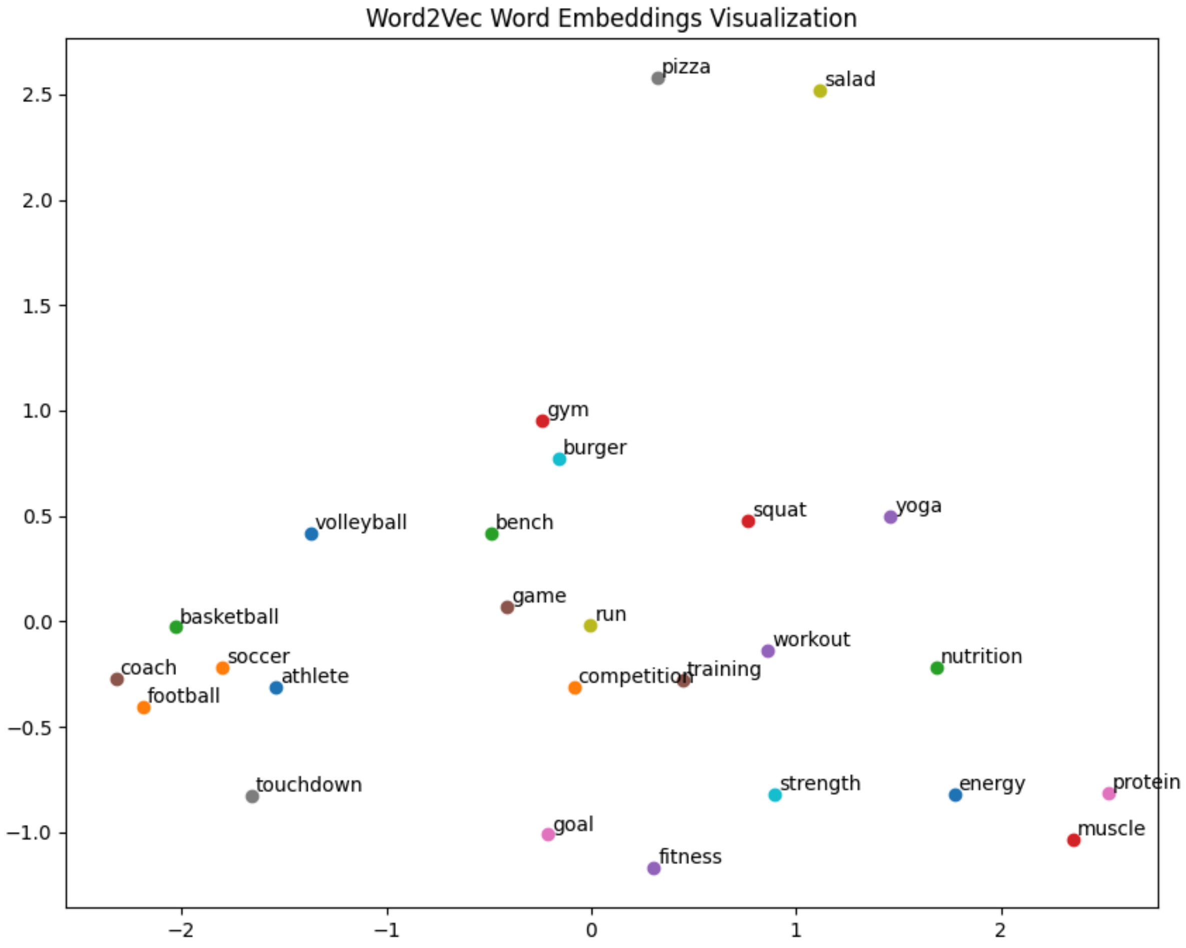


```
Vector arithmetic result for 'king - man + woman':
[('queen', 0.6960610151290894)]
```

- **Result**: **queen** with a similarity score of `0.696`.

This result demonstrates the ability of the model to understand semantic relationships between words.

Step 4: Word Embeddings Visualization



Using PCA, the Word2Vec embeddings were reduced to two dimensions for visualization. The following words were selected for visualization:

[
'athlete', 'soccer', 'basketball', 'gym', 'fitness', 'training',
'protein', 'pizza', 'salad', 'burger', 'volleyball', 'football', 'spikeball',
'bench', 'squat', 'deadlift', 'workout', 'coach', 'goal', 'touchdown',
'run', 'strength', 'energy', 'competition', 'nutrition', 'muscle', 'yoga', 'game'
]

Observations:

1. **Sports Cluster**:
    - Words like `soccer`, `basketball`, `football`, `volleyball`, and `athlete` formed a **tight cluster**, reflecting their shared sports context.

    - Related terms like `goal` and `touchdown` are also positioned nearby.

2. **Fitness Cluster**:
    - Words such as `bench`, `squat`, `deadlift`, `workout`, `training`, `strength`, and `muscle` are grouped together, indicating a strong semantic connection in the context of fitness.
3. **Food Cluster**:
    - Words like `pizza`, `salad`, and `burger` are **outliers**, located far from the sports and fitness clusters, which correctly reflects their unrelated context.
4. **Distinct Terms**:
    - `yoga`, `gym`, and `nutrition` occupy unique positions in the space, suggesting they bridge multiple contexts (e.g., fitness and health).

Observations

- The model effectively captures relationships like **gender analogies** (e.g., "king" to "queen").
- The visualization confirms semantic groupings in the embeddings, reflecting strong training on the Text8 dataset.

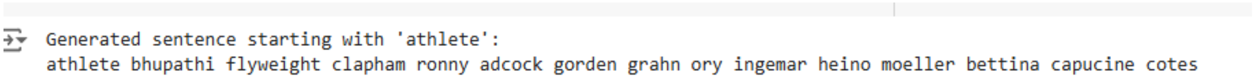
Let me know when you're ready to proceed with Part 3!

Part 3: Random Sentence Generator

Initial Attempts

The goal was to train a random sentence generator using the Word2Vec model to produce meaningful sentences based on seed words.

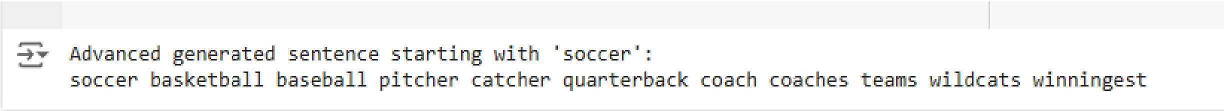
1. **Initial Output**: The first attempts produced **gibberish**, such as:



```
Generated sentence starting with 'athlete':
athlete bhupathi flyweight clapham ronny adcock gorden grahn ory ingemar heino moeller bettina capucine cotes
```

- The output consisted of **random, contextually unrelated words** due to the lack of grammar constraints.

2. **Refinement**: By applying **part-of-speech (POS) filtering** and focusing on nouns, the generator began producing sentences that were structurally coherent but stacked with nouns:

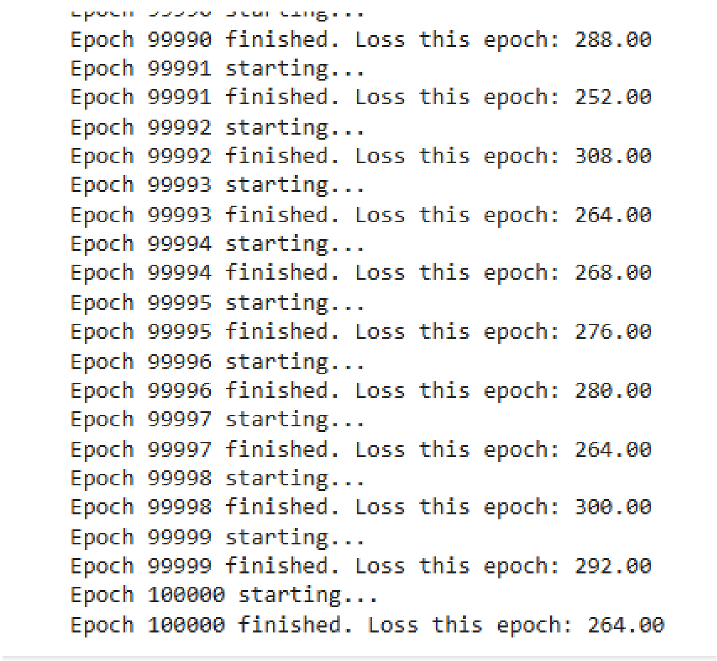


```
Advanced generated sentence starting with 'soccer':
soccer basketball baseball pitcher catcher quarterback coach coaches teams wildcats winningest
```

- While these outputs were semantically related, they lacked verbs and grammar rules for sentence formation.

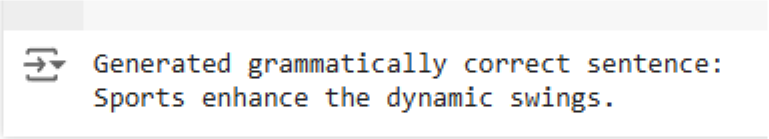
Final Attempt: Improved Grammar

Trained with 100 sentences and 100,000 epochs.

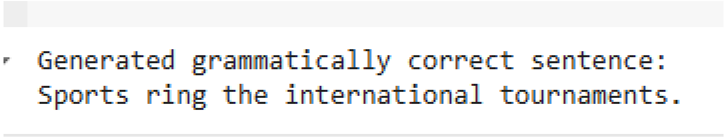


```
Epoch 99990 starting...
Epoch 99990 finished. Loss this epoch: 288.00
Epoch 99991 starting...
Epoch 99991 finished. Loss this epoch: 252.00
Epoch 99992 starting...
Epoch 99992 finished. Loss this epoch: 308.00
Epoch 99993 starting...
Epoch 99993 finished. Loss this epoch: 264.00
Epoch 99994 starting...
Epoch 99994 finished. Loss this epoch: 268.00
Epoch 99995 starting...
Epoch 99995 finished. Loss this epoch: 276.00
Epoch 99996 starting...
Epoch 99996 finished. Loss this epoch: 280.00
Epoch 99997 starting...
Epoch 99997 finished. Loss this epoch: 264.00
Epoch 99998 starting...
Epoch 99998 finished. Loss this epoch: 300.00
Epoch 99999 starting...
Epoch 99999 finished. Loss this epoch: 292.00
Epoch 100000 starting...
Epoch 100000 finished. Loss this epoch: 264.00
```

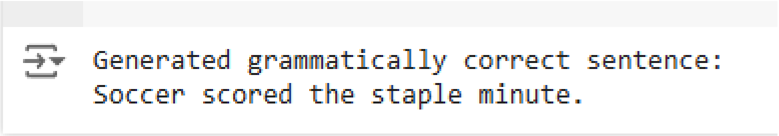
The sentence generator was refined further by enforcing **basic grammar rules** (e.g., Subject → Verb → Object structure). This produced sentences like:



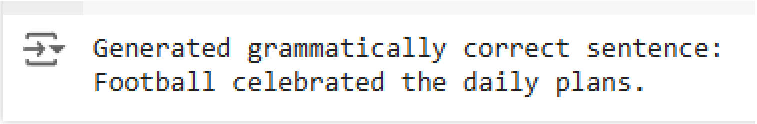
```
Generated grammatically correct sentence:
Sports enhance the dynamic swings.
```



```
Generated grammatically correct sentence:
Sports ring the international tournaments.
```



```
Generated grammatically correct sentence:
Soccer scored the staple minute.
```



```
Generated grammatically correct sentence:
Football celebrated the daily plans.
```

- **Successes**:
    - Some sentences followed logical structures and were grammatically correct.
    - The model demonstrated an improved ability to use diverse parts of speech (nouns, verbs, adjectives).
- **Limitations**:
    - The generator didn't work with every seed word; some words produced repetitive or nonsensical sentences.
    - Words with fewer relationships in the corpus struggled to integrate into meaningful sentences.

Observations

1. **Improvements**: Incorporating **POS tagging** and **grammar constraints** significantly improved the output quality.
2. **Challenges**: The generator still struggled with:
    - Seed words that lacked sufficient context in the training data.
    - Rare words or those outside the primary corpus.