README

April 12, 2024

This is the guidence on how to load the vectorized data.

0.0.1 Load the data

[1]:
```python
import pickle
import numpy as np

def load_data(file_name):
    with open(f'{file_name}.pkl', 'rb') as f:
        loaded_data = pickle.load(f)
    return loaded_data
```

[2]:
```python
german_sen_emb = load_data("german_sen_emb")
socc_sen_emb = load_data("socc_sen_emb")
german_word_emb = load_data("german_word_emb")
socc_word_emb = load_data("socc_word_emb")
german_label_expert1 = load_data("german_label_expert1")
german_label_expert2 = load_data("german_label_expert2")
socc_label = load_data("socc_label")
```

0.0.2 Data intro

There are total 7 data files
german represent for the IWG_hatespeech_public dataset (https://github.com/UCSM-DUE/IWG_hatespeech_public)
socc represent for the SFU Opinion and Comments Corpus dataset (https://github.com/sfu-discourse-lab/SOCC)
german_sen_emb and ***socc_sen_emb*** are 2-D numpy.array which are the sentences embedding of the text.
socc_word_emb and ***german_word_emb*** are the word embedding of the text.
german_label_expert1, ***german_label_expert2*** are numpy.array which are the labels annotated by two experts.
socc_label is a numpy.array which is the **toxicity_level** line in **SFU_constructiveness_toxicity_corpus.csv**.

german dataset

[3]:
```python
german_sen_emb.shape #469 is the size of dataset, 768 is the embedding dim of␣
 ↪the sent2vec model
```

```
[3]: (469, 768)
```

```
[4]: len(german_word_emb) #469 is the size of dataset, each of the german_sen_emb␣
     ↪list is a numpy.array, which in shape of (num_words, 300).
```

```
[4]: 469
```

```
[5]: german_word_emb[0].shape # 300 is the dim of the word2vec model, 11 is the␣
     ↪length of the first text (11 word vectors in total)
```

```
[5]: (11, 300)
```

NOTE: EACH ELEMENTS IN THE LIST ARE NOT IN THE SAME SHAPE

```
[6]: german_word_emb[1].shape #different shape with german_word_emb[0].shape
```

```
[6]: (9, 300)
```

```
[7]: german_label_expert1.shape #labels annotated by the first expert
```

```
[7]: (469,)
```

```
[8]: german_label_expert2.shape #labels annotated by the second expert
```

```
[8]: (469,)
```

socc dataset

```
[9]: socc_sen_emb.shape #1043 is the size of dataset, 768 is the embedding dim of␣
     ↪the sent2vec model
```

```
[9]: (1043, 768)
```

```
[10]: len(socc_word_emb) #1043 is the size of dataset, each of the german_sen_emb␣
      ↪list is a numpy.array, which in shape of (num_words, 300).
```

```
[10]: 1043
```

```
[11]: socc_word_emb[0].shape #300 is the dim of the word2vec model, 120 is the length␣
      ↪of the first text (120 word vectors in total)
```

```
[11]: (120, 300)
```

```
[12]: socc_word_emb[1].shape #300 is the dim of the word2vec model, 223 is the length␣
      ↪of the second text (223 word vectors in total)
```

```
[12]: (223, 300)
```

NOTE: socc_word_emb contains np.nan element
This is beacuse there are some samples which all the words of this sample are not included in the word2vec model.

[13]:
```
for i in range(len(socc_word_emb)):
    try:
        if np.isnan(socc_word_emb[i]):
            print(f"The {i} element of socc_word_emb is Nan")
    except:
        pass
```

```
The 123 element of socc_word_emb is Nan
The 364 element of socc_word_emb is Nan
The 422 element of socc_word_emb is Nan
```

[14]:
```
socc_label.shape #labels, the original toxicity_level has 4 levels, I simply␣
 ↪make level 1 as label 0 and levels 2, 3, 4 as label 1
```

[14]:
```
(1043,)
```

[ ]:
```

```