Generating Answers: Input Text ==> Chunking ==> Embedding ==> Search Index ==> Query ==> Search ==> Question ==> Answers

Emily Weng 20016

Table of Content

- Introduction
- Steps Made
- Results
- Modifications
- Conclusion

Introduction

- The overall process of working with input texts, embeddings, vectors, and search index involves the following steps:
  - Step 1: Generate Embeddings from Input Texts
  - Step 2: Create Vectors from Embeddings
  - Step 3: Create Search Index from Vectors
  - Step 4. Generating Answers

Step 1: Generate Embeddings from Input Texts

- In this step, we will convert input texts into numerical embeddings using pre-trained models or embedding techniques
- Step up your API key and COHERE API key
-

```
# ## Chunking
# Step 1.1 Split into a list of paragraphs
################################################
texts = text.split('\n\n')

################################################
# Step 1.2 Clean up to remove empty spaces and
#          new lines
################################################
texts = np.array([t.strip(' \n') for t in
                  texts if t])
```

```
[9] texts[:3]
```

```
array(['The rapid rise of AI has led to a rapid rise in AI jobs, and many people are building exciting careers in this field. A career is a decades-long journey, and the path is not always straightforward. Over many years, I've beer
privileged to see thousands of students as well as engineers in companies large and small navigate careers in AI. In this and the next few letters, I'd like to share a few thoughts that might be useful in charting your own course.',
       'Three key steps of career growth are learning (to gain technical and other skills), working on projects (to deepen skills, build a portfolio, and create impact) and searching for a job. These steps stack on top of each
other:',
       'Initially, you focus on gaining foundational technical skills.\nAfter having gained foundational skills, you lean into project work. During this period, you'll probably keep learning.\nLater, you might occasionally carry out
a job search. Throughout this process, you'll probably continue to learn and work on meaningful projects.\nThese phases apply in a wide range of professions, but AI involves unique elements. For example:'],
      dtype='<U2736')
```

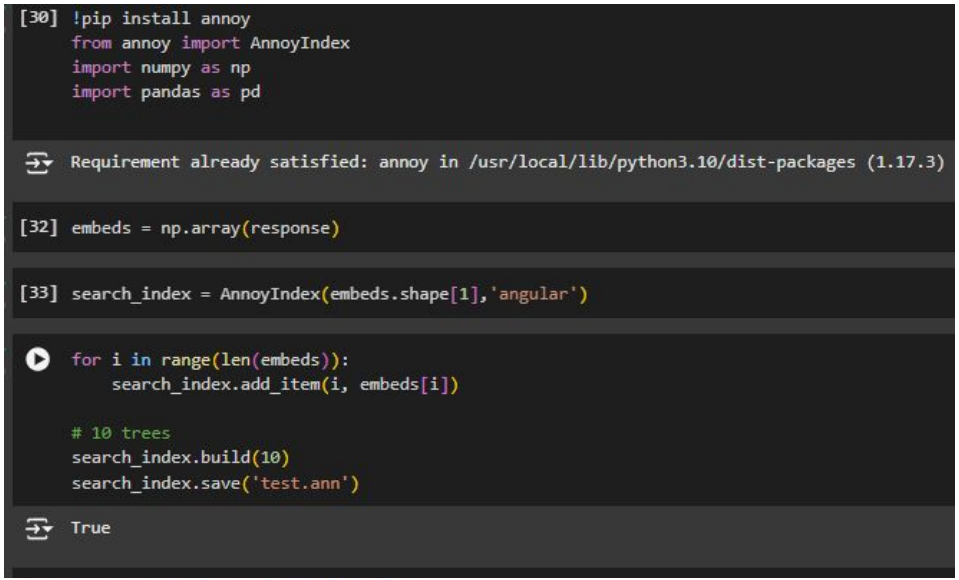
Step 1: Generate Embeddings from Input Texts

```python
import cohere

# Initialize the Cohere client with the API key
co = cohere.Client('ewRahhNSQUYVOnyrMKArj6o0AqltmhbyejN91z0M')

texts = ["example text to embed", "another example"]

# Embed texts
response = co.embed(
    texts=texts,
)

print(response.embeddings)
```

```
[[1.1992188, 1.0546875, 2.3710938, -1.0136719, 0.11791992, -2.2558594, 0.6142578, -1.0078125, 0.6323242, 1.0830078, 1.6015625, 2.9355469, 0.2121582, -1.0810547, 0.6767578, -0.6064453, 0.083496094, 1.1894531, -1.0107422, -0.4345703,
```

Step 2: Create Vectors from Embeddings

- Step 2.1 Chunking: Get the embeddings (vectors) from input texts
- Step 2.2 Use AnnoyIndex(to build a search index from the embeddings (vectors)
- Step 2.2.1 Check the dimensions of the embeddings



Step 3. Searching Articles

```
[44] def search_andrews_article(query):
        # Get the query's embedding
        query_embed = co.embed(texts=[query]).embeddings  # Ensure 'texts' is used

        # Retrieve the nearest neighbors
        similar_item_ids, _ = search_index.get_nns_by_vector(
            query_embed[0],  # Query embedding
            10,              # Number of nearest neighbors
            include_distances=True
        )

        # Fetch the results by iterating through the indices
        search_results = [texts[i] for i in similar_item_ids]  # Collect matching paragraphs

        return search_results
```

Step 4. Generating Answers

```
#Step 4.1 Generating Answers - Test Case 1
results = ask_andrews_article("Are side projects a good idea when trying to build a career in AI?")

print(results[0])
```

```
id='df4ea0cb-10a3-4246-90b5-0a0edb309111' text='Yes, side projects are a great way to build a career in AI, as they allow you to gain practical experience, demonstrate your skills, and showcase your passion for the field.' index=None
```

```
[48] # Step 4.2 Generating Answers - Test Case 2
results = ask_andrews_article("Are side projects a good idea when trying to build a career in AI?", num_generations=3)

for gen in results:
    print(gen)
    print('--')
```

```
id='119c8e6c-3c00-4157-8cb9-2ed5182e50c8' text='Yes, side projects are a great way to build a career in AI, as they allow you to gain practical experience and showcase your skills to potential employers. Andrew Ng suggests that you
--
id='526c622f-643c-43f5-9a64-f285b7602c2e' text='Yes, side projects are a great way to gain experience and build a portfolio when trying to build a career in AI.' index=None likelihood=None token_likelihoods=None finish_reason='COMPLI
--
id='45ccb21f-985f-412f-9af0-69e7532fce87' text='Yes, side projects are a great way to build a career in AI, as they allow you to gain practical experience, demonstrate your skills, and showcase your passion for the field. Andrew Ng
--
```

```
[50] # Step 4.3 Generating Answers - Test Case 3
results = ask_andrews_article("What is the most viewed televised event?", num_generations=5)

for gen in results:
    print(gen)
    print('--')
```

```
id='dac8d0eb-ce89-413a-bceb-d0b6024c735e' text='The text provided does not contain information about the most viewed televised event. The answer is not available in the excerpt.' index=None likelihood=None token_likelihoods=None fin
--
id='767083c9-c4d3-4183-b711-de94a4d2e173' text='The answer is not available in the excerpt provided. The text does not contain information about televised events or their viewership.' index=None likelihood=None token_likelihoods=None
--
id='b50ff496-f9d1-4fa1-bd99-09cb987b7f87' text='The text provided does not contain information about the most viewed televised event, so the answer is not available in the excerpt.' index=None likelihood=None token_likelihoods=None
--
id='5fcca34a-432f-42d0-8ec3-34780ad07c79' text='The text provided does not contain information about the most viewed televised event. The answer is not available in the excerpt.' index=None likelihood=None token_likelihoods=None fin
--
id='a50987c8-a1ce-43bc-b89b-1e9df0703670' text='The text provided does not contain information about the most viewed televised event, therefore the answer is not available.' index=None likelihood=None token_likelihoods=None finish_r
--
```

Modifications:

- Modified search_andrews_article function for the correct format
- Modified: response = co.embed(texts=texts).embeddings
  - From embeddings

Conclusion

- Answers were mostly accurate, test case 2 had max token issues so couldn’t get the full answer
- This could be further implemented in an actual AI bot for more experiments

Github Link

https://github.com/emilywengster/sfbu/tree/50dcca69fab68e79e771e712d76bdfaee87f86da/Generative%20AI/Fine-Tuning/Generating%20Answers

Reference Code:

https://hc.labnet.sfbu.edu/~henry/sfbu/course//deeplearning_ai/llm_semantic_search/slide/Generating_Answers.html