Firefly Algorithm for Hyperparameter Tuning of CBOW Word2Vec Embeddings Models

Defended by:
Moussa JAMOR

Defended on May 22, 2024, before the jury:
Prof. Yasser EL MADANI EL ALAMI

ENSIAS - Filière: Ingénierie en Intelligence Artificielle

Swarm Intelligence

Swarm intelligence algorithms are inspired by the collective behavior of social creatures.

Figure 1: Swarm intelligence: Birds

Firefly Algorithm

The **Firefly Algorithm (FA)**, introduced by Xin-She Yang, is inspired by the flashing behavior of fireflies.

Figure 2: Fireflies flashing at night

Firefly Algorithm

The Firefly Algorithm is based on three key hypotheses:

- **All-Attractiveness**: All fireflies are attracted to each other regardless of their gender.
- **Brightness-Based Attraction**: Attractiveness is proportional to brightness, meaning less bright fireflies move towards brighter ones.
- **Random Movement**: If no brighter fireflies are nearby, a firefly moves randomly.

Firefly Algorithm

The main equation for the Firefly Algorithm:

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2} \left(x_j^t - x_i^t\right) + \alpha \epsilon_i^t \tag{1}$$

where r_{ij} denotes the Cartesian distance between firefly i and firefly j, given by

$$r_{ij} = \|x_i - x_j\| = \sqrt{\sum_{k=1}^{n} \left(x_{i,k} - x_{j,k}\right)^2}$$

Firefly Algorithm

The attractiveness β is inversely proportional to the distance, which is given by the formula:

$$\beta = \beta_0 e^{-\gamma r^2}$$

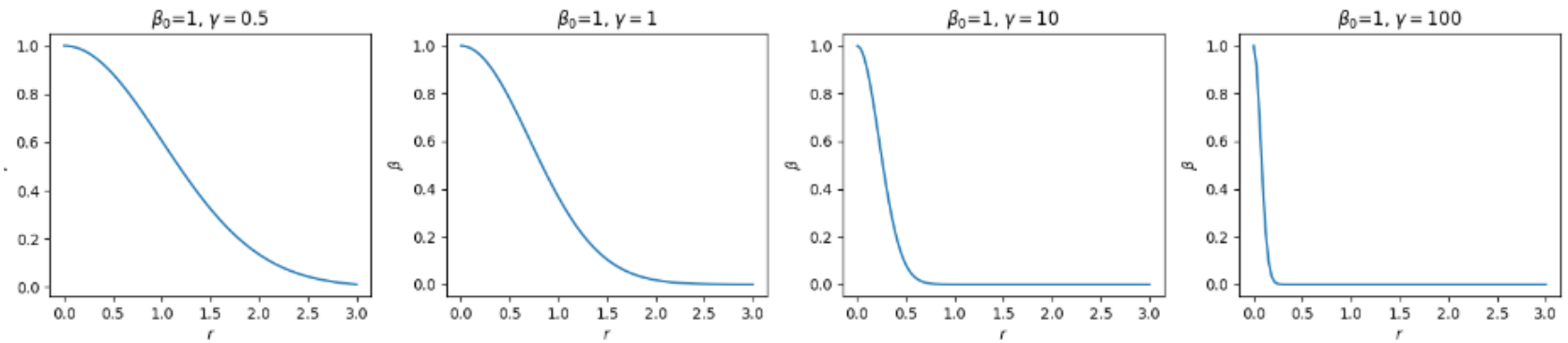


Figure 3: Effect of Gamma parameter

Note: As $r \to +\infty$, $\beta \to 0$.

Firefly Algorithm

Algorithm 1 FireFly Algorithm

Require: Light absorption coefficient γ, maximum generation $MaxGeneration$
Require: Initial population of fireflies x_i $(i = 1, 2, ..., n)$
Generate initial population of fireflies x_i $(i = 1, 2, ..., n)$
Evaluate light intensity I_i at x_i using objective function $f(x)$
Initialize iteration counter $t = 0$
while $t < MaxGeneration$ **do**
  for $i = 1$ **to** n **do**
    for $j = 1$ **to** n **do**
      if $I_j > I_i$ **then**
        Move firefly i towards j in d-dimension
      end if
      Attractiveness varies with distance r via $\exp[-\gamma r^2]$
      Evaluate new solutions and update light intensity
    end for
  end for
  Rank the fireflies and find the current best
  $t = t + 1$
end while

Overview of the CBoW Model

CBoW stands for **Continuous Bag of Words**. It is a **Word2Vec** model based on a neural network architecture, which learns to represent **words** with continuous vectors.

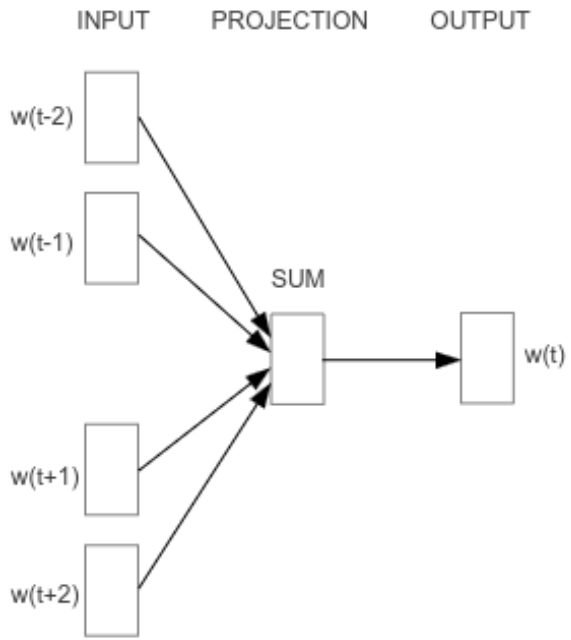


Figure 4: CBoW Model Architecture

CBoW Model Architecture

The following figure represents the model's architecture used in the experiment results:

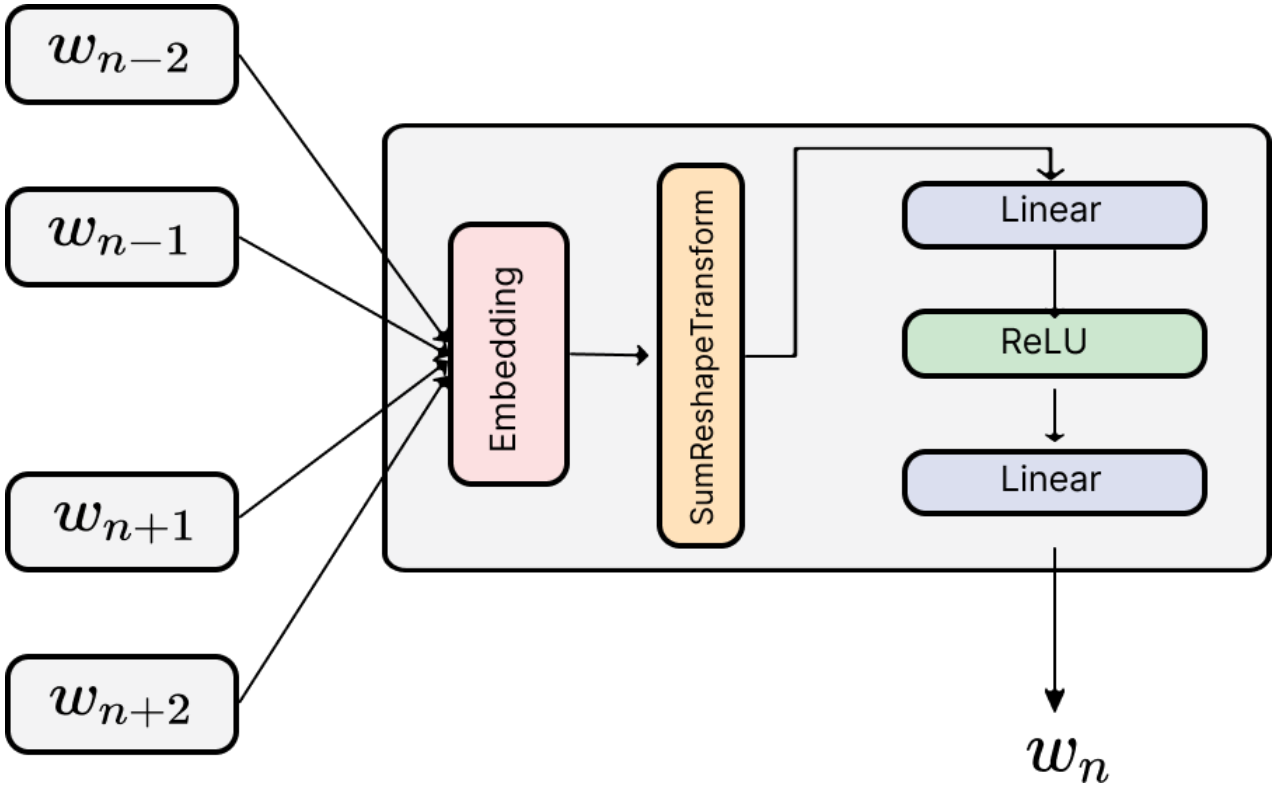


Figure 5: CBoW Model Architecture

How It Works

The CBoW model has a basic idea: given a larger corpus of text and a window of words w, it tries to predict the word in the middle.

Firefly Algorithm for Hyperparameter Tuning of CBOW Word2Vec Embeddings Models

Moussa JAMOR
National school For Computer Science Artificial intelligence major
ENSIAS Rabat, Morocco
moussa_jamor@um5.ac.ma

Hyperparameter Tuning Workflow

The following figure represents the workflow of hyperparameters tuning using the Firefly Algorithm.

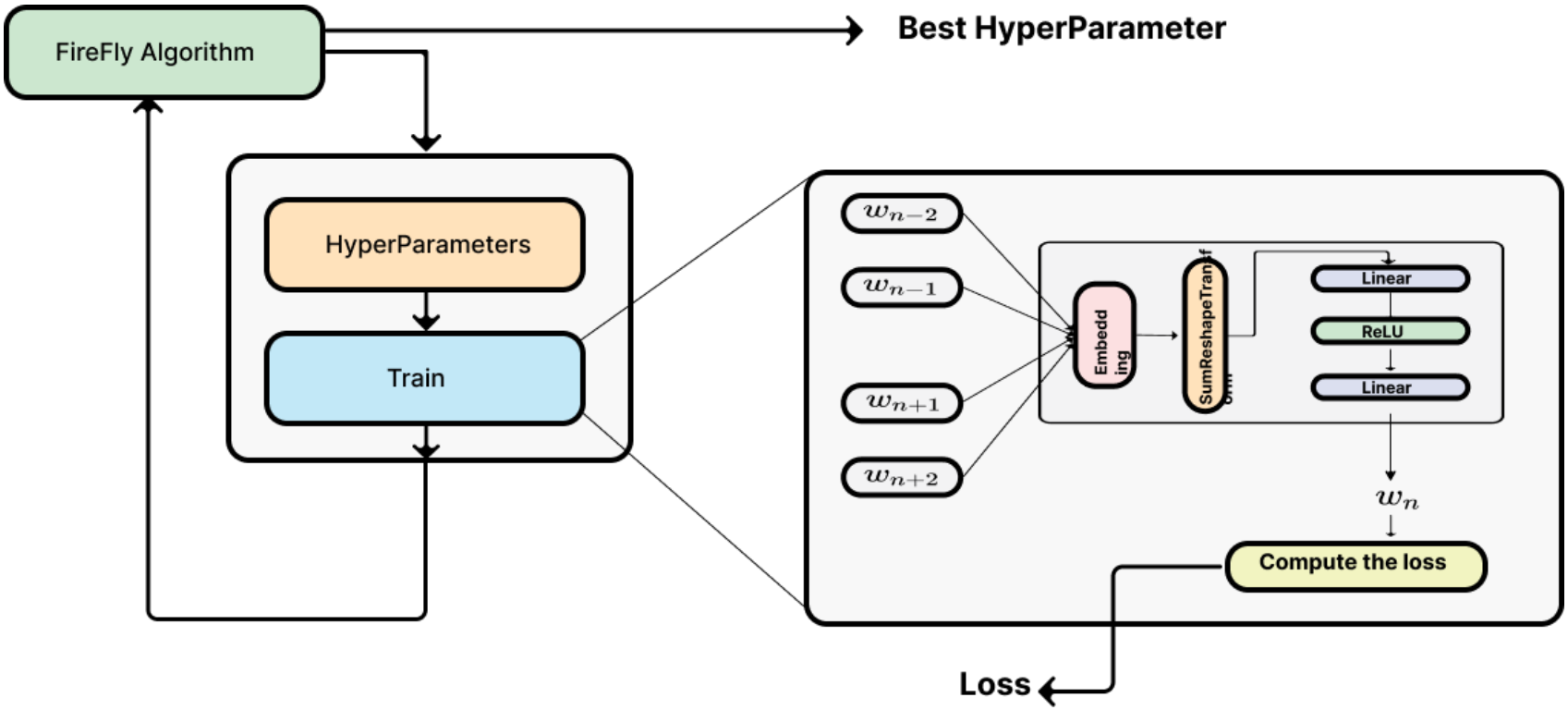


Figure 6: Workflow of Hyperparameters Tuning with the Firefly Algorithm

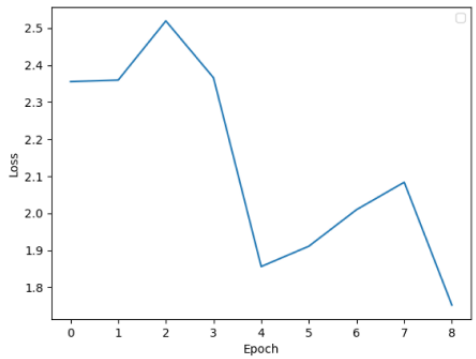
Hyperparameter Tuning Experiment

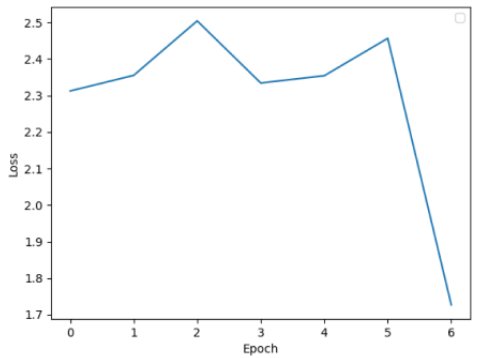
The figure below shows the script execution process for hyperparameter tuning using the Firefly Algorithm.

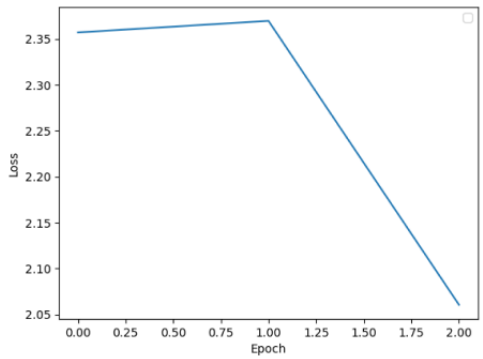
```
⊗ → FireFly-Optimizer-Deep-Learning git:(main) ✗ python3 hyperparam.py --numtest=1 --popsize=5 --maxiters=3 --gamma=0
.1 --alpha=2
/home/moussa/.local/lib/python3.11/site-packages/matplotlib/projections/__init__.py:63: UserWarning: Unable to impor
t Axes3D. This may be due to multiple versions of Matplotlib being installed (e.g. as a system package and as a pip
package). As a result, the 3D projection is not available.
  warnings.warn("Unable to import Axes3D. This may be due to multiple versions of "
/home/moussa/.local/lib/python3.11/site-packages/torch/nn/init.py:412: UserWarning: Initializing zero-element tensor
s is a no-op
  warnings.warn("Initializing zero-element tensors is a no-op")
Loss: 3.63418: 100%|| 10/10 [00:01<00:00,  8.48it/s]
/home/moussa/.local/lib/python3.11/site-packages/torch/nn/init.py:412: UserWarning: Initializing zero-element tensor
s is a no-op
  warnings.warn("Initializing zero-element tensors is a no-op")
Loss: 5.79429: 100%|| 10/10 [00:01<00:00,  8.65it/s]
Loss: 4.82992: 100%|| 10/10 [00:01<00:00,  8.34it/s]
Loss: 6.45045: 100%|| 10/10 [00:01<00:00,  8.69it/s]
Loss: 4.12953: 100%|| 10/10 [00:01<00:00,  8.44it/s]
Loss: 2.18995:  60%|| 6/10 [00:00<00:00,  7.39it/s]
Loss: 2.18995:  60%|| 6/10 [00:01<00:00,  5.75it/s]
```

Figure 7: Execution of the Hyperparameter Tuning Script

Experiment HyperParamerts tuning







Parameter	Figure 1	Figure 2	Figure 3
Learning Rate (lr)	0.0093	0.0100	0.0100
β_1	0.899	0.9003	0.901
β_2	0.9981	0.9989	0.9991
Embedding Dimension	2	4	5
Window Size (w)	3	3	2
Best Intensity	1.83	**1.72**	2.06

Table 1: Hyperparameters and Best Intensities for Different Figures

Training with Optimized Hyperparameters

The figure below illustrates the process of running training scripts using the hyperparameters found through the tuning process.

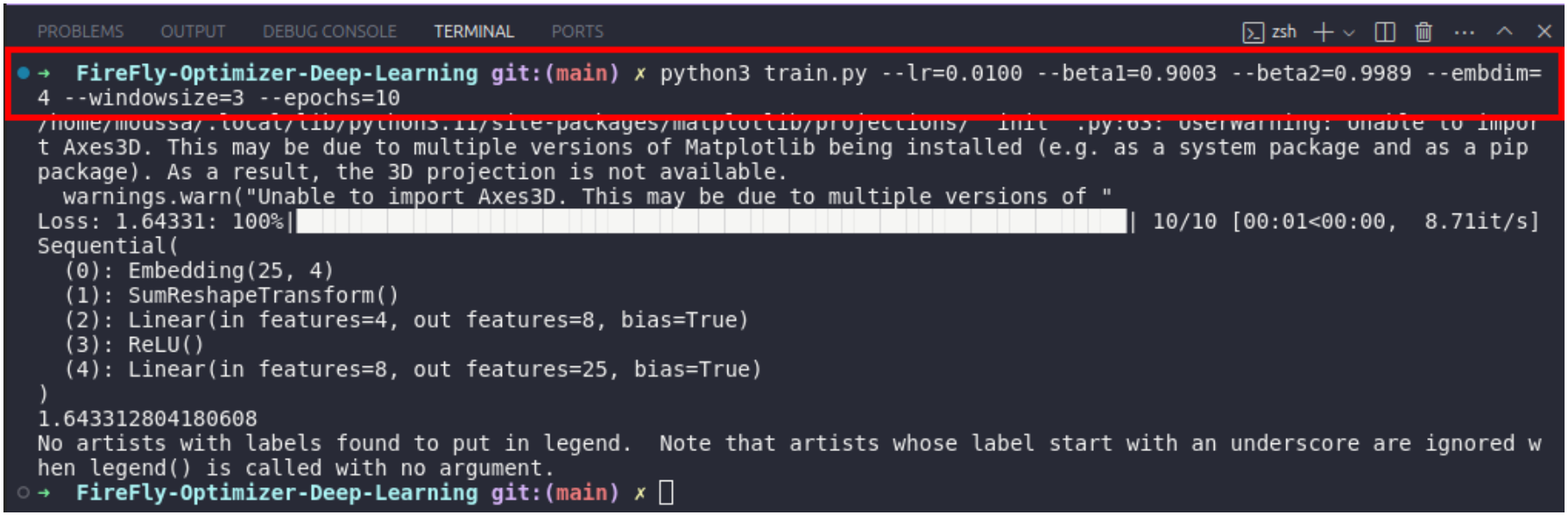


Figure 8: Execution of Training Scripts with Optimized Hyperparameters

Training with Optimized Hyperparameters

The figure below shows the training loss and the plot of words in 2D space. **PCA** is used for dimensionality reduction.

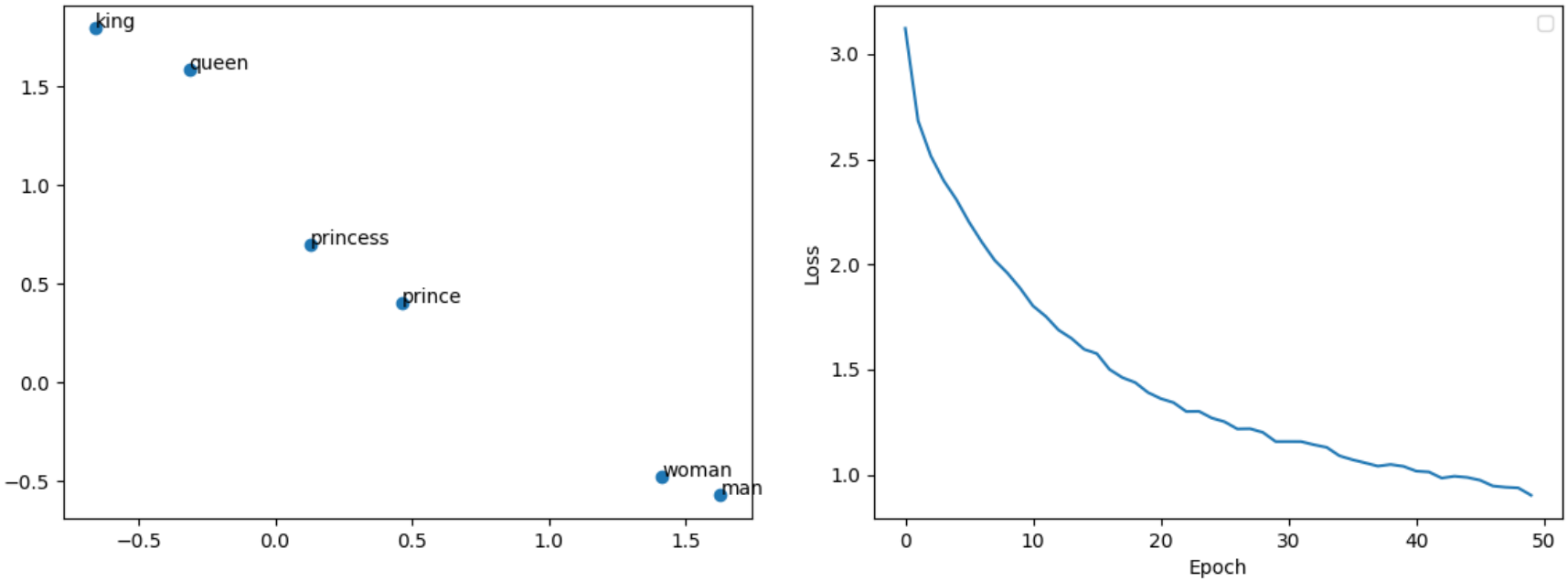


Figure 9: Training Loss and Words Plot in 2D Space