Assignment 5

ASSIGNMENT 5:-

```
import numpy as np
import keras.backend as K
from keras.models import Sequential
from keras.layers import Dense,Embedding,Lambda
from keras.utils import np_utils
from keras.preprocessing import sequence
from keras.preprocessing.text import Tokenizer
import gensim

data = open("/content/corona.txt","r")
covid_data= [text for text in data if text.count("")>=2]
vectorize=Tokenizer()
vectorize.fit_on_texts(covid_data)
covid_data=vectorize.texts_to_sequences(covid_data)
total_vocab=sum(len(s) for s in covid_data)
word_count=len(vectorize.word_index)+1
window_size=2


def cbow_model(data,windows_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=to
tal_length)
      final_target=np_utils.to_categorical(target, total_vocab)
      yield(contextual, final_target)


model=Sequential()
model.add(Embedding(input_dim=total_vocab,output_dim=100,input_length=w
indow_size*2))
model.add(Lambda(lambda x:K.mean(x,axis=1), output_shape=(100,)))
model.add(Dense(total_vocab, activation="softmax"))
model.compile(loss="categorical_crossentropy", optimizer="adam")
for i in range(10):
  cost=0
  for x, y in cbow_model(data,window_size, total_vocab):
    cost+=model.train_on_batch(contextual, final_target)
```

```
  print(i, cost)

dimensions = 100
vect_file=open("/content/drive/MyDrive/vector.txt", "w")
vect_file.write('{} {}\n'.format(total_vocab, dimensions))

weight=model.get_weights()[0]
for text, i in vectorize.word_index.items():
  final_vec="".join(map(str, list(weight[i,:])))
  vect_file.write('{}{}\n'.format(text, final_vec))
vect_file.close()

cbow_output=gensim.models.KeyedVectors.load_word2vec_format("/content/d
rive/MyDrive/vector.txt", binary=False)
cbow_output.most_similar(positive=["virus"])
```

OUTPUT:-

0 0 1 0 2 0 3 0 4 0 5 0 6 0 7 0 8 0 9 0

8