Search index

FAISS and Elasticsearch enables searching for examples in a dataset. This can be useful when you want to retrieve specific examples from a dataset that are relevant to your NLP task. For example, if you are working on an Open Domain Question Answering task, you may want to only return examples that are relevant to answering your question.

This guide will show you how to build an index for your dataset that will allow you to search it.

FAISS

FAISS retrieves documents based on the similarity of their vector representations. In this example, you will generate the vector representations with the DPR model.

1. Download the DPR model from 🤗 Transformers:

```
>>> from transformers import DPRContextEncoder, DPRContextEncoderTokenizer
>>> import torch
>>> torch.set_grad_enabled(False)
>>> ctx_encoder = DPRContextEncoder.from_pretrained("facebook/dpr-ctx_encoder-single-nq-base")
>>> ctx_tokenizer = DPRContextEncoderTokenizer.from_pretrained("facebook/dpr-ctx_encoder-single-nq
```

2. Load your dataset and compute the vector representations:

```
>>> from datasets import load_dataset
>>> ds = load_dataset('crime_and_punish', split='train[:100]')
>>> ds_with_embeddings = ds.map(lambda example: {'embeddings': ctx_encoder(**ctx_tokenizer(example
```

3. Create the index with Dataset.add_faiss_index():

```
>>> ds_with_embeddings.add_faiss_index(column='embeddings')
```

4. Now you can query your dataset with the `embeddings` index. Load the DPR Question Encoder, and search for a question with Dataset.get_nearest_examples():

```
>>> from transformers import DPRQuestionEncoder, DPRQuestionEncoderTokenizer
>>> q_encoder = DPRQuestionEncoder.from_pretrained("facebook/dpr-question_encoder-single-nq-base")
>>> q_tokenizer = DPRQuestionEncoderTokenizer.from_pretrained("facebook/dpr-question_encoder-singl

>>> question = "Is it serious ?"
>>> question_embedding = q_encoder(**q_tokenizer(question, return_tensors="pt"))[0][0].numpy()
>>> scores, retrieved_examples = ds_with_embeddings.get_nearest_examples('embeddings', question_em
>>> retrieved_examples["line"][0]
'_that_ serious? It is not serious at all. It’s simply a fantasy to amuse\r\n'
```

5. You can access the index with Dataset.get_index() and use it for special operations, e.g. query it using `range_search`:

```
>>> faiss_index = ds_with_embeddings.get_index('embeddings').faiss_index
>>> limits, distances, indices = faiss_index.range_search(x=question_embedding.reshape(1, -1), thr
```

6. When you are done querying, save the index on disk with Dataset.save_faiss_index():

```
>>> ds_with_embeddings.save_faiss_index('embeddings', 'my_index.faiss')
```

7. Reload it at a later time with Dataset.load_faiss_index():

```
>>> ds = load_dataset('crime_and_punish', split='train[:100]')
>>> ds.load_faiss_index('embeddings', 'my_index.faiss')
```

Elasticsearch

Unlike FAISS, Elasticsearch retrieves documents based on exact matches.

Start Elasticsearch on your machine, or see the Elasticsearch installation guide if you don't already have it installed.

1. Load the dataset you want to index:

```
>>> from datasets import load_dataset
>>> squad = load_dataset('rajpurkar/squad', split='validation')
```

2. Build the index with Dataset.add_elasticsearch_index():

```
>>> squad.add_elasticsearch_index("context", host="localhost", port="9200")
```

3. Then you can query the `context` index with Dataset.get_nearest_examples():

```
>>> query = "machine"
>>> scores, retrieved_examples = squad.get_nearest_examples("context", query, k=10)
>>> retrieved_examples["title"][0]
'Computational_complexity_theory'
```

4. If you want to reuse the index, define the `es_index_name` parameter when you build the index:

```
>>> from datasets import load_dataset
>>> squad = load_dataset('rajpurkar/squad', split='validation')
>>> squad.add_elasticsearch_index("context", host="localhost", port="9200", es_index_name="hf_squa
>>> squad.get_index("context").es_index_name
hf_squad_val_context
```

5. Reload it later with the index name when you call Dataset.load_elasticsearch_index():

```
>>> from datasets import load_dataset
>>> squad = load_dataset('rajpurkar/squad', split='validation')
>>> squad.load_elasticsearch_index("context", host="localhost", port="9200", es_index_name="hf_squ
>>> query = "machine"
>>> scores, retrieved_examples = squad.get_nearest_examples("context", query, k=10)
```

For more advanced Elasticsearch usage, you can specify your own configuration with custom settings:

```
>>> import elasticsearch as es
>>> import elasticsearch.helpers
>>> from elasticsearch import Elasticsearch
>>> es_client = Elasticsearch([{"host": "localhost", "port": "9200"}])  # default client
>>> es_config = {
...     "settings": {
...         "number_of_shards": 1,
...         "analysis": {"analyzer": {"stop_standard": {"type": "standard", " stopwords": "_englis
...     },
...     "mappings": {"properties": {"text": {"type": "text", "analyzer": "standard", "similarity":
... }  # default config
>>> es_index_name = "hf_squad_context"  # name of the index in Elasticsearch
>>> squad.add_elasticsearch_index("context", es_client=es_client, es_config=es_config, es_index_na
```