```
'''
Implement the Continuous Bag of Words (CBOW) Model. Stages can be:
a. Data preparation
b. Generate training data
c. Train model
d. Output
'''


import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,\
      Embedding, Lambda
from tensorflow.keras.preprocessing.text import Tokenizer
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
import re

#a. Data preparation
data = """We are about to study the idea of a computational process.
Computational processes are abstract beings that inhabit computers.
As they evolve, processes manipulate other abstract things called
data.
The evolution of a process is directed by a pattern of rules
called a program. People create programs to direct processes. In
effect,
we conjure the spirits of the computer with our spells."""


sentences = data.split(".")


sentences

['We are about to study the idea of a computational process',
 '\nComputational processes are abstract beings that inhabit
computers',
 '\nAs they evolve, processes manipulate other abstract things called
data',
 '\nThe evolution of a process is directed by a pattern of rules\
ncalled a program',
 ' People create programs to direct processes',
 ' In effect,\nwe conjure the spirits of the computer with our
spells',
 '']


#Clean Data
clean_sentences = []
for sentence in sentences:
    # skip empty string
    if sentence == "":
        continue;
    # remove special characters
    sentence = re.sub('[^A-Za-z0-9]+', ' ', sentence)
    # remove 1 letter words
    sentence = re.sub(r'(?:^| )\w(?:$| )', ' ', sentence).strip()
    # lower all characters
    sentence = sentence.lower()
    clean_sentences.append(sentence)
```

```
clean_sentences
```

```
['we are about to study the idea of computational process',
 'computational processes are abstract beings that inhabit computers',
 'as they evolve processes manipulate other abstract things called
data',
 'the evolution of process is directed by pattern of rules called
program',
 'people create programs to direct processes',
 'in effect we conjure the spirits of the computer with our spells']
```

```
# Define the corpus
corpus = clean_sentences
```

```
# Convert the corpus to a sequence of integers
tokenizer = Tokenizer()
tokenizer.fit_on_texts(corpus)
sequences = tokenizer.texts_to_sequences(corpus)
print("After converting our words in the corpus \
into vector of integers:")
print(sequences)
```

```
After converting our words in the corpus into vector of integers:
[[4, 5, 11, 6, 12, 1, 13, 2, 7, 8], [7, 3, 5, 9, 14, 15, 16, 17], [18,
19, 20, 3, 21, 22, 9, 23, 10, 24], [1, 25, 2, 8, 26, 27, 28, 29, 2,
30, 10, 31], [32, 33, 34, 6, 35, 3], [36, 37, 4, 38, 1, 39, 2, 1, 40,
41, 42, 43]]
```

```
# creating dictionary for word to index and index to word
index_to_word_map = {}
word_to_index_map = {}
for index_1, sequence in enumerate(sequences):
    print(sequence)
    words_in_sentence = clean_sentences[index_1].split()
    print(words_in_sentence)
    for index_2, value in enumerate(sequence):
        index_to_word_map[value] = words_in_sentence[index_2]
        word_to_index_map[words_in_sentence[index_2]] = value
```

```
[4, 5, 11, 6, 12, 1, 13, 2, 7, 8]
['we', 'are', 'about', 'to', 'study', 'the', 'idea', 'of',
'computational', 'process']
[7, 3, 5, 9, 14, 15, 16, 17]
['computational', 'processes', 'are', 'abstract', 'beings', 'that',
'inhabit', 'computers']
[18, 19, 20, 3, 21, 22, 9, 23, 10, 24]
['as', 'they', 'evolve', 'processes', 'manipulate', 'other',
'abstract', 'things', 'called', 'data']
[1, 25, 2, 8, 26, 27, 28, 29, 2, 30, 10, 31]
['the', 'evolution', 'of', 'process', 'is', 'directed', 'by',
'pattern', 'of', 'rules', 'called', 'program']
[32, 33, 34, 6, 35, 3]
['people', 'create', 'programs', 'to', 'direct', 'processes']
[36, 37, 4, 38, 1, 39, 2, 1, 40, 41, 42, 43]
['in', 'effect', 'we', 'conjure', 'the', 'spirits', 'of', 'the',
'computer', 'with', 'our', 'spells']
```

```
print(index_to_word_map)
print("\n")
print(word_to_index_map)
```

```
{4: 'we', 5: 'are', 11: 'about', 6: 'to', 12: 'study', 1: 'the', 13:
'idea', 2: 'of', 7: 'computational', 8: 'process', 3: 'processes', 9:
'abstract', 14: 'beings', 15: 'that', 16: 'inhabit', 17: 'computers',
18: 'as', 19: 'they', 20: 'evolve', 21: 'manipulate', 22: 'other', 23:
'things', 10: 'called', 24: 'data', 25: 'evolution', 26: 'is', 27:
'directed', 28: 'by', 29: 'pattern', 30: 'rules', 31: 'program', 32:
'people', 33: 'create', 34: 'programs', 35: 'direct', 36: 'in', 37:
'effect', 38: 'conjure', 39: 'spirits', 40: 'computer', 41: 'with',
42: 'our', 43: 'spells'}


{'we': 4, 'are': 5, 'about': 11, 'to': 6, 'study': 12, 'the': 1,
'idea': 13, 'of': 2, 'computational': 7, 'process': 8, 'processes': 3,
'abstract': 9, 'beings': 14, 'that': 15, 'inhabit': 16, 'computers':
17, 'as': 18, 'they': 19, 'evolve': 20, 'manipulate': 21, 'other': 22,
'things': 23, 'called': 10, 'data': 24, 'evolution': 25, 'is': 26,
'directed': 27, 'by': 28, 'pattern': 29, 'rules': 30, 'program': 31,
'people': 32, 'create': 33, 'programs': 34, 'direct': 35, 'in': 36,
'effect': 37, 'conjure': 38, 'spirits': 39, 'computer': 40, 'with':
41, 'our': 42, 'spells': 43}
```

```
#gen training data

# Define the parameters
vocab_size = len(tokenizer.word_index) + 1
```

```
embedding_size = 10
window_size = 2

# Generate the context-target pairs
contexts = []
targets = []
for sequence in sequences:
     for i in range(window_size, len(sequence) - window_size):
          context = sequence[i - window_size:i] + sequence[i + 1:i +
window_size + 1]
          target = sequence[i]
          contexts.append(context)
          targets.append(target)


# sample of training data
for i in range(5):
    words = []
    target = index_to_word_map.get(targets[i])
    for j in contexts[i]:
        words.append(index_to_word_map.get(j))
    print(words, "=>", target)
```

```
['we', 'are', 'to', 'study'] => about
['are', 'about', 'study', 'the'] => to
['about', 'to', 'the', 'idea'] => study
['to', 'study', 'idea', 'of'] => the
['study', 'the', 'of', 'computational'] => idea
```

```
# Convert the contexts and targets to numpy arrays
X = np.array(contexts)
Y = np.array(targets)

#train  model

# Define the CBOW model
model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=embedding_size,
input_length=2 * window_size))
model.add(Lambda(lambda x: tf.reduce_mean(x, axis=1)))
model.add(Dense(256, activation='relu'))
model.add(Dense(512, activation='relu'))
model.add(Dense(units=vocab_size, activation='softmax'))
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python310\lib\site-
packages\keras\src\layers\core\embedding.py:97: UserWarning: Argument
`input_length` is deprecated. Just remove it.
  warnings.warn(
```