Assignment No.5

Title: Implement the Continuous Bag of Words (CBOW) Model

Aim: Implement the Continuous Bag of Words (CBOW) Model. Stages can be:

a. Data preparation

b. Generate training data

c. Train model

d. Output

Theory：

1) What is NLP ?
2) What is Word embedding related to NLP ?
3) Explain Word2Vec techniques.
4) Enlist applications of Word embedding in NLP.
5) Explain CBOW architecture.
6) What will be input to CBOW model and Output to CBW model.
7) What is Tokenizer .
8) Explain window size parameter in detail for CBOW model.
9) Explain Embedding and Lmbda layer from keras
10) What is yield()

Steps/ Algorithm

1. Dataset link and libraries :

Create any English 5 to 10 sententece paragraph

as inputImport following data from keras :

keras.models import Sequential

keras.layers import Dense, Embedding, Lambda

keras.utils import np_utils

keras.preprocessing import sequence

keras.preprocessing.text import Tokenizer

Import Gensim for NLP operations : requirements :

Gensim runs on Linux, Windows and Mac OS X, and should run on any other platform that supports Python 3.6+ and NumPy. Gensim depends on the following software: Python, testedwith versions 3.6, 3.7 and 3.8. NumPy for number crunching.

a) Import following libraries gemsim and numpy set i.e. text file created . It should bepreprocessed.

b) Tokenize the every word from the paragraph . You can call in built tokenizer present inGensim

c) Fit the data to tokenizer

d) Find total no of words and total no of sentences.

e) Generate the pairs of Context words and target words :

```
e.g. cbow_model(data, window_size,
   total_vocab):total_length =
   window_size*2
   for text in data:
      text_len =
      len(text)
      for idx, word in
         enumerate(text):
         context_word = []
         target  = []
         begin = idx -
   window_size end = idx +
   window_size + 1
context_word.append([text[i] for i in range(begin, end) if 0 <= i < text_len and i
  != idx])
         target.append(word)
         contextual = sequence.pad_sequences(context_word, total_length=total_length)
         final_target = np_utils.to_categorical(target, total_vocab)
```

yield(contextual, final_target)

f) Create Neural Network model with following parameters . Model type : sequential

Layers : Dense , Lambda ,
embedding. Compile Options :
(loss='categorical_crossentropy',
optimizer='adam')

g) Create

vector file of

some word

for

testinge.g.:di

mensions=1

00

vect_file = open('/content/gdrive/My

Drive/vectors.txt' ,'w')

vect_file.write('{}

{}\n'.format(total_vocab,dimensions)

h) Assign weights to your trained model
e.g. weights = model.get_weights()[0]

for text, i in vectorize.word_index.items():

final_vec = '

'.join(map(str,

list(weights[i, :])))

vect_file.write('{}

{}\n'.format(text,

final_vec)

Close()

i) Use the vectors created in Gemsim :

e.g. cbow_output =
gensim.models.KeyedVectors.load_word2vec_forma
t('/content/gdrive/MyDrive/vectors.txt',

binary=False)

j) choose the

word to get similar

type ofwords:

cbow_output.most_similar

(positive=['Yourword'])

Conclusion: In this experiment, we saw what a CBOW model is and how it works. We also implemented the model on a custom dataset and got good output. We learnt what word embeddings are and how CBOW is useful. These can be used for text recognition, speech to text conversion etc.