CUAI NLP Study

Research Paper Review

HTET ARKAR

Undergraduate

School of Computer Science and Engineering

Chung-Ang University

Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis[†‡], Ethan Perez[⋆],

Aleksandra Piktus[†], Fabio Petroni[†], Vladimir Karpukhin[†], Naman Goyal[†], Heinrich Küttler[†],

Mike Lewis[†], Wen-tau Yih[†], Tim Rocktäschel[†‡], Sebastian Riedel[†‡], Douwe Kiela[†]

[†]Facebook AI Research; [‡]University College London; [⋆]New York University;
plewis@fb.com

Outline

- Introduction
- Methodology
- Conclusion

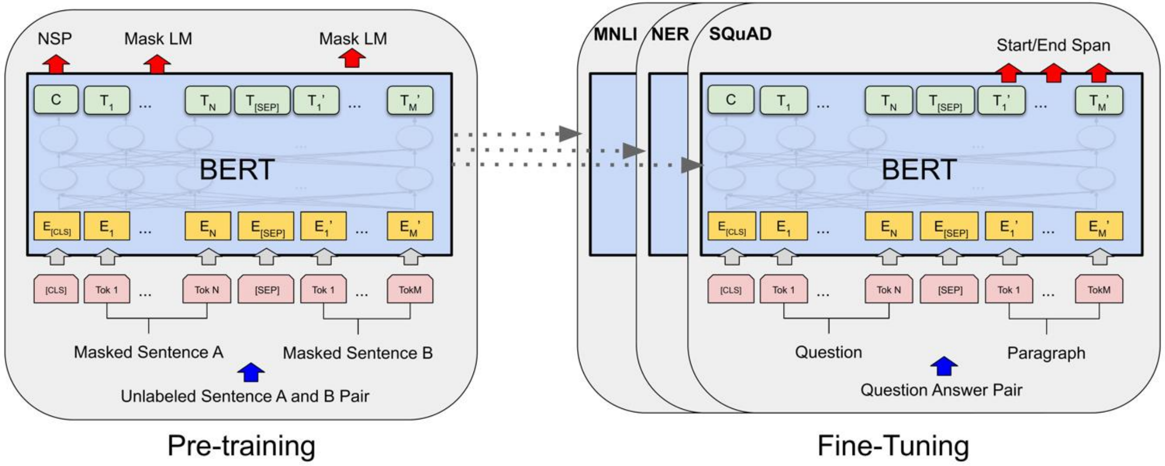
Outline

- **Introduction**
  - Problem Definition
  - Proposed Method
  - Results Highlight
- **Methodology**
- **Conclusion**

Problem Definition

❖ Pre-trained Neutral Language Models

- ❑ Learn a substantial amount of in-depth knowledge from data
- ❑ Without any access to an external memory ➔ ***parameterized*** implicit knowledge base
- ❑ Achieve strong results on a variety of downstream NLP tasks

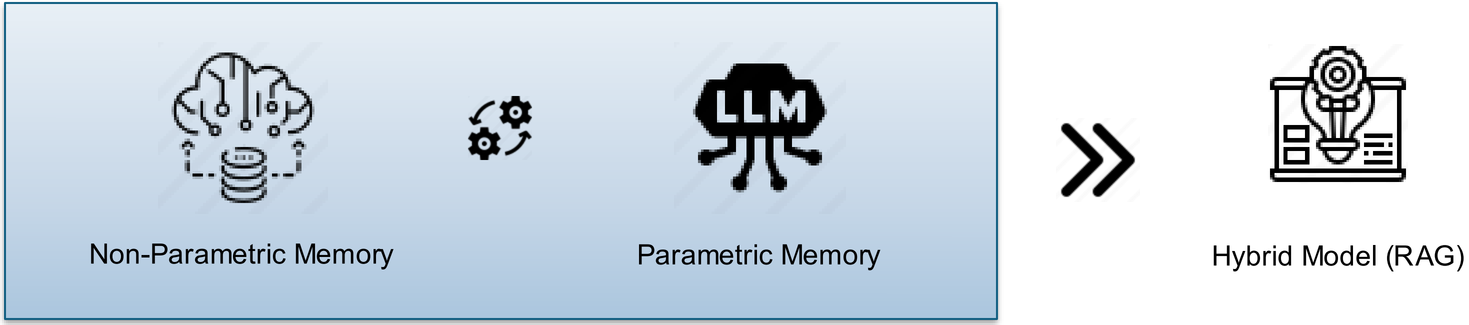


Problem Definition

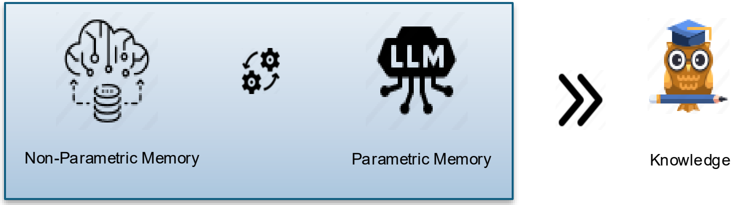
Limitations

- Cannot easily expand or revise their memory
- Cannot straightforwardly provide insight into their predictions
- May produce “hallucinations” (generating incorrect information)

Solution

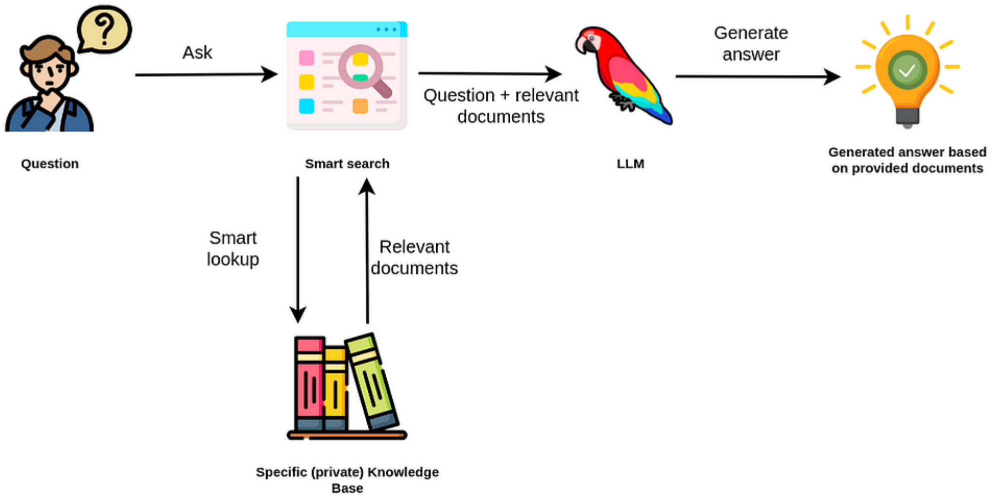


Problem Definition



❖ Hybrid Model

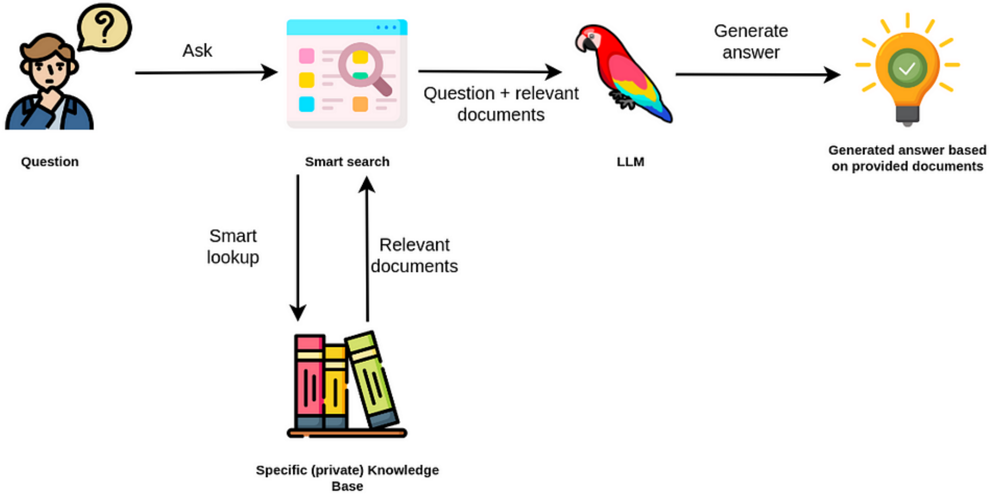
- ❑ Knowledge can be directly revised and expanded
- ❑ Accessed knowledge can be inspected and interpreted



Proposed Method

❖ Retrieval-augmented generation (RAG)

- ❑ Pre-trained, parametric-memory generation models with a non-parametric memory
- ❑ A general-purpose fine-tuning approach
- ❑ Can be fine-tuned on any seq2seq task



Proposed Method

❖ RAG: Overview

- ❑ Parametric memory ➔ A pre-trained seq2seq transformer
- ❑ Non-parametric memory ➔ A dense vector index of Wikipedia (DPR)
- ❑ Combination ➔ A probabilistic model trained end-to-end

Proposed Method

❖ RAG: Overview

- ❑ Dense Passage Retriever
  - ➢ Provide latent documents conditioned on the input
- ❑ seq2seq model (BART)
  - ➢ Condition on these latent documents together with the input to generate the output
- ❑ Marginalize the latent documents with a top-K approximation
  - ➢ Either on a **per-output basis** (assuming the same document is responsible for all tokens)
  - ➢ Or a **per-token basis** (where different documents are responsible for different tokens)

Results Highlight

❖ Generation for Knowledge-intensive tasks

- ❑ Tasks that humans could not reasonably be expected to perform without access to an external knowledge source
- ❑ RAG models achieve SOTA performance (General NLP Tasks)
  - ➢ Open Natural Questions, WebQuestions and CuratedTrec
- ❑ Strongly outperform on TriviaQA (Specialised pre-training objectives)
- ❑ Generate responses that **are more factual, specific, and diverse** than a BART baseline
  - ➢ MS-MARCO and Jeopardy question generation (Knowledge-intensive tasks)
- ❑ One of Top SOTA models ➔ FEVER fact verification

Non-parametric memory ~~can be~~ replaced to update the models' knowledge as the world changes.

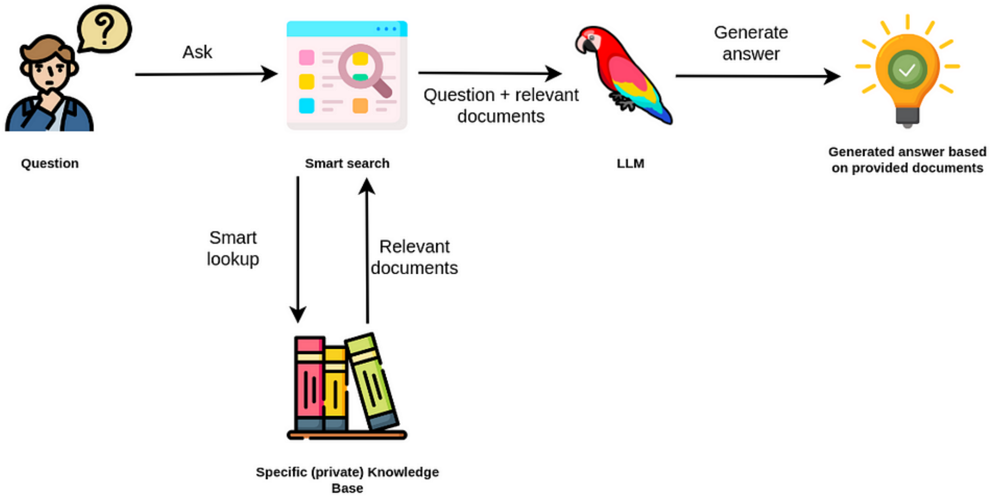
Outline

- Introduction
- **Methodology**
  - RAG-Sequence Model
  - RAG-Token Model
  - Retriever: DPR
  - Generator: BART
  - Training
  - Decoding
- Conclusion

Methodology

❖ RAG Pipeline

- ❑ Use the input sequence x to retrieve text documents z
- ❑ Use them as additional context when generating the target sequence y



RAG Pipeline

❖ Two Components

- ❑ (i) a retriever $p_\eta(z \mid x)$ with parameters η
  - ➢ Return (top-K truncated) distributions over text passages given a query x
- ❑ (ii) a generator $p_\theta(y_i \mid x, z, y_{1:i-1})$ parametrized by θ
  - ➢ Generate 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

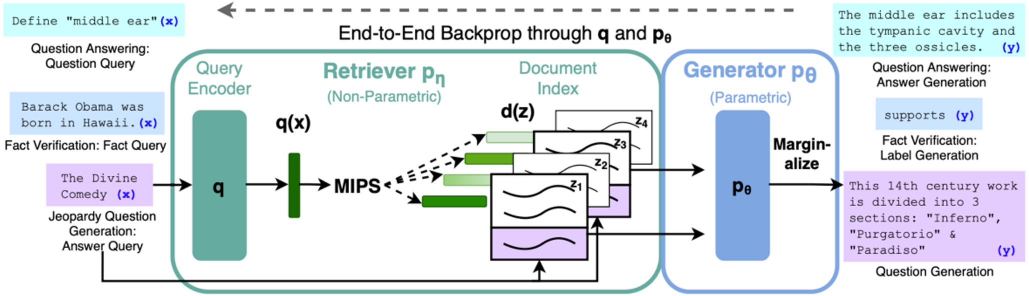


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat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

RAG Pipeline

❖ Training the retriever and generator end-to-end

- ❑ Treat the retrieved document as a latent variable
- ❑ Two models: marginalize over the latent documents to produce a distribution over generated text
  - ➢ **RAG-Sequence** - use the same document to predict each target token
  - ➢ **RAG-Token** - predict each target token based on a different document

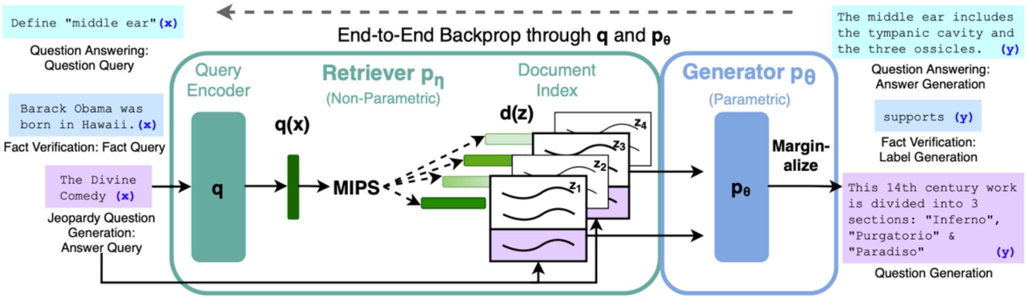


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat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

RAG-Sequence Model

$$p_{\text{RAG-Sequence}}(y|x) \approx \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) p_\theta(y|x,z) = \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) \prod_i^N p_\theta(y_i|x,z,y_{1:i-1})$$

❖ Using the same retrieved document to generate the complete sequence

- ❑ Treat the retrieved document as a single latent variable
- ❑ Marginalized to get the seq2seq probability ***p(y*** | ***x***) via a top-K approximation
- ❑ The top K documents are retrieved using the retriever
- ❑ The generator produces the output sequence probability for each document

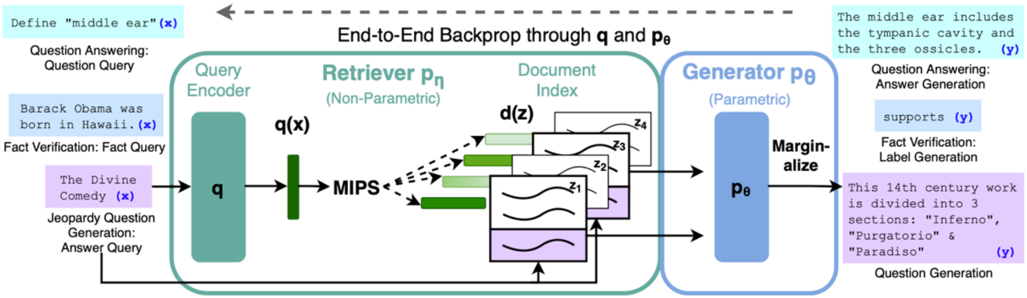


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat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

RAG-Token Model

$$p_{\text{RAG-Token}}(y|x) \approx \prod_{i}^{N} \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) p_\theta(y_i|x, z, y_{1:i-1})$$

❖ Drawing a different latent document for each target token

- ❑ Allow the generator to choose content from several documents when producing an answer
- ❑ The top K documents are retrieved using the retriever
- ❑ The generator produces a distribution for the next output token for each document,
- ❑ Then marginalizing and repeating the process

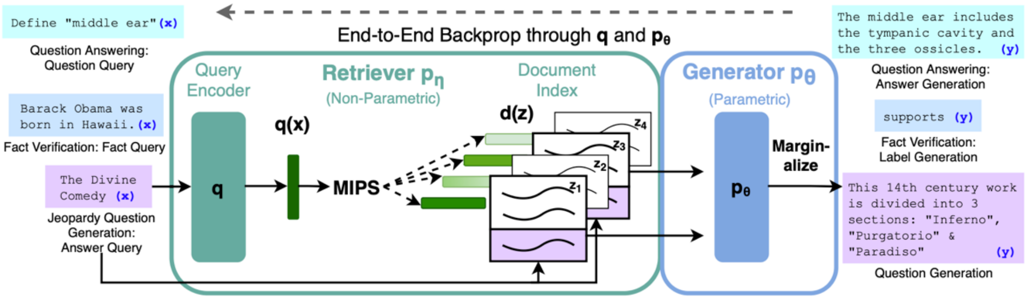


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat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

RAG-Components

- **Retriever: DPR** (A bi-encoder architecture)
  - A pre-trained bi-encoder from DPR to initialize the retriever and to build the document index

Document Encoder   Query Encoder

$$p_\eta(z|x) \propto \exp\left(\mathbf{d}(z)^\top \mathbf{q}(x)\right) \qquad \mathbf{d}(z) = \mathrm{BERT}_d(z), \;\; \mathbf{q}(x) = \mathrm{BERT}_q(x)$$

  - Top-k ($p_\eta(\cdot|x)$) - List of k documents z with highest prior probability $p_\eta(z|x)$

- **Generator: BART** (Encoder-Decoder architecture)
  - Concatenate the input x with the retrieved content z ➔ Generating from BART
  - BART generator parameters θ ➔ the parametric memory

RAG-Components

❖ Training

- Both the retriever (query encoder) and generator are trained jointly,
  - Optimizing the negative marginal log-likelihood of the target sequence
- The document encoder and document index remain fixed
  - To avoid expensive periodic re-indexing
- No explicit supervision is provided regarding which document should be retrieved

❖ Decoding

- RAG-Token:
  - Uses standard beam decoding
  - where each token may be conditioned on a different retrieved document
- RAG-Sequence:
  - Requires running beam search separately for each document
  - Then marginalizing over the beams to approximate the final output probability

Conclusion

Large pre-trained language models

- Store factual knowledge in their parameters,
- Achieve SPTA results when fine-tuned on downstream NLP tasks
- Their ability to access and precisely manipulate knowledge is still limited
- Especially on knowledge-intensive tasks

Retrieval-Augmented Generation

- A general-purpose fine-tuning recipe for retrieval-augmented generation
- Models which combine pre-trained parametric and non-parametric memory for language generation

Thank You!

HTET ARKAR
hak3601@gmail.com