Final Project

Retrieval Augmented Generation

Based on NPL 89b Course Content

Erin Rebholz

CSCI S-89B Introduction to Natural Language Processing
Fall 2024
Harvard Extension School

Retrieval Augmented Generation
Introduction

Project Overview: This project leverages a RAG, or Retrieval Augmented Generation, model and incorporates a large language model to retrieve answers to questions only from specified materials.

Content Used: The recommended readings, lecture notes, and syllabus from CS89B Natural Language Processing have been used to test RAG model application in a working chatbot.

RAG Benefits: Large language models have a tendency toward hallucination (made up answers) and a lack of specific referenced content to validate information returned by the model. A RAG model helps to mitigate these downsides by drawing answers from specific content and by creating traceback mechanisms to the sources of model answers.

Testing the RAG Model: Two types of prompts are developed to test the RAG model. The first is a prompt of general knowledge with specific text references. The second is using a selection of quiz questions from the class as a 'ground truth' to test the answers of the RAG model. Answers for both tests are also compared to 'out of the box' answers returned by the GPT 3.5 turbo model of OpenAi.

Overview of RAG Prototype
Methodology

Load Model Data Into Memory → Create Embeddings, Store in Vector Store For Future Retrieval → Create RAG Model Prompt, Context, and Provide Answers → Test Results

Langchain loading tools: Used to load text files to documents (which include meta data about the document) and then to split content in to 'chunks' to be converted to character embeddings.

Open AI embeddings used to complement the OpenAI large language model (GPT 3.5 turbo)

Weaviate Vector Storage used to store the resulting embeddings. The 'embedded' option which doesn't require a server or cloud implementation

Visualization of embeddings

Langchain passes the 'context' from embeddings in the vector store, along with the 'prompt' which provides a set of general instructions for the **LLM**, in addition to the 'question' that has been provided by the user.

The 'Answer' from the RAG model is returned, derived from the specific content that has been loaded to memory.

To test the RAG model we used two prompt types:

1) General Knowledge Retrieval
2) Sample Quiz Questions

Langchain Model Orchestration

Choice of large language model

While GPT 4 provides a larger context window and more robust answers, for our demo purposes (answering multiple choice and true false questions) GPT 3.5 provides straight forward answers, ~10 times more quickly

GPT 3.5 Turbo – 2.1 Seconds

```
[9]: # Example usage
prompt = "Explain variational autencoders in simple terms."
print(query_gpt_3_5(prompt))
```

```
Variational Autoencoders (VAEs) are a type of neural network model that can lea
has been trained on. This can be used in tasks such as image generation, text ge

In simple terms, a VAE works by taking an input data point, encoding it into a l
ent space), and then decoding this representation back into a reconstructed vers
rence with VAEs compared to traditional autoencoders is that VAEs also learn a d
r the generation of new data points that are similar to the input data.

This means that VAEs can not only reconstruct input data well, but also generate
o the input data. This can be useful for tasks such as generating new images or
```

OpenAI

ChatGPT

SELECTED MODEL

Note: Answers to Questions Using Only ChatGPT (no RAG yet)

GPT 4 Turbo - - 19.6 Seconds, Longer Context Window

```
prompt = "Explain variational autencoders in simple terms."
print(query_gpt_4(prompt))
```

```
Certainly! Variational Autoencoders, often abbreviated as VAEs, are a type of neural network used primarily in the field of m
achine learning. They are used to generate complex models from simple data and are particularly popular in tasks that involve
the generation of new data that resembles the input data, like creating new images, text, or music that look or sound like th
ey could have been part of the original dataset.

To understand VAEs in a simple way, let's break it down:

### 1. **Autoencoder Basics:**
   First, it's helpful to understand what a standard autoencoder does. An autoencoder is a type of artificial neural network
used to learn efficient codings of unlabeled data. It has two main parts:
   - **Encoder:** This part of the network compresses the input into a smaller, dense representation, which is often called t
he "latent space" or "encoding". This process is like summarizing or compressing the information.
   - **Decoder:** This component takes the compressed data and reconstructs the input data back from this compressed version
as closely as possible.

   The idea is to capture the most important features of the data in the encoding, which allows the decoder to recreate the o
riginal input data accurately.

### 2. **Variational Aspect:**
   Variational Autoencoders take this concept a step further. Instead of encoding an input as a single point, they encode it
as a distribution over the latent space. This means that every input is thought of not as one set of values in the latent spa
ce, but rather as a statistical distribution (defined by parameters like mean and variance) of possible values.

### 3. **Why "Variational"?**
   The "variational" part of VAEs comes from their foundation in variational inference, a method in Bayesian statistics. VAEs
use probabilistic ways to describe the data in the latent space and learn the parameters of this probability distribution. Th
is approach allows them to not just compress data, but also generate new data that shares similar characteristics with the or
iginal data.

### 4. **Generating New Data:**
   Since the encoder of a VAE maps input data to a distribution in the latent space, we can sample points from these distribu
tions to feed into the decoder. Sampling different points allows us to generate new and varied outputs, which might look or s
ound somewhat different but maintain a relationship with the original data.

### 5. **Training:**
   VAEs are trained through a process that involves two main objectives:
   - They minimize the difference between the original input and the reconstructed output (like traditional autoencoders).
   - They ensure that the distributions in the latent space stay "regular" (typically Gaussian), which mathematically involve
s reducing what's known as the Kullback-Leibler divergence between the learned distribution and a prior.

### Summary:
In essence, Variational Autoencoders are powerful because they not only learn how to compress data, but in doing so, they lea
```

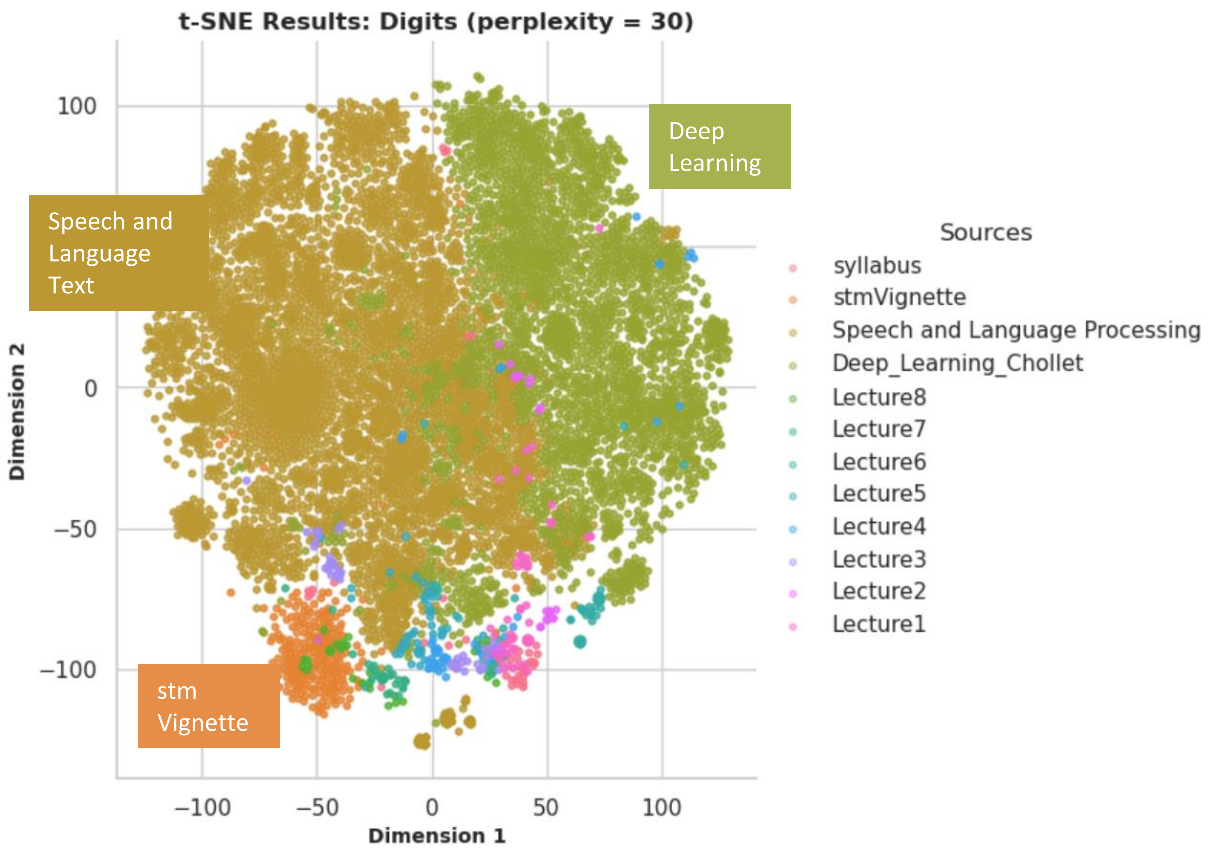
OpenAI Embeddings by Source

Documents generated by source data align nicely. Lecture topics and textbook sources demonstrate a nice overlap, indicating they are complementary

t-SNE Overview:

The **t-SNE transformation reduces dimensionality of** OpenAI **embeddings** (1500+ dimensions) down to a two-dimensional plot.

Sources are **shaded** in different colors **to see how well** they **align or diverge.**



Insights:

Seems to be a **vertical boundary between two texts**(Deep Learning and Speech and Language Processing), with a **strong overlap at the center, where** the **topics** may **overlap between** the **two texts**

Lecture Notes are **intermixed with** documents from the **two texts**, which may **indicate** that the **texts and lecture topics align** well.

Lecture8 and stmVignette closely aligned (orange and green in bottom left) and are **somewhat distinct from the other topics as might be expected**

A basic retrieval augmented model, orchestrated with LangChain, is used for the project

How it works:

A basic retrieval augmented generation model takes a user ‘question’ and passes that question, along with information ‘retrieved’ from a vector store, through a structure ‘prompt’ and then leverages a large language model to return an answer to the user question.

RAG Model Overview

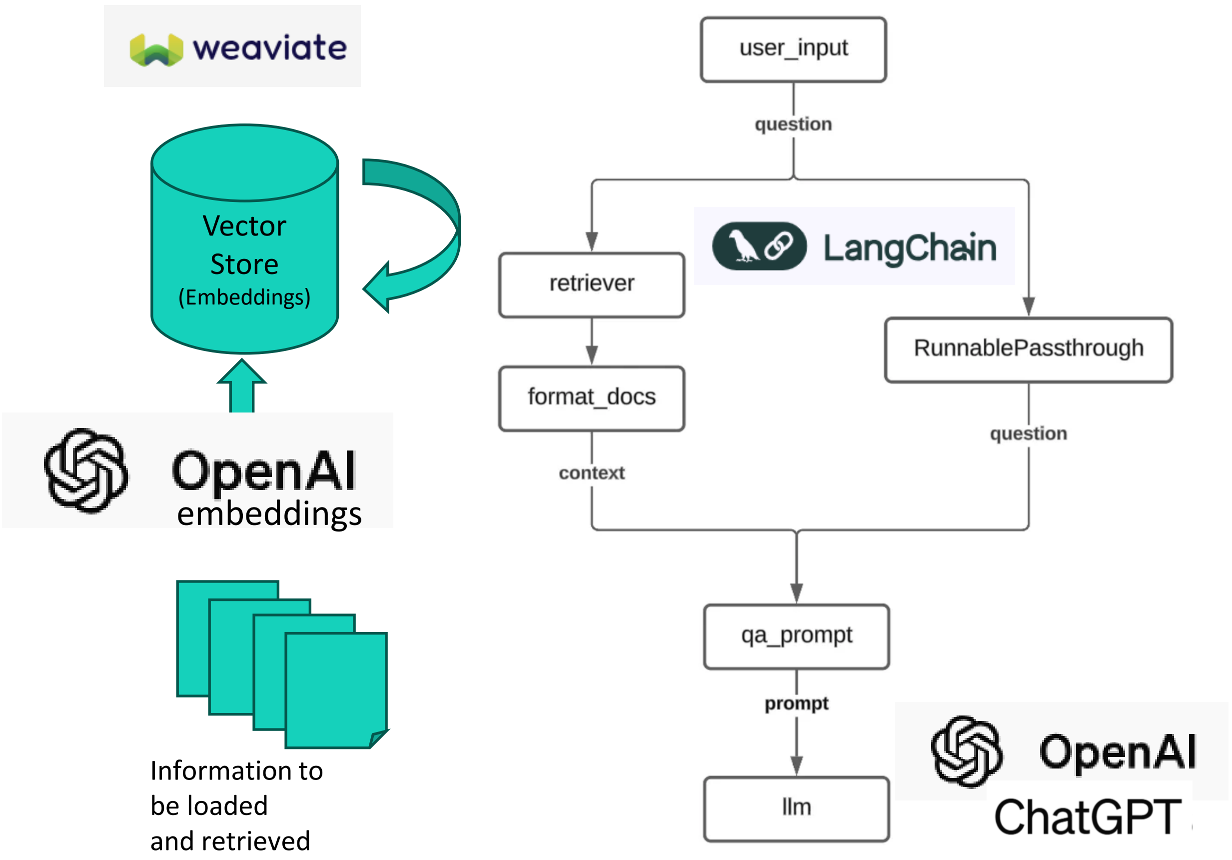


Image Credit: https://towardsdatascience.com/building-a-rag-chain-using-langchain-expression-language-lcel-3688260cad05

Testing the RAG Model: Prompts

Generic Content Questions:

```python
#Create a prompt template to use to answer a question with the given context

from langchain.prompts import ChatPromptTemplate

template = """You are an assistant for question-answering tasks.
Use the following pieces of retrieved context to answer the question.
If you don't know the answer, just say that you don't know.
Provide specific text references.
Use three sentences maximum and keep the answer concise.
Question: {question}
Context: {context}
Answer:
"""
prompt = ChatPromptTemplate.from_template(template)

print(prompt)
```

Answers to general NLP questions can be answered by GPT as a stand alone model. However, the RAG model provides document sourcing and course specific answers.

Query	GPT_3_5_Answers	RAG_Model_Answers
What is semantic role labeling?	Semantic role labeling is a natural language processing task that involves assigning a semantic role to each word or phrase in a sentence in order to identify the relationships between different elements in the sentence. These semantic roles typically represent the different functions that words or phrases serve in relation to the main verb in a sentence, such as agent, patient, instrument, and more. Semantic role labeling is used in various NLP applications, such as information extraction, question answering, and machine translation.	Rag model answer: Semantic role labeling is the task of assigning semantic role labels to the constituents of a sentence. It is generally treated as a supervised machine learning task, with models trained on PropBank or FrameNet. Algorithms start by parsing a sentence and then automatically tag each parse tree node with a semantic role (source: data/ed3bookaug20_2024.pdf, page 485).
What is the attention mechanism?	The attention mechanism is a component of neural network architectures that allows the model to focus on certain parts of the input data based on their relevance to the task at hand. This mechanism helps the model to selectively attend to different parts of the input, giving more weight to those parts that are deemed important, while ignoring irrelevant or redundant information. Attention mechanisms have been widely used in natural language processing tasks, such as machine translation and text summarization, as well as in computer vision tasks, such as image captioning and object detection.	The attention mechanism allows the decoder to access information from all hidden states of the encoder, not just the last hidden state (source: data/ed3bookaug20_2024.pdf, page 186). It has become one of the most influential ideas in deep learning, particularly effective for sequence data (source: data/deep_learning_chollet_2021.pdf, page 360). The attention mechanism enhances efficiency by utilizing grouped-query attention mechanisms in models like Llama-3-8B2 and InternLM2-20B (source: data/ed3bookaug20_2024.pdf, page 255).
What is the professors name for natural language processing 89b?	The professor's name for Natural Language Processing 89b is not provided in the query.	The professor's name for natural language processing 89b is Dmitry Kurochkin. (Source: data/IntroNLP_Lecture8.pd

Rag Model Can Provide Document Specific Sourcing of Answers

Rag Model Can Answer Course Specific Content Questions

Testing the RAG Model: Prompts

Generic Content Questions:

```
#Create a prompt template to use to answer a question with the given context

from langchain.prompts import ChatPromptTemplate

template = """You are an assistant for question-answering tasks.
Use the following pieces of retrieved context to answer the question.
If you don't know the answer, just say that you don't know.
Provide specific text references.
Use three sentences maximum and keep the answer concise.
Question: {question}
Context: {context}
Answer:
"""
prompt = ChatPromptTemplate.from_template(template)

print(prompt)
```

Specific Quiz Questions (Six, Random):

```
#Create a prompt template to use to answer a question with the given context

from langchain.prompts import ChatPromptTemplate

template = """You are answering a quiz with multiple choice and true / false
questions. If you don't know the answer, just say that you don't know.
Answer with the specific letter or true/false answer.
Provide a one sentence rationale for your answer.
Question: {question}
Context: {context}
Answer:
"""
prompt = ChatPromptTemplate.from_template(template)

print(prompt)
```

Both GPT and the RAG models can answer some of our basic quiz questions accurately:

Quiz_ Question	Question Text	GPT_Answer	RAG_Model
q1_5	If learning rate of the Gradient Descent is too large, then the training algorithm will likely:(A) converge to a local minimum and the convergence will be fast(B) converge to a local minimum, but the convergence will be slow(C) diverge(D) none of these	**(C) diverge** If the learning rate is too large, the Gradient Descent algorithm may overshoot the minimum and oscillate around it or move further away from it, resulting in the algorithm never converging. This is known as divergence.	**(C) diverge** Rationale: If the learning rate is too large, the algorithm may overshoot the minimum of the loss function, causing it to diverge rather than converge.
q6_1	Disadvantages of one-hot encoding of words is that the representations are:(A) sparse(B) high dimensional(C) both (A) and (B)(D) none of (A), (B)	**(C) both (A) and (B)**	**(C) both (A) and (B)** Rationale: The text explicitly states that one-hot encoding of words leads to sparse representations (mostly zeros) and very high-dimensional vectors, making both statements true.
q6_2	An autoencoder is called undercomplete if(A) dimensionality of internal representations is higher(B) dimensionality of internal representations is lower(C) some of the layers are frozen(D) some of the connections are broken(E) none of (A), (B), (C), (D)	**(B)** dimensionality of internal representations is lower An undercomplete autoencoder has a lower dimensionality of internal representations compared to the input data. This forces the autoencoder to learn a compressed representation of the input data.	**(B)** dimensionality of internal representations is lower Rationale: An autoencoder is called undercomplete when the dimensionality of the internal representations is lower, forcing it to learn the most important features.
q6_3	True or False: Stacked autoencoder refers to an autoencoder which represents a deep neural network.	**True.**	**True** Rationale: Stacked autoencoders refer to autoencoders that are stacked on top of each other to create a deep neural network.
q8_2	Which function is used to preprocess text data for analysis in the 'stm' R package?(A) \`selectModel()\`(B) \`prepareModel()\`(C) \`textProcessor()\`(D) \`stm()\`	**(C)** \`textProcessor()\`	**(C)** \`textProcessor()\` Rationale: The \`textProcessor()\` function is specifically mentioned as a pre-processing function for cleaning and preparing text data in the 'stm' package.
q10_3	What role does the SpaCy library play in Named Entity Recognition (NER) tasks?Please select all that apply:(A) It provides datasets for training machine learning models.(B) It visualizes NLP pipelines in a graphical user interface.(C) It offers built-in, trained NER models for various languages.(D) It translates text between different languages.	**(C)** It offers built-in, trained NER models for various languages.	**(C)** It offers built-in, trained NER models for various languages. Rationale: The SpaCy library provides pre-trained NER models for different languages, making it easier to perform Named Entity Recognition tasks.

Project Findings:

Guardrails for Ethical Use

- The contrast between these two RAG model tests provides a useful example of how prompt structure can provide guardrails for ethical use:
- While I had originally planned to use the course quiz questions as a ground truth to more robustly test our RAG model, even the basic Open AI model seemed very well equipped to answer the quiz questions correctly. This exercise highlighted for me ethical limitations of using these llm tools and the need for academic integrity policies.
- That said, the general knowledge query was extremely helpful in providing the context behind a question answer and the RAG model construct provides useful answers with references to specific areas of lecture and recommended readings that can be used to explore materials further. This approach provides support to the learning process versus a shortcut.

Operational vs. Analytic Complexity

- This project also shows how using large language models represent a transition from detailed analytic approaches architected from the grounds up to more development operations concerns. In general, the hardest part was getting all the packages to work together. In the end, I have new appreciation for LLM Ops:
  - Lots of codes packages, require updating of packages frequently to ensure dependencies are working.
  - Warnings are helpful to decode required install/packages that are more up to date and work together with existing packages

Video Presentation

https://youtu.be/LPXrw7oDv1c