Zero Shot Topic Classification

January 3, 2022

Zero Shot Learning for Topic Modelling with Keywords

In this notebook we will be performing the task of assigning label names to keywords by using Zero Shot Learning. The approach has been tested on different datasets and the results generated as keywords with their assigned labl names with their corresponding accuracy score.

This project/notebook consists of several Tasks.

- **Task 1**: Importing the required libraries in the environment.
- **Task**: Instantiating the classifier by using huggingface pipeline
- **Task 4**: Forming Class Names to which the keywords will be assigned to
- **Task 5**: Passing the keywords and the class names through the classifier
- **Task 6**: Analysis of the labels assigned to keywords

0.0.1 Task 1: Importing the required libraries in the environment.

[1]:
```
#Importing the necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import classification_report, confusion_matrix

import transformers
from transformers import pipeline

import nltk

import warnings
warnings.filterwarnings("ignore")
```

```
2021-10-25 20:32:17.729784: W
tensorflow/stream_executor/platform/default/dso_loader.cc:59] Could not load
dynamic library 'libcudart.so.10.1'; dlerror: libcudart.so.10.1: cannot open
shared object file: No such file or directory
2021-10-25 20:32:17.729849: I tensorflow/stream_executor/cuda/cudart_stub.cc:29]
Ignore above cudart dlerror if you do not have a GPU set up on your machine.
```

[2]:
```
print(transformers.__version__)
```

4.10.2

0.0.2 Configure the FB-Bart-Large-mnli model pipeline

This transformer has been trained on a massily amount of data so already understands loads about the structure of text.

[11]:
```
classifier = pipeline('zero-shot-classification', model = 'facebook/
↪bart-large-mnli')
```

0.0.3 Define the class Labels

We'll be using `Bart for multi-class text classification.` To perform, we first need to provide it with a list of classes and it will figure our which keywords should be assigned to which class (intent). It does this by presenting each candidate label as a hypothesis to the model, with the sequence text representing the "premise."

To come up with labels, we can do EDA, but for starting, we can try the official intents i.e., Informational, Navigational, Transactional, Commercial

[ ]:
```
df = pd.read_csv('df.csv')
df = df[["Keyword","Label"]]
df.head()
```

[ ]:
```
df.shape
```

[ ]:
```
fig=plt.figure(figsize=(10, 5))

plt.hist(df.Label,color = "skyblue", lw=0)
plt.xlabel('Types of Labels')
plt.ylabel('Number of Instances')
plt.title('Distribution of Label in Dataset by SEO');
```

[ ]:
```
df_copy = df.copy()
df_copy = df_copy[["Keyword"]]
df_copy.head()
```

0.0.4 Task 4: Forming Class Names to which the keywords will be assigned to

[43]:
```
classes = ['Informational',
           'Local',
           'Transactional',
           'Navigational']
```

[44]:
```
keyword = df_copy['Keyword'][0]
```

[ ]:
```
result = classifier(keyword, classes, multi_label=False)
result
```

0.0.5 Task 5: Passing the keywords and the class names through the classifier

[46]:
```
df_copy['labels'] = df_copy.apply(lambda x: classifier(x.Keyword, classes,␣
↪multi_label=False), axis=1)
```

[47]:
```
df_copy['predicted_label'] = df_copy.apply(lambda row:␣
↪row['labels']['labels'][0], axis = 1)
df_copy['score'] = df_copy.apply(lambda row: row['labels']['scores'][0], axis=1)
```

[ ]:
```
df_copy.head(10)
```

0.0.6 Task 6: Analysis of the labels assigned to keywords

[ ]:
```
result = pd.merge(df, df_copy, on='Keyword', how='inner')
result = result[['Keyword','Label','predicted_label','score']]
#result = result.groupby('predicted_label').head(20).reset_index(drop=False)
result.head(20)
```

[ ]:
```
result.loc[result['predicted_label'] == 'Local'].head(7)
#result.loc[result['column_name'] == some_value]
```

[ ]:
```
result.loc[result['predicted_label'] == 'Navigational'].head(7)
```