

infer_01_er_siamese_bert_cos_sim_0.06251

April 27, 2023

1 Model 01 inference

Evidence retrieval using a Siamese BERT classification model.

Ref: - [STS continue training guide](#)

1.1 Setup

1.1.1 Working Directory

```
[ ]: # Change the working directory to project root
import pathlib
import os
ROOT_DIR = pathlib.Path.cwd()
while not ROOT_DIR.joinpath("src").exists():
    ROOT_DIR = ROOT_DIR.parent
os.chdir(ROOT_DIR)
```

1.1.2 File paths

```
[ ]: MODEL_PATH = ROOT_DIR.joinpath("./result/models/*")
OUTPUT_PATH = ROOT_DIR.joinpath("./result/inference")
```

1.1.3 Dependencies

```
[ ]: # Imports and dependencies
import torch
from sentence_transformers import SentenceTransformer, LoggingHandler, util
from src.torch_utils import get_torch_device
from src.data import load_from_json
from src.model_01 import run_inference
import logging
import random
random.seed(a=42)

torch_device = get_torch_device()
```

```
/opt/homebrew/Caskroom/miniconda/base/envs/comp90042_project/lib/python3.8/site-  
packages/tqdm/auto.py:21: TqdmWarning: IProgress not found. Please update  
jupyter and ipywidgets. See  
https://ipywidgets.readthedocs.io/en/stable/user\_install.html  
    from .autonotebook import tqdm as notebook_tqdm
```

Torch device is 'mps'

1.1.4 Names

```
[ ]: model_save_path = MODEL_PATH.with_name(f"model_01_base_e5_equal_neg")  
    inference_output_path = OUTPUT_PATH.joinpath(model_save_path.name)
```

1.1.5 Logging

```
[ ]: logging.basicConfig(format='%(asctime)s - %(message)s',  
    datefmt='%Y-%m-%d %H:%M:%S',  
    level=logging.INFO,  
    handlers=[LoggingHandler()]  
)
```

1.2 Dataset

```
[ ]: data_names = ["train-claims", "dev-claims", "test-claims-unlabelled",  
    ↪ "evidence"]  
    train_claims, dev_claims, test_claims, all_evidence = load_from_json(data_names)
```

```
Loaded train-claims  
Loaded dev-claims  
Loaded test-claims-unlabelled  
Loaded evidence  
Loaded evidence
```

```
[ ]: print(len(test_claims))  
    print(len(dev_claims))  
    print(len(all_evidence))
```

```
153  
154  
1208827
```

As `all_evidence` exceeds maximum size limit for `tensor.save`, we will test with a reduced set for now.

```
[ ]: # Extract a set of named evidence ids  
    related_evidence_ids = set()  
    for dataset in [train_claims, dev_claims]:  
        for claim in dataset.values():
```

```

        related_evidence_ids.update(set(claim["evidences"]))
len(related_evidence_ids)

```

[]: 3443

```

[ ]: random_evidence_ids = random.sample(
    population=set(all_evidence.keys()),
    k=5000
)
len(random_evidence_ids)

```

[]: 5000

```

[ ]: evidence_lib_ids = related_evidence_ids.union(random_evidence_ids)
len(evidence_lib_ids)

```

[]: 8432

```

[ ]: reduced_evidence = {k:v for k, v in all_evidence.items() if k in
    ↳evidence_lib_ids}

```

1.3 Select load model from file

```

[ ]: model = SentenceTransformer(
    model_name_or_path=model_save_path,
    device=torch_device
)
model

```

2023-04-27 22:01:29 - Load pretrained SentenceTransformer: /Users/johnsonzhou/git/comp90042-project/result/models/model_01_base_e5_equal_neg

```

[ ]: SentenceTransformer(
  (0): Transformer({'max_seq_length': 512, 'do_lower_case': False}) with
  Transformer model: BertModel
  (1): Pooling({'word_embedding_dimension': 768, 'pooling_mode_cls_token':
  False, 'pooling_mode_mean_tokens': True, 'pooling_mode_max_tokens': False,
  'pooling_mode_mean_sqrt_len_tokens': False})
)

```

1.4 Run inference

```

[ ]: run_inference(
    name="dev",
    model=model,
    claims=dev_claims,
    evidence=reduced_evidence,

```

```
scorer=util.cos_sim,  
threshold=0.6251,  
output_path=inference_output_path,  
batch_size=64,  
device=torch_device,  
verbose=True  
)
```

```
Generate claim embeddings n=154  
Loaded claim embeddings from file  
Generate evidence embeddings n=8432  
Batches: 100%|      | 132/132 [00:14<00:00, 8.82it/s]  
Saved evidence embeddings to file  
Calculate scores  
Retrieve top scoring evidences  
claims: 154it [00:00, 966.57it/s]  
Average retrievals = 507.000000  
Done!
```