Demo 1: Word Embeddings Using Keras Sequential API

Sara El-Metwally

Demo Objective:

Word embedding utilizes the idea of mapping the text words that have the same meaning into similar representation. Each word will be represented as a vector of real numbers. Keras is the most famous python library used for machine learning and in this demo we will used it to implement a simple classification task based on word embedding to read the mood of your written sentences (i.e. sentiment analysis: classify the written sentences into positive (1) or negative (0).

Tools:

Assume that we are currently using Anaconda platform.

- **Keras:** enable you to implement the **word embeddings** using its **sequential API**
- **Installation:** open your terminal and write: `conda install -c anaconda keras`

Corpus (i.e. Data):

- The data set will be simple reviews about books. The following sentences will create our corpus:

```
This book is amazing
I really like it
It is exceptionally good
It is a brilliant book
long boring book
It is really bad
I did not like it
waste of time
do not read it
I really heat it
```

Guiding Steps:

- Corpus pre-processing:
  - Define a **corpus** vector initialized with the above book review sentences.
  - Define a **sentiment** vector that has a position to every sentence in the corpus and has a value of 1 for positive review and 0 for a negative one.
  - Extract all words (i.e. tokens) from the corpus (Hint: use **word_tokenize** from **nltk.tokenize**).
  - Determine unique tokens in the corpus (Hint: use a **set** function)
  - Convert the sentences in the corpus to numbers in order to start processing them with Keras embedding layer (Hint: use **one_hot** function from **keras.preprocessing.text).**
  - Keras embedding layer assumes all sentences have equal length, try to fix this by making all sentences have the same length (Hind: compute the longest sentence and append zeros to the others accordingly).

Exercise 1: (12 points)

[1]. How many words in the defined corpus?
[2]. How many unique words in the corpus?
[3]. What is one-hot encoding for the second sentence in the corpus?
[4]. The word book was mentioned in the first, fourth and fifth sentences, which numeric representation is given for it? What did you notice?
[5]. What is the longest sentence in corpus and how many tokens on it?
[6]. How many added zeros to the fourth and seventh sentences in the corpus and why?

- Train a simple classification model (Hint: try to use the following code as a seed to complete your task) :
  - The embedding layer has three parameters: the number of unique words in the corpus (note: add extra 5 as a buffer), the number of dimensions used for each word **(i.e. 20)**, and the length of the input sequence.
  - Create a **sequential** model with the first layer is the above created embedding layer.
  - The embedding layer is **flattened** to connected directly with a densely (i.e. output) layer.
  - In the output layer, we will use a **sigmoid** activation function.
  - The model will be **compiled** and the model summary will be printed.
  - The model will be trained using a **fit** function with 100 epochs.

```
from keras.models import --------------
from keras.layers import --------------
from keras.layers import --------------
from keras.layers.embeddings import -----------

model = ------------()
model.add(Embedding(----,-----,input_length=-------))
model.add(Flatten())
model.add(Dense(1,activation='---------------'))
model.compile(optimizer='adam',loss='binary crossentropy',metrics=['acc'])
print(model.summary())
model.fit(---------, ----------,----------, verbose=1)
```

Exercise 2: (4 points)

[1]. How many parameters used in the embedding layer and why?
[2]. How many parameters used in the output layer and why?

- Test :
  - Because this is a toy example to explain the basic idea of word embedding using keras sequential model, we will test the model using the same corpus data (i.e. the train and test data should be different in the real applications).

Exercise 3: (4 points)

[1]. What is the expected accuracy in the case if you are using the same data set for testing and training?
[2]. What is the expected loss in the case if you are using the same data set for testing and training?

Appendix:

Questions	Answers
E1, Q1	40
E1, Q2	24
E1, Q3	[25, 19, 15, 13]
E1, Q4	19 All words ' **book** 'have the same numeric representation
E1,Q5	It is a brilliant book, 5 words
E1, Q6	Nothing because they are the longest ones.
E2, Q1	580 24+5 no of unique words each has 20 dimensions
E2, Q2	101 The output from the embedding layer will be a sentence that has 5 words and each word is represented by a 20 dimensional vector.
E3, Q1	100.00
E3, Q4	0.00316

Lab Notebook:

https://github.com/SaraEl-Metwally/TOT-ITI-EPITA/blob/main/Word_Embedding_Keras_Lib_Simple.ipynb