Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

LLMs hallucinate. We can assist them with the help of a retriever (DPR).

> **Do RAGs use vector database?**
>
> Not always, in this paper they don't.

It seems that this is end-to-end trained, which in practice is quite expensive and people don't do this as much.

Architecture

Composed of 2 components:

- (i) a *retriever* $p_\eta(z|x)$ with parameters η that returns (top-K truncated) distributions over text passages given a query x
- (ii) a *generator* $p_\theta(y_i|x, z, y_{1:i-1})$ that generates a current token based on a context of the previous $i-1$ tokens $y_{1:i-1}$, the original input x and a retrieved passage z.

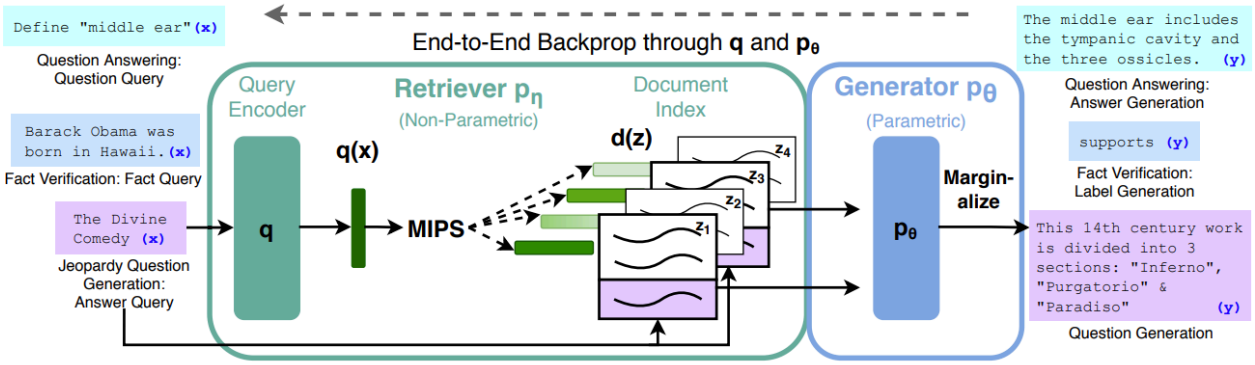

Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat z as a latent variable and marginalize over seq2seq predictions given different documents.

How does it get trained end to end?

- "We use a pre-trained bi-encoder from DPR to initialize our retriever and to build the document index."

How does a generator read from the retrieved document z?

- "To combine the input x with the retrieved content z when generating from BART, we simply concatenate them"

> **After, when the documents changes, what happens? or if we want to increase the number of documents?**
>
> I suppose the document just goes through the BERT encoder again, and everything just works out of the box. The embedding value of the document will change, in which case the dot product value between query and document will also change, so the document's ranking may change.

Terminology

- parametric memory = knowledge already stored in the model

- non-parametric memory = retrieval-based memory (knowledge that model doesn't have directory access to)

Training

They keep the document encoder (and index) fixed, only fine-tuning the query encoder BERTq and the BART generator."

Questions

What are the authors trying to do? Articulate their objectives.
The author introduces explore a general-purpose fine-tuning recipe for retrieval-augmented generation (RAG) model, which augments a pre-trained generative model (BART) with an information retriever, to drastically improves the performance of the model and reduce hallucination.

How was it done prior to their work, and what were the limits of current practice?
Without this, LLMs tend to hallucinate. When it was asked about information that it did not know about, it would invent something since it was unable to search for knowledge. While there was some exploration done in previous papers about combining retrieval systems with generative models (ex: ORQA), these papers only explored open-domain extractive question answering tasks, and did not explore generative downstream tasks.

What is new in their approach, and why do they think it will be successful?
It is composed of two parts: a retriever (implemented with a DPR) and a generator (implemented with BART). Because it is trained end to end, the retriever learns to fetch the most relevant documents, and the generator is conditioned on a retrieved document provided by the retriever, and learns to generate the correct output token by minimizing the cross-entropy loss.

What are the mid-term and final "exams" to check for success? (i.e., How is the method evaluated?)
Similar to the DPR paper, they evaluate on a series of question-answer datasets such as NQ, TQA, WQ and CT. They show that they consistently outperform the base model, when the model is given the ability to access the latest information via the RAG system. This shows that their method directly addresses a major limitation of parametric-only models.