

```
import pandas as pd
import numpy as np
import os
```

```
!pip install -U sentence-transformers
```

```
Collecting sentence-transformers
```

```
  Downloading sentence_transformers-2.3.1-py3-none-any.whl (132 kB)
```

```
132.8/132.8 kB 4.1 MB/s eta 0:00:00
```

```
Requirement already satisfied: transformers<5.0.0,>=4.32.0 in /usr/local/lib/python3.10/dist-packages (from sentence-tra
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (4.66.2)
Requirement already satisfied: torch>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (2.
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (1.25.2)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (1.2
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (1.11.4)
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (3.8.1)
Requirement already satisfied: sentencepiece in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (0.
Requirement already satisfied: huggingface-hub>=0.15.1 in /usr/local/lib/python3.10/dist-packages (from sentence-transfo
Requirement already satisfied: Pillow in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (9.4.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.15.1->sente
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.15.1
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.15.1->sente
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.15.1->sen
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-h
Requirement already satisfied: packaging>=20.9 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.15.1-
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch>=1.11.0->sentence-transforme
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.11.0->sentence-transfo
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.11.0->sentence-transform
Requirement already satisfied: triton==2.1.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.11.0->sentence-tr
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers<5.0.0,>=4
Requirement already satisfied: tokenizers<0.19,>=0.14 in /usr/local/lib/python3.10/dist-packages (from transformers<5.0.
Requirement already satisfied: safetensors>=0.4.1 in /usr/local/lib/python3.10/dist-packages (from transformers<5.0.0,>=
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk->sentence-transformers) (8.1.
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk->sentence-transformers) (1.3
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->sente
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.11.0->s
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->huggi
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface-hub>=
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.11.0->sente
Installing collected packages: sentence-transformers
Successfully installed sentence-transformers-2.3.1
```

```
from sentence_transformers import SentenceTransformer, InputExample, losses, evaluation, util, models
from torch.utils.data import DataLoader
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from torch import nn
```

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Unzipping tokenizers/punkt.zip.
True
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```

import pandas as pd

# Read and store content
# of an excel file
read_file = pd.read_excel ("/content/drive/MyDrive/analysis_results.xlsx")

# Write the dataframe object
# into csv file
read_file.to_csv ("/content/drive/MyDrive/filter.csv",
                  index = None,
                  header=True)

# read csv file and convert
# into a dataframe object
df = pd.DataFrame(pd.read_csv("/content/drive/MyDrive/filter.csv"))

# show the dataframe
df

```

	Questions	Answer	score	ViLT	ViLT score	Filter	Filter Score
0	How many vehicles are there?	3	1	1	0	0	0.0
1	Which camera is this image from?	Back Camera	1	front	0	unknown	0.5
2	Are there any vehicles in ego lane?	No	1	Yes	0	No	1.0
3	Is it safe to initiate a lane change?	unable to tell	1	Yes	0	No	0.0
4	Which camera is this image from?	unable to tell	1	front	0	top	0.0
5	Is it okay to initiate a lane change?	No	1	Yes	0	No	1.0
6	Are there any vehicles coming behind?	unable to tell	1	Yes	0	No	0.0
7	Which street is this?	Summer Street	1	unknown	0	unknown	0.0
8	Which camera is this image from?	Back Right	1	unknown	0	unknown	0.0
9	Do I need to stop?	No	1	No	1	No	1.0
10	Are there any pedestrians on the sidewalk ?	No	1	No	1	No	1.0
11	Can I park on the right?	No	1	No	1	No	1.0
12	Which camera is this image from?	Front Camera	1	front	1	top	0.5
13	Is there snow on the road?	Yes	1	No	0	No	0.0
14	Are there any pedestrians?	Yes	1	No	0	No	0.0
15	Can I go right in this	Yes	1	No	0	No	0.0

```
model1 =SentenceTransformer('nli-distilbert-base')
```

```

/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:88: U:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tal
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access pi
warnings.warn(
modules.json: 100%                229/229 [00:00<00:00, 8.64kB/s]
config_sentence_transformers.json: 100%          122/122 [00:00<00:00, 4.58kB/s]
README.md: 100%                    3.96k/3.96k [00:00<00:00, 147kB/s]
sentence_bert_config.json: 100%              53.0/53.0 [00:00<00:00, 1.95kB/s]
config.json: 100%                   538/538 [00:00<00:00, 18.1kB/s]
pytorch_model.bin: 100%              265M/265M [00:05<00:00, 49.4MB/s]
/usr/local/lib/python3.10/dist-packages/torch/_utils.py:831: UserWarning: Typ
return self.fget.__get__(instance, owner)()
tokenizer_config.json: 100%           426/426 [00:00<00:00, 19.5kB/s]
vocab.txt: 100%                     232k/232k [00:00<00:00, 4.27MB/s]
tokenizer.json: 100%                 466k/466k [00:00<00:00, 21.0MB/s]
special_tokens_map.json: 100%         112/112 [00:00<00:00, 5.79kB/s]
1_Pooling/config.json: 100%          190/190 [00:00<00:00, 5.54kB/s]

```

## ✓ Function for Embedding Generation and Cosine Similarity

```

def pred(model,q_s, ans):
    try:
        return util.cos_sim(model.encode(q_s), model.encode(ans))[0].cpu().item()
    except:
        return 0.0

df['predict_nli_LXMERT'] = df.apply(lambda row: pred(model1, row['Answer'],row['ViLT']), axis=1)
df['predict_nli_filter'] = df.apply(lambda row: pred(model1, row['Answer'],row['Filter']), axis=1)

df

```

	Questions	Answer	score	ViLT	ViLT score	Filter	Filter Score	predict_nli_LXMEI
0	How many vehicles are there?	3	1	1	0	0	0.0	0.7969
1	Which camera is this image from?	Back Camera	1	front	0	unknown	0.5	0.7503
2	Are there any vehicles in ego lane?	No	1	Yes	0	No	1.0	0.7635
3	Is it safe to initiate a lane change?	unable to tell	1	Yes	0	No	0.0	0.6435
4	Which camera is this image from?	unable to tell	1	front	0	top	0.0	0.6305
5	Is it okay to initiate a lane change?	No	1	Yes	0	No	1.0	0.7635
6	Are there any vehicles coming behind?	unable to tell	1	Yes	0	No	0.0	0.6435
7	Which street is this?	Summer Street	1	unknown	0	unknown	0.0	0.4226
8	Which camera is this image from?	Back Right	1	unknown	0	unknown	0.0	0.6958
9	Do I need to stop?	No	1	No	1	No	1.0	1.0000

```
def pearson_correlation(x,y):
    mean_x = sum(x) / len(x)
    mean_y = sum(y) / len(y)
    cov = sum((a - mean_x) * (b - mean_y) for (a, b) in zip(x, y)) / len(x)

    std_x, std_y = np.std(x), np.std(y)

    p = cov / (std_x * std_y)

    return float(p)

print("Pearson Correlation for LXMERT:", pearson_correlation(df['ViLT score'], df['predict_nli_LXMERT']))
print("Pearson Correlation for Filter:", pearson_correlation(df['Filter Score'], df['predict_nli_filter']))

Pearson Correlation for LXMERT: 0.7557564290096992
Pearson Correlation for Filter: 0.7503914226376601
```

```
model2= SentenceTransformer('all-mpnet-base-v2') #This one is identified as one of the best performing model in SBERT docume
```

modules.json: 100%	349/349 [00:00<00:00, 18.8kB/s]
config_sentence_transformers.json: 100%	116/116 [00:00<00:00, 5.01kB/s]
README.md: 100%	10.6k/10.6k [00:00<00:00, 437kB/s]
sentence_bert_config.json: 100%	53.0/53.0 [00:00<00:00, 3.29kB/s]
config.json: 100%	571/571 [00:00<00:00, 33.1kB/s]
pytorch_model.bin: 100%	438M/438M [00:05<00:00, 124MB/s]
tokenizer_config.json: 100%	363/363 [00:00<00:00, 12.0kB/s]
vocab.txt: 100%	232k/232k [00:00<00:00, 5.04MB/s]
tokenizer.json: 100%	466k/466k [00:00<00:00, 18.2MB/s]
special_tokens_map.json: 100%	239/239 [00:00<00:00, 17.0kB/s]
1_Pooling/config.json: 100%	190/190 [00:00<00:00, 10.3kB/s]

```
df['predict_mpnet_LXMERT'] = df.apply(lambda row: pred(model2, row['Answer'],row['ViLT']), axis=1)
df['predict_mpnet_filter'] = df.apply(lambda row: pred(model2, row['Answer'],row['Filter']), axis=1)
```

```
df
```

	Questions	Answer	score	ViLT	ViLT score	Filter	Filter Score	predict_nli_LXMEI
0	How many vehicles are there?	3	1	1	0	0	0.0	0.7969
1	Which camera is this image from?	Back Camera	1	front	0	unknown	0.5	0.7503
2	Are there any vehicles in ego lane?	No	1	Yes	0	No	1.0	0.7635
3	Is it safe to initiate a lane change?	unable to tell	1	Yes	0	No	0.0	0.6435
4	Which camera is this image from?	unable to tell	1	front	0	top	0.0	0.6305
5	Is it okay to initiate a lane change?	No	1	Yes	0	No	1.0	0.7635
6	Are there any vehicles coming behind?	unable to tell	1	Yes	0	No	0.0	0.6435
7	Which street is this?	Summer Street	1	unknown	0	unknown	0.0	0.4226
8	Which camera is this image from?	Back Right	1	unknown	0	unknown	0.0	0.6958
9	Do I need to stop?	No	1	No	1	No	1.0	1.0000

```
print("Pearson Correlation for LXMERT:", pearson_correlation(df['ViLT score'], df['predict_mpnet_LXMERT']))
print("Pearson Correlation for Filter:", pearson_correlation(df['Filter Score'], df['predict_mpnet_filter']))
```

```
Pearson Correlation for LXMERT: 0.623083262742679
Pearson Correlation for Filter: 0.736992778735498
```

## Basic BERT Model

```
word_embedding_model = models.Transformer('bert-base-uncased', max_seq_length=256)
pooling_model = models.Pooling(word_embedding_model.get_word_embedding_dimension())
dense_model = models.Dense(in_features=pooling_model.get_sentence_embedding_dimension(), out_features=256, activation_functi
```

```
model3 = SentenceTransformer(modules=[word_embedding_model, pooling_model, dense_model])
```

```
config.json: 100% 570/570 [00:00<00:00, 6.13kB/s]
model.safetensors: 100% 440M/440M [00:07<00:00, 94.8MB/s]
tokenizer_config.json: 100% 48.0/48.0 [00:00<00:00, 817B/s]
vocab.txt: 100% 232k/232k [00:00<00:00, 2.48MB/s]
tokenizer.json: 100% 466k/466k [00:00<00:00, 2.32MB/s]
```

```
df['bert_LXMERT'] = df.apply(lambda row: pred(model3, row['Answer'],row['ViLT']), axis=1)
df['bert_filter'] = df.apply(lambda row: pred(model3, row['Answer'],row['Filter']), axis=1)
```

df

	Questions	Answer	score	ViLT	ViLT score	Filter	Filter Score	predict_nli_LXMEI
0	How many vehicles are there?	3	1	1	0	0	0.0	0.7969
1	Which camera is this image from?	Back Camera	1	front	0	unknown	0.5	0.7503
2	Are there any vehicles in ego lane?	No	1	Yes	0	No	1.0	0.7635
3	Is it safe to initiate a lane change?	unable to tell	1	Yes	0	No	0.0	0.6435
4	Which camera is this image from?	unable to tell	1	front	0	top	0.0	0.6305
5	Is it okay to initiate a lane change?	No	1	Yes	0	No	1.0	0.7635
6	Are there any vehicles coming behind?	unable to tell	1	Yes	0	No	0.0	0.6435
7	Which street is this?	Summer Street	1	unknown	0	unknown	0.0	0.4226
8	Which camera is this image from?	Back Right	1	unknown	0	unknown	0.0	0.6958
9	Do I need to stop?	No	1	No	1	No	1.0	1.0000

```
print("Pearson Correlation for LXMERT:", pearson_correlation(df['ViLT score'], df['bert_LXMERT']))
print("Pearson Correlation for Filter:", pearson_correlation(df['Filter Score'], df['bert_filter']))
```

```
Pearson Correlation for LXMERT: 0.5479966159038716
Pearson Correlation for Filter: 0.6968016248794486
```

## ✓ GPT2 Model -- GPT3 is not freely available

```
!pip install openai
```

```
Collecting openai
  Downloading openai-1.12.0-py3-none-any.whl (226 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 226.7/226.7 kB 3.8 MB/s eta 0:00:00
Requirement already satisfied: anyio<5,>=3.5.0 in /usr/local/lib/python3.10/dist-packages (from openai) (3.7.1)
Requirement already satisfied: distro<2,>=1.7.0 in /usr/lib/python3/dist-packages (from openai) (1.7.0)
Collecting httpx<1,>=0.23.0 (from openai)
  Downloading httpx-0.27.0-py3-none-any.whl (75 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 75.6/75.6 kB 6.7 MB/s eta 0:00:00
Requirement already satisfied: pydantic<3,>=1.9.0 in /usr/local/lib/python3.10/dist-packages (from openai) (2.6.1)
Requirement already satisfied: sniffio in /usr/local/lib/python3.10/dist-packages (from openai) (1.3.0)
Requirement already satisfied: tqdm>4 in /usr/local/lib/python3.10/dist-packages (from openai) (4.66.2)
Requirement already satisfied: typing-extensions<5,>=4.7 in /usr/local/lib/python3.10/dist-packages (from openai) (4.9.0)
Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.10/dist-packages (from anyio<5,>=3.5.0->openai) (3.6)
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-packages (from anyio<5,>=3.5.0->openai)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from httpx<1,>=0.23.0->openai) (2024)
Collecting httpcore==1.* (from httpx<1,>=0.23.0->openai)
  Downloading httpcore-1.0.4-py3-none-any.whl (77 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 77.8/77.8 kB 7.8 MB/s eta 0:00:00
Collecting h11<0.15,>=0.13 (from httpcore==1.*->httpx<1,>=0.23.0->openai)
  Downloading h11-0.14.0-py3-none-any.whl (58 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.3/58.3 kB 5.9 MB/s eta 0:00:00
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1.9)
Requirement already satisfied: pydantic-core==2.16.2 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1.9.0)
Installing collected packages: h11, httpcore, httpx, openai
Successfully installed h11-0.14.0 httpcore-1.0.4 httpx-0.27.0 openai-1.12.0
```

```

import torch
import pandas as pd
from transformers import GPT2Tokenizer, GPT2Model
from sklearn.metrics.pairwise import cosine_similarity

# Load the pre-trained GPT-2 model and tokenizer
model_name = 'gpt2'
tokenizer = GPT2Tokenizer.from_pretrained(model_name)
model = GPT2Model.from_pretrained(model_name)

# Generate embeddings for the text pairs in the DataFrame
embeddings_list = []

for _, row in df.iterrows():
    text1 = row['Answer']
    text2 = row['ViLT']

    # Tokenize and encode the texts using the GPT-2 tokenizer
    inputs1 = tokenizer(text1, return_tensors='pt', padding=False, truncation=True)
    inputs2 = tokenizer(text2, return_tensors='pt', padding=False, truncation=True)

    # Generate embeddings for the texts using GPT-2
    with torch.no_grad():
        embeddings1 = model(**inputs1).last_hidden_state.mean(dim=1)
        embeddings2 = model(**inputs2).last_hidden_state.mean(dim=1)

    embeddings_list.append((embeddings1, embeddings2))

# Add embeddings to the DataFrame
df['embeddings1'], df['embeddings2'] = zip(*embeddings_list)

# Calculate cosine similarity between the embeddings and store it in a new column
cosine_similarities = []

for _, row in df.iterrows():
    similarity = cosine_similarity(row['embeddings1'].numpy(), row['embeddings2'].numpy())
    cosine_similarities.append(similarity[0][0])

df.drop(columns=['embeddings1', 'embeddings2'], inplace=True)

df['gpt2_LXMERT'] = cosine_similarities

# The DataFrame now contains text embeddings and cosine similarity values

embeddings_list1 = []

for _, row in df.iterrows():
    text1 = row['Answer']
    text2 = row['Filter']

    # Tokenize and encode the texts using the GPT-2 tokenizer
    inputs1 = tokenizer(text1, return_tensors='pt', padding=False, truncation=True)
    inputs2 = tokenizer(text2, return_tensors='pt', padding=False, truncation=True)

    # Generate embeddings for the texts using GPT-2
    with torch.no_grad():
        embeddings1 = model(**inputs1).last_hidden_state.mean(dim=1)
        embeddings2 = model(**inputs2).last_hidden_state.mean(dim=1)

    embeddings_list1.append((embeddings1, embeddings2))

# Add embeddings to the DataFrame
df['embeddings1'], df['embeddings2'] = zip(*embeddings_list1)

# Calculate cosine similarity between the embeddings and store it in a new column
cosine_similarities1 = []

for _, row in df.iterrows():
    similarity = cosine_similarity(row['embeddings1'].numpy(), row['embeddings2'].numpy())
    cosine_similarities1.append(similarity[0][0])

df.drop(columns=['embeddings1', 'embeddings2'], inplace=True)

df['gpt2_filter'] = cosine_similarities1

# The DataFrame now contains text embeddings and cosine similarity values

```

df

	Questions	Answer	score	ViLT	ViLT score	Filter	Filter Score	predict_nli_LXMEI
0	How many vehicles are there?	3	1	1	0	0	0.0	0.7969
1	Which camera is this image from?	Back Camera	1	front	0	unknown	0.5	0.7503
2	Are there any vehicles in ego lane?	No	1	Yes	0	No	1.0	0.7635
3	Is it safe to initiate a lane change?	unable to tell	1	Yes	0	No	0.0	0.6435
4	Which camera is this image from?	unable to tell	1	front	0	top	0.0	0.6305
5	Is it okay to initiate a lane change?	No	1	Yes	0	No	1.0	0.7635
6	Are there any vehicles coming behind?	unable to tell	1	Yes	0	No	0.0	0.6435
7	Which street is this?	Summer Street	1	unknown	0	unknown	0.0	0.4226
8	Which camera is this image from?	Back Right	1	unknown	0	unknown	0.0	0.6958
9	Do I need to stop?	No	1	No	1	No	1.0	1.0000
10	Are there any pedestrians on the sidewalk ?	No	1	No	1	No	1.0	1.0000
11	Can I park on the right?	No	1	No	1	No	1.0	1.0000
12	Which camera is this image from?	Front Camera	1	front	1	top	0.5	0.9197
13	Is there snow on the road?	Yes	1	No	0	No	0.0	0.7635

```
print("Pearson Correlation for LXMERT:", pearson_correlation(df['ViLT score'], df['gpt2_LXMERT']))
print("Pearson Correlation for Filter:", pearson_correlation(df['Filter Score'], df['gpt2_filter']))
```

```
Pearson Correlation for LXMERT: 0.32197969843309704
Pearson Correlation for Filter: 0.47371799388021146
```

## ✓ Mean Absolute Error

```
from sklearn.metrics import mean_absolute_error
```

```
print(" For NLI Model ")
print("MAE for LXMERT:", mean_absolute_error(df['ViLT score'],df['predict_nli_LXMERT']))
print("MAE for Filter:", mean_absolute_error(df['Filter Score'],df['predict_nli_filter']))
```

```
For NLI Model
MAE for LXMERT: 0.5659514913956324
```

```
MAE for Filter: 0.4941691334048907
```

```
print(" For mpnet Model ")  
print("MAE for LXMERT:", mean_absolute_error(df['ViLT score'],df['predict_mpnet_LXMERT']))  
print("MAE for Filter:", mean_absolute_error(df['Filter Score'],df['predict_mpnet_filter']))
```

```
For mpnet Model  
MAE for LXMERT: 0.3989448168625434  
MAE for Filter: 0.2802131517479817
```

```
print(" For BERT Model ")  
print("MAE for LXMERT:", mean_absolute_error(df['ViLT score'],df['bert_LXMERT']))  
print("MAE for Filter:" mean_absolute_error(df['Filter Score'] df['bert filter']))
```

**Please note that we also provided an anonymised copy of the Subjective Scoring Framework's paper that is under IEEE review process for reference.**