Lab#4, NLP@CGU Spring 2023

This is due on 2023/04/20 16:00, commit to your github as a PDF (lab4.pdf) (File>Print>Save as PDF).

IMPORTANT: After copying this notebook to your Google Drive, please paste a link to it below. To get a publicly-accessible link, hit the *Share* button at the top right, then click "Get shareable link" and copy over the result. If you fail to do this, you will receive no credit for this lab!

LINK: paste your link here

https://colab.research.google.com/drive/1eUImdaTfOVK0krdGgBr8o-X_OXzt9_Sf?usp=sharing

---

Student ID:B0928013

Name:吳佳恩

Word Embeddings for text classification

請訓練一個 kNN或是SVM 分類器來和 Google's Universal Sentence Encoder (a fixed-length 512-dimension embedding) 的分類結果比較

```
!wget -O Dcard.db https://github.com/cjwu/cjwu.github.io/raw/master/courses/nlp2023/lab4-Dcard-Dataset.db
```

```
--2023-04-24 12:31:24--  https://github.com/cjwu/cjwu.github.io/raw/master/courses/nlp2023/lab4-Dcard-Dataset.db
Resolving github.com (github.com)... 140.82.121.3
Connecting to github.com (github.com)|140.82.121.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/cjwu/cjwu.github.io/master/courses/nlp2023/lab4-Dcard-Dataset.db [following]
--2023-04-24 12:31:24--  https://raw.githubusercontent.com/cjwu/cjwu.github.io/master/courses/nlp2023/lab4-Dcard-Dataset.db
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.109.133, 185.199.111.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 151552 (148K) [application/octet-stream]
Saving to: 'Dcard.db'

Dcard.db            100%[===================>] 148.00K  --.-KB/s    in 0.01s

2023-04-24 12:31:24 (13.2 MB/s) - 'Dcard.db' saved [151552/151552]
```

```
import sqlite3
import pandas as pd

conn = sqlite3.connect("Dcard.db")
df = pd.read_sql("SELECT * FROM Posts;", conn)
df
```

	createdAt	title	excerpt	categories	topics	forum_en	forum_zh
0	2022-03-04T07:54:19.886Z	專題需要數據🥺🥺幫填～	希望各位能花個20秒幫我填一下			dressup	穿搭
1	2022-03-	#詢問 找衣服🥺	想找這套衣服🥺，但發現不知道該用什麼關鍵字	詢問	衣服 | 鞋子 | 衣物 |	dressup	穿搭

```
!pip3 install -q tensorflow_text
!pip3 install -q faiss-cpu
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.0/6.0 MB 46.8 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 17.6/17.6 MB 43.1 MB/s eta 0:00:00
```

```
import tensorflow_hub as hub
import numpy as np
import tensorflow_text
import faiss

embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")
```

```
docid = 355
texts • = • "[" • + • df['title'] • + • '] • [' • + • df['topics'] • + • '] • ' • + • df['excerpt'] •
texts[docid]
```

```
'[開了新頻道] [Youtuber | 頻道 | 有趣 | 日常 | 搞笑] 昨天上了第一支影片，之前有發過沒有線條的動畫影片，新的頻道改成有線條的，感覺大家好像比較喜歡這種風格，試試看新的風格，影片內容主要是分享自己遇到的小故事，不知道這樣的頻道大家是否會想要看呢？喜歡的話也'
```

```
embeddings = embed_model(texts)
embed_arrays = np.array(embeddings)
index_arrays = df.index.values
topk = 10
# Step 1: Change data type
embeddings = embed_arrays.astype("float32")

# Step 2: Instantiate the index using a type of distance, which is L2 here
index = faiss.IndexFlatL2(embeddings.shape[1])

# Step 3: Pass the index to IndexIDMap
index = faiss.IndexIDMap(index)

# Step 4: Add vectors and their IDs
index.add_with_ids(embeddings, index_arrays)

D, I = index.search(np.array([embeddings[docid]]), topk)

plabel = df.iloc[docid]['forum_zh']

cols_to_show = ['title', 'excerpt', 'forum_zh']
plist = df.loc[I.flatten(), cols_to_show]

precision = 0
for index, row in plist.iterrows():
    if plabel == row["forum_zh"]:
        precision += 1

print("precision = ", precision/topk)
precision = 0

df.loc[I.flatten(), cols_to_show]
```

```
precision =  0.8
```

	title	excerpt	forum_zh
355	開了新頻道	昨天上了第一支影片，之前有發過沒有線條的動畫影片，新的頻道改成有線條的，感覺大家好像比較喜歡...	YouTuber
359	一個隨性系YouTube頻道	哈哈哈哈，沒錯我就是親友團來介紹一個我覺得很北七的頻道，現在觀看真的低的可憐，也沒事啦，就多...	YouTuber
330	《庫洛魔法使》（迷你）服裝製作	又來跟大家分享新的作品了~，頻道常常分享 {縫紉} {服裝製作} 等相關教學，大家對服裝製...	YouTuber
342	自己沒搞清楚狀況就不要亂黑勾惡	勾惡幫主在自己頻道簡介跟每部影片的下方都已經說明了，要分會會長以上才能看全部影片，這個說明已...	YouTuber
338	廚師系YouTuber	友人傳了這篇文給我，我一看，十大廚師系YouTuber，就猜一定有MASA，果不其然，榜上有...	YouTuber
243	毀我童年的家人	小時候都很喜歡看真珠美人魚和守護甜心，但是！！，每次晚餐看電視的時候，只要有播映到這種場景....	有趣
349	喜歡看寵物頻道的有嗎？🙋		YouTuber

Implemement Your kNN or SVM classifier Here!

請比較分類結果中選出 topk 相近的筆數，並計算 forum_zh 是否都有在 query text 的 forum_zh 中

> [開了新頻道] [Youtuber | 頻道 | 有趣 | 日常 | 搞笑]

```
precision = 0
topk = 10

# YOUR CODE HERE!
# IMPLEMENTIG TRIE IN PYTHON


import tensorflow_hub as hub
import numpy as np
import tensorflow_text
import faiss
from sklearn.cluster import KMeans

# 载入 Universal Sentence Encoder
embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")

# 将文本转换为向量
texts = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']
embeddings = embed_model(texts)
embed_arrays = np.array(embeddings)

# 运行 k-means
num_clusters = 10
kmeans = KMeans(n_clusters=num_clusters, random_state=42).fit(embed_arrays)

# 获取每个文本所属的类别
labels = kmeans.labels_

# 创建一个字典，将每个类别对应的文本索引存储在其中
clusters = {}
for i, label in enumerate(labels):
    if label not in clusters:
        clusters[label] = []
    clusters[label].append(i)

# 对每个查询进行处理
docid = 355
query = texts[docid]

# 获取与查询相似的文本
query_embedding = embed_model([query])[0]
query_embedding_array = np.array([query_embedding])

# 在 Faiss 中进行相似度搜索
index = faiss.IndexFlatIP(embeddings.shape[1])
index.add(embed_arrays)
D, I = index.search(query_embedding_array, k=10)

# 计算查询的类别
query_label = kmeans.predict(query_embedding_array)[0]

# 获取相似的文本并计算准确率
precision = 0
for i in range(10):
    index = I[0][i]
    label = kmeans.labels_[index]
    if label == query_label:
        precision += 1




# # DO NOT MODIFY THE BELOW LINE!
print("precision = ", precision/topk)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto
  warnings.warn(
precision =  0.9
```

```
docid = 355
texts = "[" + df['title'] + '] [' + df['topics'] + '] '
texts[docid]
```

```
'[開了新頻道] [Youtuber | 頻道 | 有趣 | 日常 | 搞笑] '
```

```
!pip install pytrie
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pytrie
  Downloading PyTrie-0.4.0.tar.gz (95 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 95.1/95.1 kB 7.1 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: sortedcontainers in /usr/local/lib/python3.9/dist-packages (from pytrie) (2.4.0)
Building wheels for collected packages: pytrie
  Building wheel for pytrie (setup.py) ... done
  Created wheel for pytrie: filename=PyTrie-0.4.0-py3-none-any.whl size=6104 sha256=522225411198cd7c533ea098c4b50e6c5749c9349df22bdba8fb2591bc0d
  Stored in directory: /root/.cache/pip/wheels/47/4f/6b/8b01679863e246ca7b7e9e572ba9aef8cd7b76bc3c9fc0b9ba
Successfully built pytrie
Installing collected packages: pytrie
Successfully installed pytrie-0.4.0
```

```
!pip install pygtrie
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pygtrie
  Downloading pygtrie-2.5.0-py3-none-any.whl (25 kB)
Installing collected packages: pygtrie
Successfully installed pygtrie-2.5.0
```

```
import numpy as np
import pandas as pd
import sqlite3
import tensorflow_hub as hub
import pygtrie as trie
from sklearn.neighbors import KNeighborsClassifier

# 读取数据
conn = sqlite3.connect("Dcard.db")
df = pd.read_sql("SELECT * FROM Posts;", conn)
texts = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']

# 加载预训练模型
embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")

# 使用Trie数据结构存储文本数据
trie_dict = trie.StringTrie(separator=' ')
for i, text in texts.iteritems():
        trie_dict[text] = i

# 定义查询文本和docid
docid = 355
query_text = texts[docid]

# 在Trie中寻找与查询文本相似的前10个文本的索引
similar_indices = [index for _, index in trie_dict.items(prefix=query_text, max_matches=10)]

# 提取相似文本的标签（即forum_zh）
similar_labels = df.iloc[similar_indices]['forum_zh'].values

# 对训练集进行编码，训练KNN分类器
train_embeddings = embed_model(texts)
train_labels = df['forum_zh'].values
n_neighbors = 10
knn = KNeighborsClassifier(n_neighbors=n_neighbors)
knn.fit(train_embeddings, train_labels)

# 对查询文本进行编码，进行预测
test_embeddings = embed_model([query_text])
pred_labels = knn.predict(test_embeddings)

# 计算精度
precision = 0
for i, label in enumerate(similar_labels):
        if label == pred_labels:
                precision += 1
print(f"precision = {precision/n_neighbors}")
```

```
<ipython-input-39-c9cd10904dcf>:18: FutureWarning: iteritems is deprecated and will be removed in a future vers
  for i, text in texts.iteritems():
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-39-c9cd10904dcf> in <cell line: 26>()
     24
     25 # 在Trie中寻找与查询文本相似的前10个文本的索引
---> 26 similar_indices = [index for _, index in trie_dict.items(prefix=query_text, max_matches=10)]
     27
     28 # 提取相似文本的标签（即forum_zh）

TypeError: items() got an unexpected keyword argument 'max_matches'
```

SEARCH STACK OVERFLOW

Colab 付費產品 - 按這裡取消合約

4 秒 完成時間：晚上11:19

無法連至 reCAPTCHA 服務。請檢查你的網際網路連線，並重新載入頁面以取得 reCAPTCHA 驗證問題。