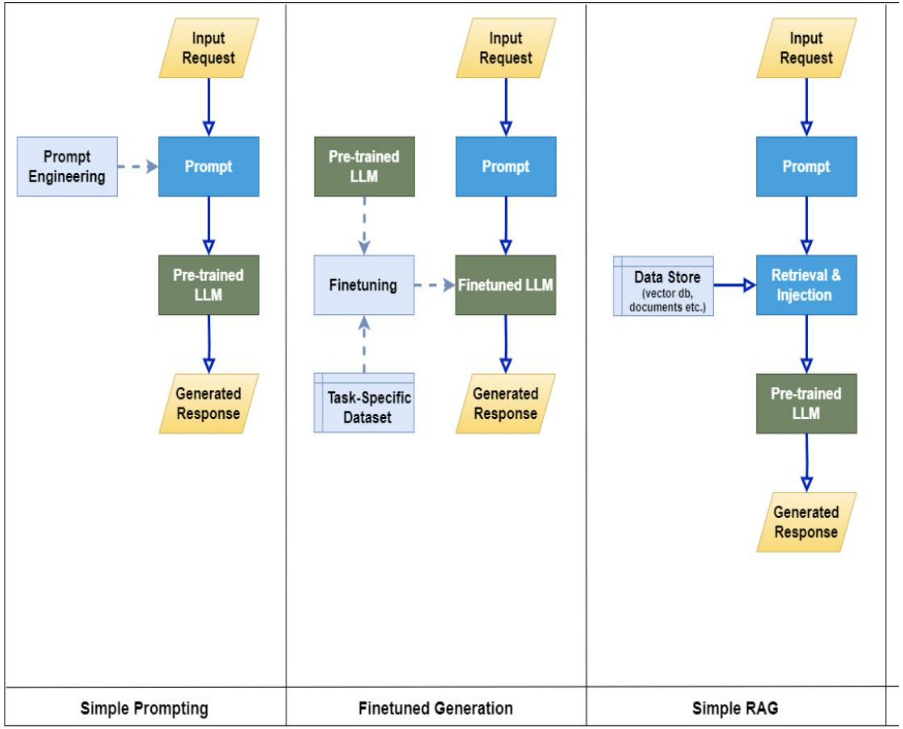
Improving LLM Performance

BIODS 271 / CS 277

Tanveer Syeda-Mahmood

Improving Performance of Foundational Models

- Prompt engineering
  - Ensures prompts optimized to help the model understand the intent and context of the request.
- RAG
  - Anchors the generation on factual and time relevant information, along with information source transparency.
- Fine-tuning
  - SFT
    - Training on high-quality, curated datasets to improve task-specific performance.
  - Reinforcement learning
    - Using reward models to improve performance
  - Synthetic data generation
    - High quality synthetic data generation
  - InstructLab:
    - Mix curated examples with synthetic data generation for fine-tuning
- Most of these are well-developed for LLMs, still emerging for VLM, LVLMs



Elements of a prompt

A prompt contains any of the following elements:

Instruction - a specific task or instruction you want the model to perform

Context - external information or additional context that can steer the model to better responses

Input Data - the input or question that we are interested to find a response for

- **Output Indicator** - the type or format of the output.

Prompt engineering

- Zero-shot prompting
- Prompt chaining
- Chain of thought
- Object-guided prompting
- Meta prompting

Prompts do matter:

- E.g. Different results with slightly different prompt
  - A photo of a plane
  - A photo of a plane parked at the gate
  - A photo of a plane parked at the gate or flying in the sky

The art of prompting

Be Specific: Prompt should be specific
Indicate Intent: Make sure your intent is clear
Show Examples: use them for formatting and inferring commonalities
Given Clear Instructions: The instructions shouldn’t be ambiguous
Indicate Desired Output: Indicate the format of the output

Efficient Prompting Methods for Large Language Models: A Survey

Few-shot prompting

- Is a type of "in-context learning" or "learning by example."
  - Pattern recognition:
    - identifying patterns in how inputs are transformed into outputs.
  - Task inference:
    - The nature of the task being asked to perform
  - Generalization:
    - Generalize from given examples to new inputs
  - Application:
    - Apply the learned pattern to new input
- What are some pitfalls?
  - What if there isn't enough variety?
  - What if there is too much variety?

Few shot example:

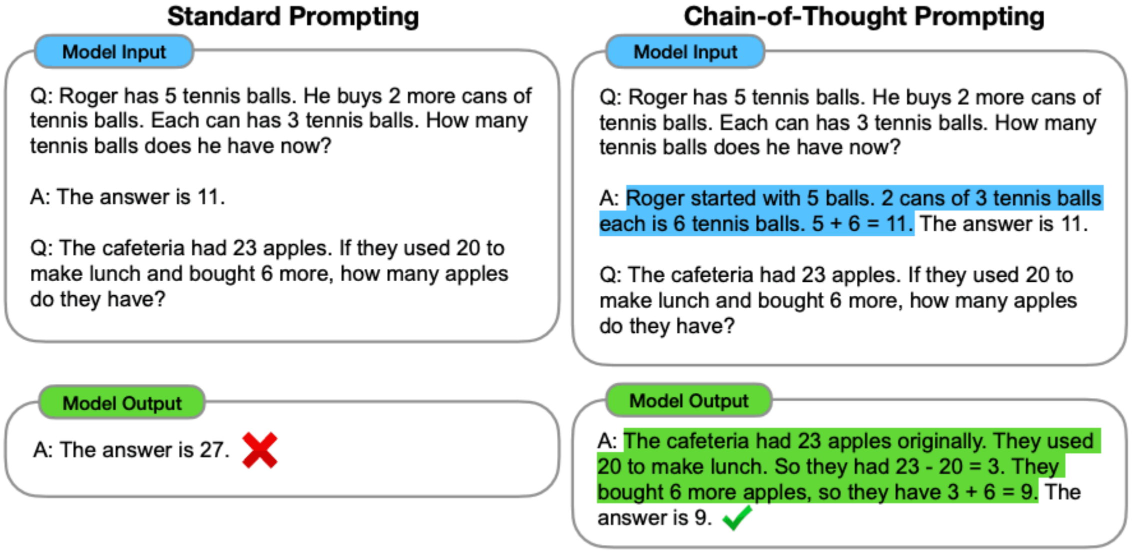
- Classify the sentiment of the third movie review. Use the information from the first two examples:
- Review: "This movie was a waste of time.
- "Sentiment: NegativeReview:
- "I couldn't stop laughing throughout the film!
- "Sentiment: PositiveReview:
- "The special effects were amazing, but the plot was confusing
- "Sentiment:"

Provide a possible diagnosis and explain your reasoning:
Symptoms: Fever, cough, fatigue
Diagnosis: Common cold
Explanation: The combination of fever, cough, and fatigue is typical of a common cold. No severe symptoms are present, suggesting a mild viral infection.
Symptoms: Chest pain, shortness of breath, dizziness
Diagnosis: Possible heart attack
Explanation: The combination of chest pain, shortness of breath, and dizziness are warning signs of a possible heart attack. Immediate medical attention is required.
Symptoms: Headache, sensitivity to light, nausea
Diagnosis:
Explanation:

https://www.datacamp.com/tutorial/few-shot-prompting

Chain-of-thought prompting

- Tackle address complex problems by breaking them into simpler, intermediate steps and encouraging the model to "think" before answering



Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

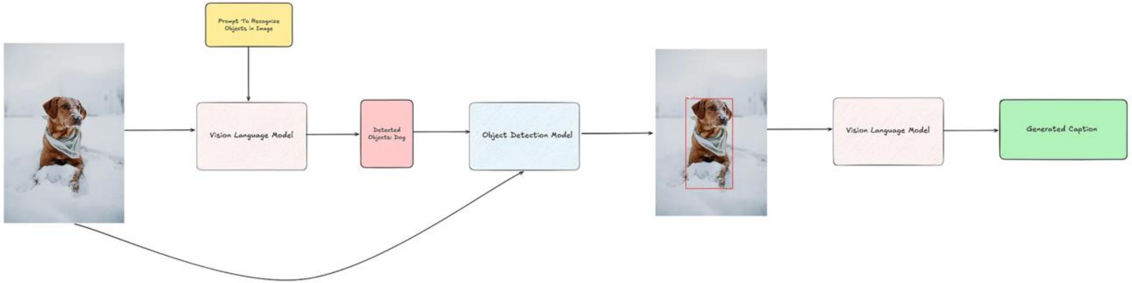
A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

Object-guided prompting

- Combined open vocabulary tagging with object detection to generate a more focused caption

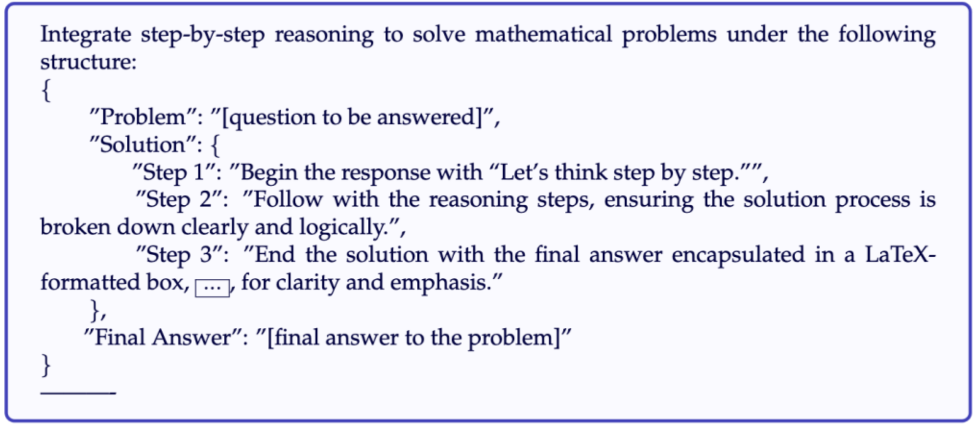
```
system_prompt="""You are a helpful assistant that can analyze images
and provide captions. You are provided with images that also contain
bounding box annotations of the important objects in them, along with
their labels.
Analyze the overall image and the provided bounding box information
and provide an appropriate caption for the image.""",
```



https://medium.com/data-science/prompting-with-vision-language-models-bdabe00452b7

Meta-prompting

- Focuses on the structure rather than infer from the answer unlike COT



Integrate step-by-step reasoning to solve mathematical problems under the following structure:
{
”Problem”: ”[question to be answered]”,
”Solution”: {
”Step 1”: ”Begin the response with “Let’s think step by step.””,
”Step 2”: ”Follow with the reasoning steps, ensuring the solution process is broken down clearly and logically.”,
”Step 3”: ”End the solution with the final answer encapsulated in a LaTeX-formatted box, $\boxed{\ldots}$, for clarity and emphasis.”
},
”Final Answer”: ”[final answer to the problem]”
}

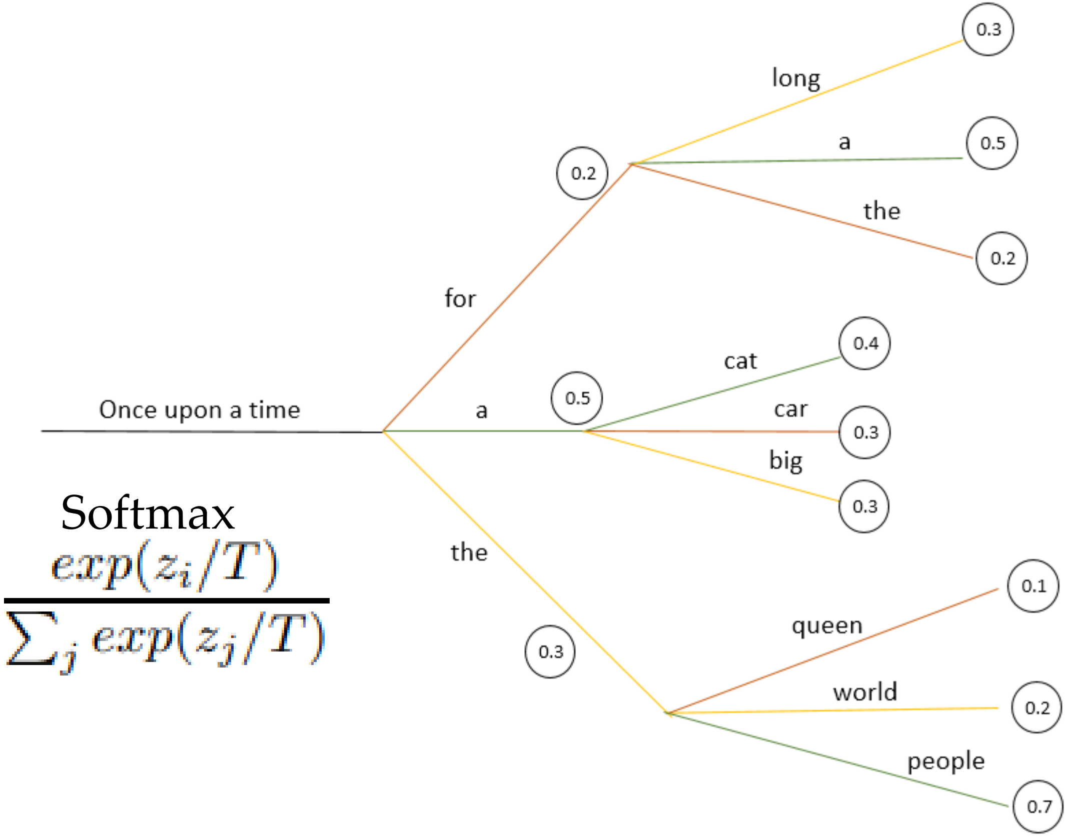
Meta Prompting for AI Systems

Understanding the LLM text generation process

- Transformer decoder
- Next token selector
  - Beam search
    - Deterministic
    - Stochastic
      - Random
      - Temperature-based
      - Top-k
      - Top-p

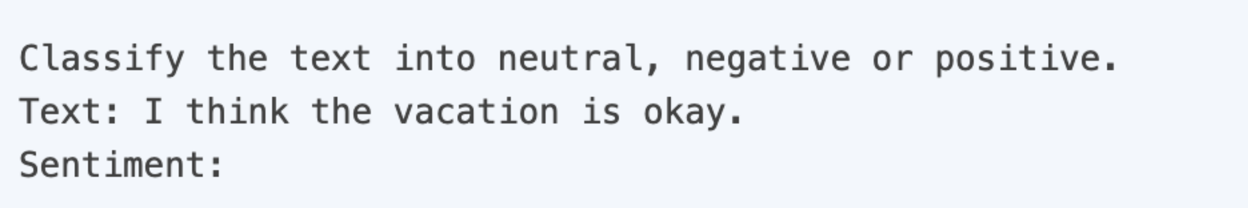
Softmax

$$q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$$

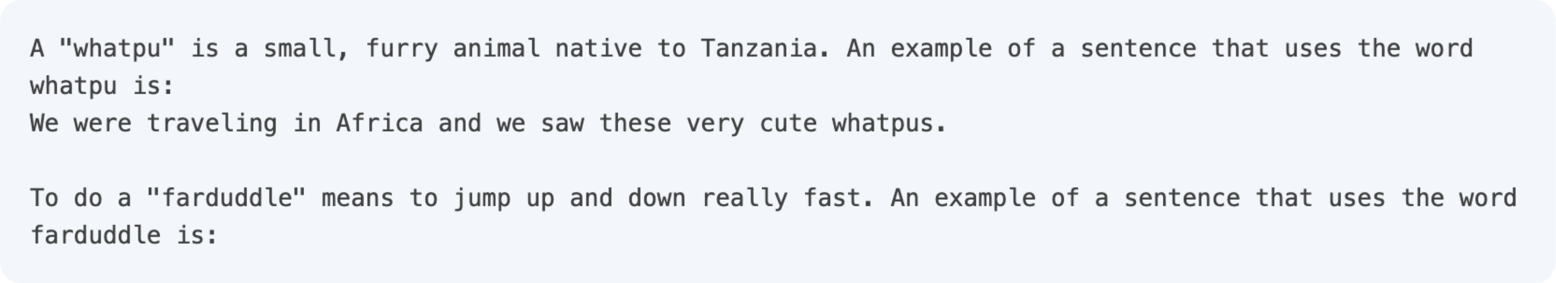


A Thorough Examination of Decoding Methods in the Era of LLMs

Exercise: Guess prompting type

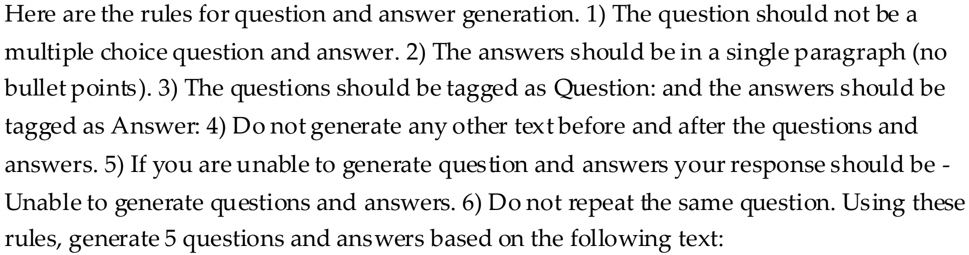


```
Classify the text into neutral, negative or positive.
Text: I think the vacation is okay.
Sentiment:
```



```
A "whatpu" is a small, furry animal native to Tanzania. An example of a sentence that uses the word
whatpu is:
We were traveling in Africa and we saw these very cute whatpus.

To do a "farduddle" means to jump up and down really fast. An example of a sentence that uses the word
farduddle is:
```

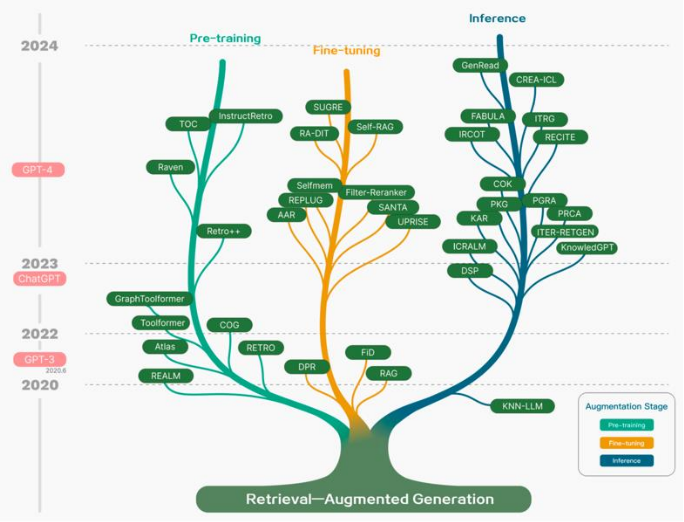


Here are the rules for question and answer generation. 1) The question should not be a multiple choice question and answer. 2) The answers should be in a single paragraph (no bullet points). 3) The questions should be tagged as Question: and the answers should be tagged as Answer: 4) Do not generate any other text before and after the questions and answers. 5) If you are unable to generate question and answers your response should be - Unable to generate questions and answers. 6) Do not repeat the same question. Using these rules, generate 5 questions and answers based on the following text:

https://arxiv.org/abs/2005.14165

Retrieval augmented generation

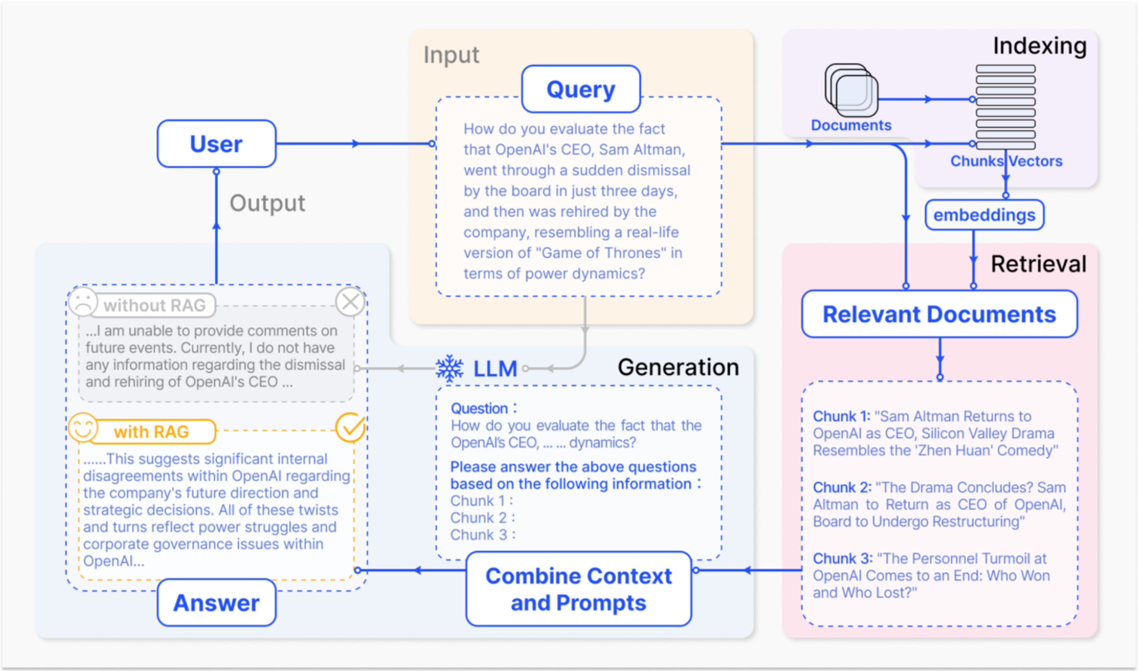
- A way to improve the performance of LLM
  - Could be used during pre-training or fine-tuning but mostly used during inference
- Practical approach to improve the model response with latest onsite knowledge
- Minimizes hallucinations with provided knowledge
- Time-relevant responses
- Increases transparency and gives traceable reasoning
- Cost-effective way to deploy models
- Pivotal technology for chatbots and practical applications.
- Widely implemented in commercial applications and popularized vector databases



Retrieval-Augmented Generation for Large Language Models: A Survey, Jan 2024

RAG – Key idea

- Given a query to an LLM
  - Search/retrieve relevant documents
  - Construct prompts with the relevant documents
  - LLM to generate revised response
- How to determine relevancy?
- How does the new information integrate into LLM?
- How to evaluate the improvement?



How do you evaluate the fact that OpenAI's CEO, Sam Altman, went through a sudden dismissal by the board in just three days, and then was rehired by the company, resembling a real-life version of "Game of Thrones" in terms of power dynamics?

...I am unable to provide comments on future events. Currently, I do not have any information regarding the dismissal and rehiring of OpenAI's CEO ...

......This suggests significant internal disagreements within OpenAI regarding the company's future direction and strategic decisions. All of these twists and turns reflect power struggles and corporate governance issues within OpenAI...

Question :
How do you evaluate the fact that the OpenAI's CEO, ... dynamics?

Please answer the above questions based on the following information :
Chunk 1 :
Chunk 2 :
Chunk 3 :

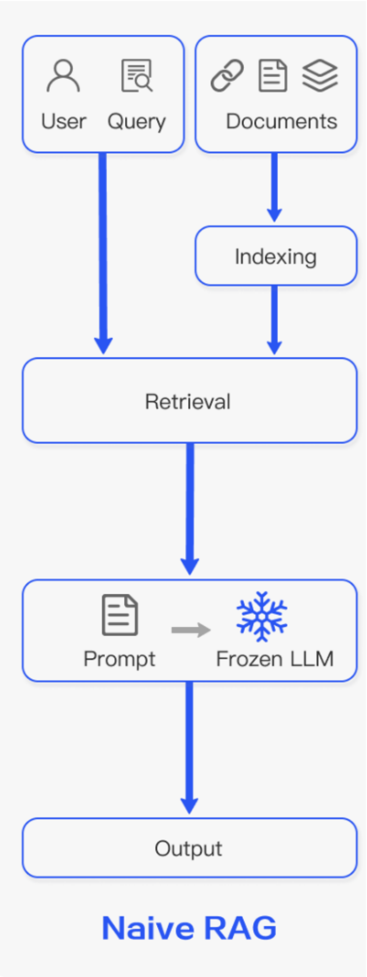
Chunk 1: "Sam Altman Returns to OpenAI as CEO, Silicon Valley Drama Resembles the 'Zhen Huan' Comedy"

Chunk 2: "The Drama Concludes? Sam Altman to Return as CEO of OpenAI, Board to Undergo Restructuring"

Chunk 3: "The Personnel Turmoil at OpenAI Comes to an End: Who Won and Who Lost?"

Naïve RAG

- Indexing
  - Text conversion
  - Chunking documents
  - Encoding documents
  - Index generation (k-NN), e.g. Faiss index
- Retrieval
  - Cosine similarity retrieval by encoding query
- Generation
  - Responses to the information contained within the provided documents
  - In cases of ongoing dialogues, any existing conversational history can be integrated into the prompt,
  - enabling the model to engage in multi-turn dialog



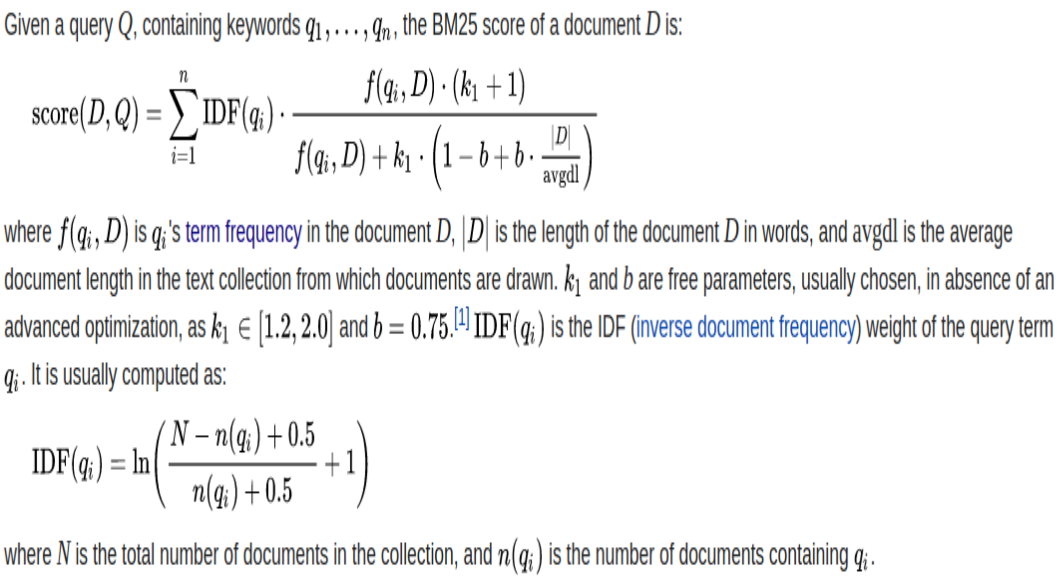
- Problems:
  - Low precision and recall
  - Irrelevant context in responses
  - Augmentation from the retrieved chunks tricky due to redundancies and repetition
  - Novel content generation may not be possible

Search/Retrieval methods for RAG

- Unsupervised approaches
  - Sparse retrieval
  - Dense retrieval
  - Sparse vector retrieval
- Supervised approaches
  - Neural IR
  - Re-ranking models

Lexical or Sparse Retrieval Methods - Unsupervised

- Based on traditional document retrieval approaches
- Breaks up a document and query into tokens
- Match is determined using TF/IDF rules
  - Term Frequency (TF)
    - How often a term occurs in a document
  - Inverse Document Frequency (IDF)
    - How common is such occurrence across documents
- BM25 is a popular sparse retrieval method
  - Uses TF/IDF
  - Document Length Normalization
    - Measure should be robust to document lengths



Given a query Q, containing keywords $q_1, \ldots, q_n$, the BM25 score of a document D is:

$$\text{score}(D,Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i,D) \cdot (k_1+1)}{f(q_i,D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)}$$

where $f(q_i, D)$ is q_i's term frequency in the document D, $|D|$ is the length of the document D in words, and avgdl is the average document length in the text collection from which documents are drawn. k_1 and b are free parameters, usually chosen, in absence of an advanced optimization, as $k_1 \in [1.2, 2.0]$ and $b = 0.75$.[1] $\text{IDF}(q_i)$ is the IDF (inverse document frequency) weight of the query term q_i. It is usually computed as:

$$\text{IDF}(q_i) = \ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

where N is the total number of documents in the collection, and $n(q_i)$ is the number of documents containing q_i.

Sparse retrieval methods:

- Allow exact matching – both a boon and a curse
- Efficient inverted indexing

Dense or Semantic retrieval - Unsupervised

- Given a set of documents
  - Extract chunks from documents
  - Vectorize the chunks using an embedding
- Given a query:
  - Vectorize the query using the same embedding
- Find the closest match to the query using distance or similarity metrics:
  - Inner product
  - Cosine similarity
  - L2 distance
  - Softmax over relevant scores to convert to probability
- When the documents grow large, the vectors could grow to very large vector stores
  - Compression
  - Approximate nearest neighbor search. Through indexing

$$a \cdot b = \sum_{i=1}^{n} a_i b_i$$

$$\cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

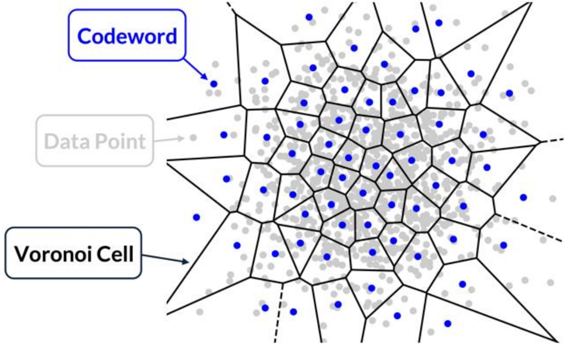
$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

$$p(z \mid x) = \frac{\exp f(x, z)}{\sum_{z'} \exp f(x, z')},$$

$$f(x, z) = \texttt{Embed}_{\texttt{input}}(x)^{\top} \texttt{Embed}_{\texttt{doc}}(z)$$

Large scale vector compression

- Vector quantization

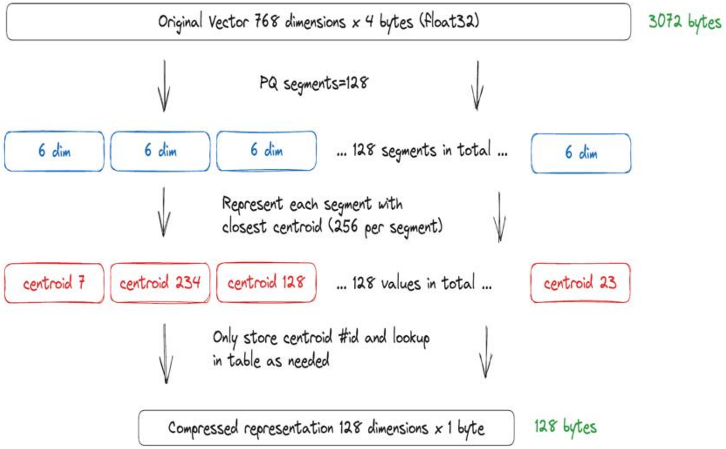


- Lossy compression
  - Data represented by the nearest neighbor codeword
  - Error can be reduced by increasing codewords
  - Works for low-dimensional vectors
  - Reduces search accuracy by 20-30%

- Product Quantization
  - Works for high dimensional vectors
    - Divided into multiple low dimensional segments
    - vector quantization applied for each segment independently
  - Compression errors still depend on the distances between segment vectors and their nearest neighbor codewords

$$d = \sum_{i=1}^{V}(X_i - X'_i)^2+(Y_i - Y'_i)^2$$

$$d = \sum_{i=1}^{S}\sum_{j=1}^{V_s}(X_{ji} - X'_{ji})^2 + (Y_{ji} - Y'_{ji})^2$$


Original Vector 768 dimensions x 4 bytes (float32)
3072 bytes
PQ segments=128
6 dim
6 dim
6 dim
... 128 segments in total ...
6 dim
Represent each segment with closest centroid (256 per segment)
centroid 7
centroid 234
centroid 128
... 128 values in total ...
centroid 23
Only store centroid #id and lookup in table as needed
Compressed representation 128 dimensions x 1 byte
128 bytes

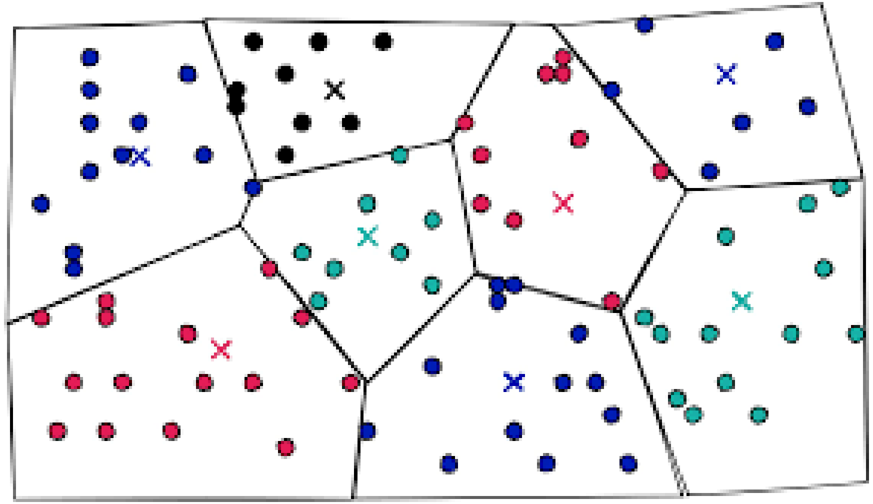
V=3072 #vector dimension
Vs=6 #segment dimension
S=V/Vs #Number of segments
P=32 # bit precision
C=256 #Number of clusters per segment
Pc=8 #bit size needed to represent cluster ID
N=10**10 #Number of vectors =10 billion

$$\text{Compression ratio} = \frac{(N*V*P)}{(N*Pc*S)+(C*Vs*P*S)}$$

Compression ratios a factor of 15 to 25!

Approximate nearest neighbor search

- K-NN algorithms used to reduce search time
- Search accuracy is affected
- Faiss is a popular indexing library supported by many vector db
- Distance metrics include:
    - L2, cosine similarity
- Several indexing algorithms – an active field of research
    - IVF
    - IVF_PQ
    - HNSW
    - Disk ANN



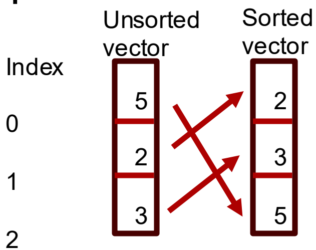
https://www.pinecone.io/learn/series/faiss/product-quantization/

SortPQ – Sorting transformation –balancing search accuracy and compression ratios

Bringing vectors close by finding their commonality

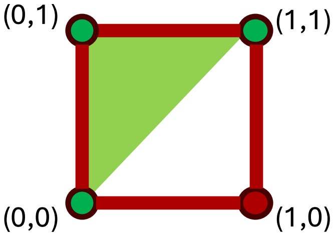
$$\pi_x^*, \pi_y^* = argmin_{\pi_x,\pi_y} |\pi_x(x) - \pi_y(y)|$$

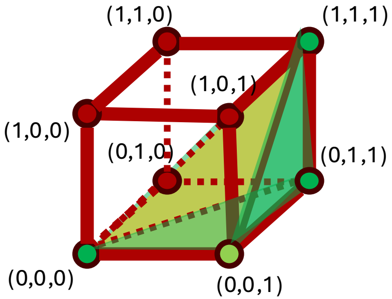
π_x is a permutation for x

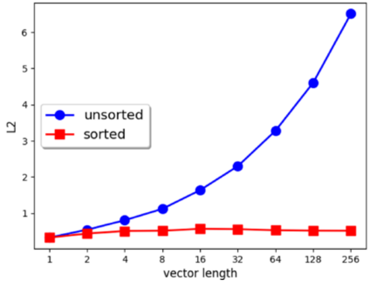


π_y^*

(0,2)(1,0),(2,2)

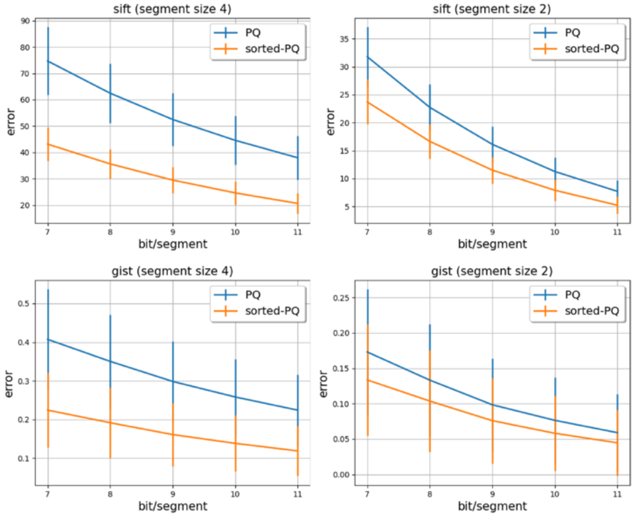






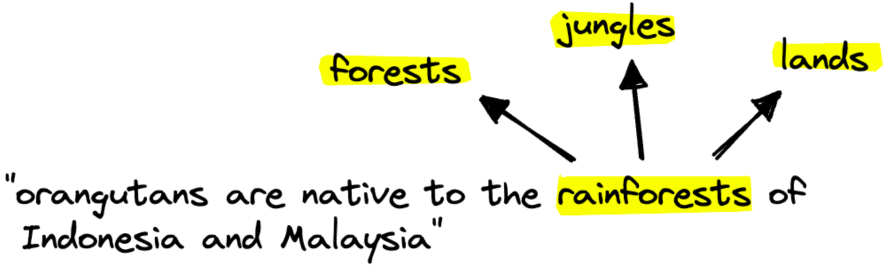
- Vector distance can be reduced by permutation or sorting transformation
- Among all permutation transformations, sorting transformation optimizes following distance/similarity measure for vector data
    - Minimize L2-norm
    - Maximize cosine similarity and Pearson correlation

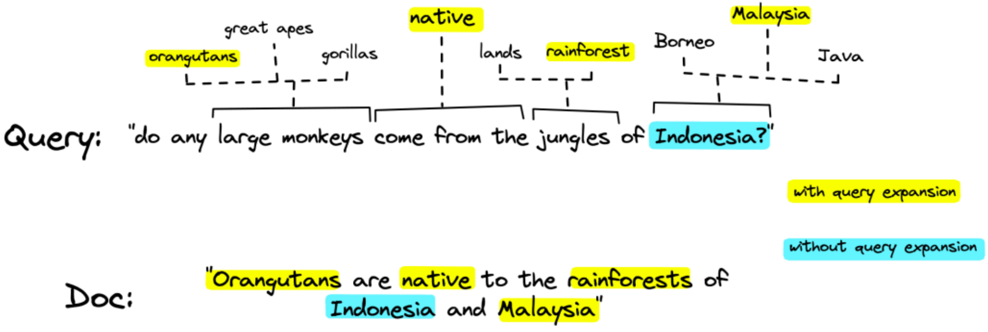
1M SIFT vectors



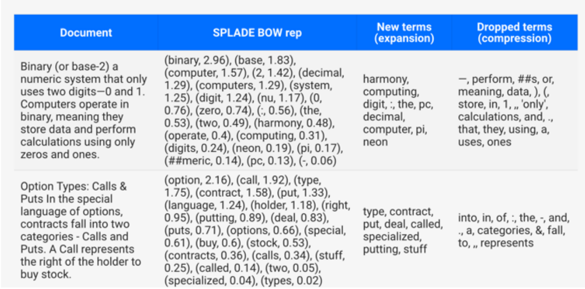
https://ieeexplore.ieee.org/document/10825761

Sparse learned embeddings for retrieval


forests
jungles
lands
"orangutans are native to the rainforests of Indonesia and Malaysia"


great apes
orangutans
gorillas
native
lands
rainforest
Malaysia
Borneo
Java
Query: "do any large monkeys come from the jungles of Indonesia?"
with query expansion
without query expansion
Doc: "Orangutans are native to the rainforests of Indonesia and Malaysia"

https://www.pinecone.io/learn/splade/

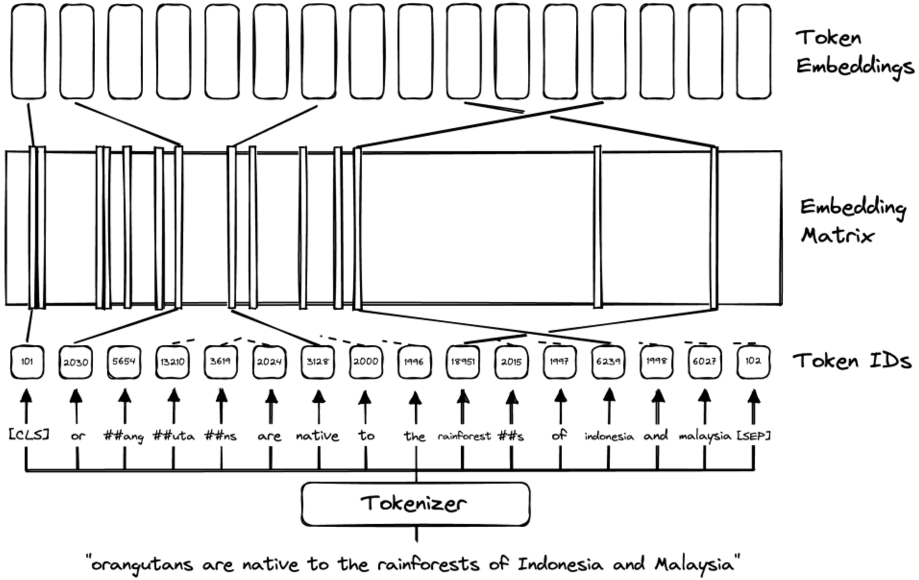


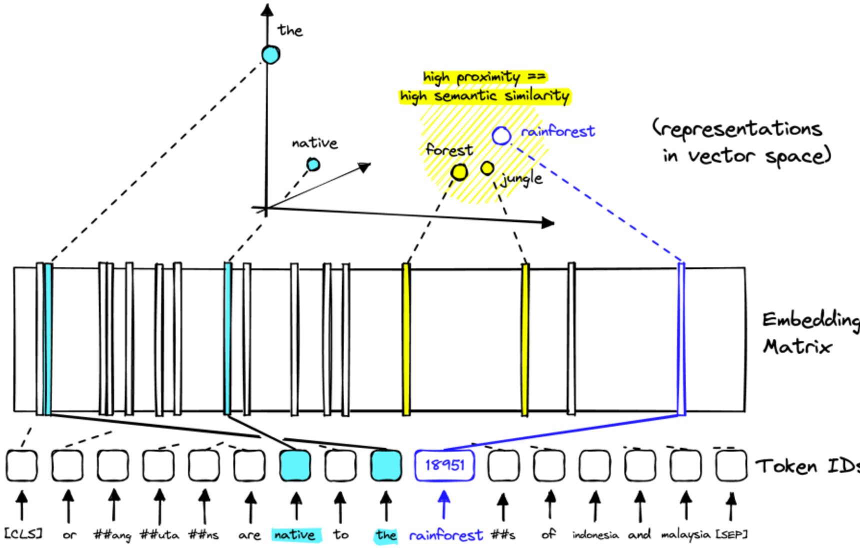
Document	SPLADE BOW rep	New terms (expansion)	Dropped terms (compression)
Binary (or base-2) a numeric system that only uses two digits—0 and 1. Computers operate in binary, meaning they store data and perform calculations using only zeros and ones.	(binary, 2.96), (base, 1.83), (computer, 1.57), (2, 1.42), (decimal, 1.29), (computers, 1.29), (system, 1.25), (digit, 1.24), (nu, 1.17), (0, 0.76), (zero, 0.74), (:, 0.56), (the, 0.53), (two, 0.49), (harmony, 0.48), (operate, 0.4), (computing, 0.31), (digits, 0.24), (neon, 0.19), (pi, 0.17), (##meric, 0.14), (pc, 0.13), (-, 0.06)	harmony, computing, digit, :, the, pc, decimal, computer, pi, neon	—, perform, ##s, or, meaning, data,), (, store, in, 1, ,, 'only', calculations, and, ., that, they, using, a, uses, ones
Option Types: Calls & Puts In the special language of options, contracts fall into two categories - Calls and Puts. A Call represents the right of the holder to buy stock.	(option, 2.16), (call, 1.92), (type, 1.75), (contract, 1.58), (put, 1.33), (language, 1.24), (holder, 1.18), (right, 0.95), (putting, 0.89), (deal, 0.83), (puts, 0.71), (options, 0.66), (special, 0.61), (buy, 0.6), (stock, 0.53), (contracts, 0.36), (calls, 0.34), (stuff, 0.25), (called, 0.14), (two, 0.05), (specialized, 0.04), (types, 0.02)	type, contract, put, deal, called, specialized, putting, stuff	into, in, of, :, the, -, and, ., a, categories, &, fall, to, ,, represents

https://europe.naverlabs.com/blog/splade-a-sparse-b
encoder-bert-based-model-achieves-effective-and-efficient-first-stage-ranking/

SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval, SIGIR 2021

SPLADE model

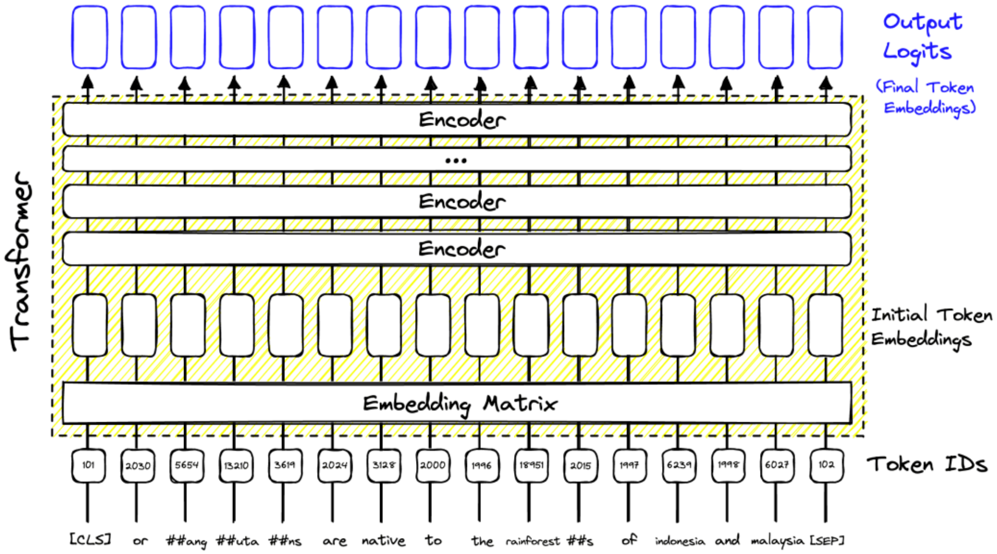


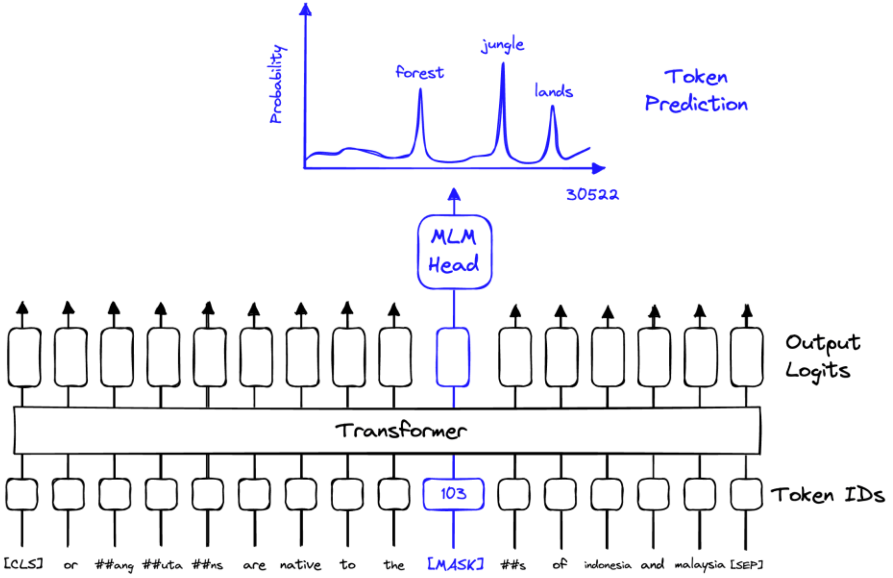


Uses the MLM head of transformer to predict the in-content term expansions

https://www.pinecone.io/learn/splade/

SPLADE model



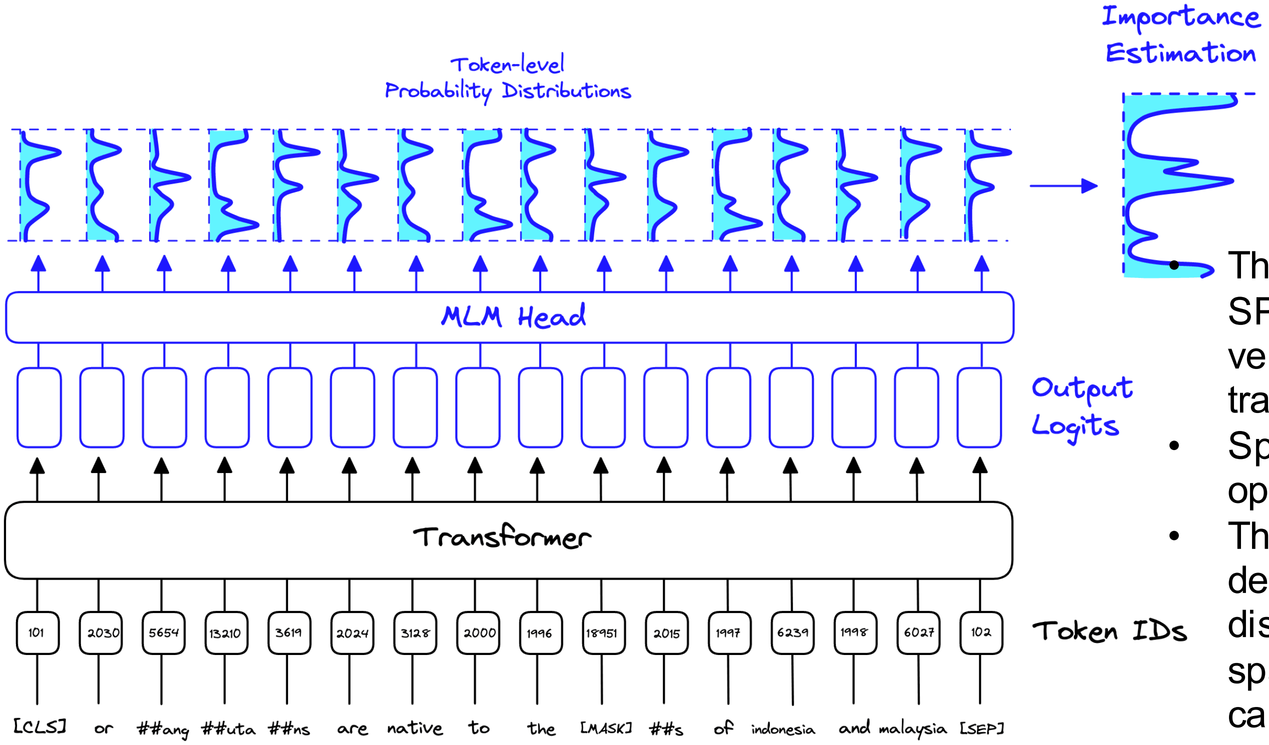


Output logits cover the entire vocabulary

Uses the MLM head of transformer to predict the in-content term expansions
Vocabulary-size vector for each position in the MLM head.

https://www.pinecone.io/learn/splade/

SPLADE model



- The number of non-zero values in SPLADE query and document vectors is typically greater than in traditional sparse vectors
- Sparse retrieval systems are not optimized for this.
- The distribution of non-zero values deviates from the traditional distribution expected by the sparse retrieval systems, again causing slowdowns.

The MLM head gives us a probability distribution for each token, whether or not they have been masked. These distributions are aggregated to give the importance estimation.

Hybrid search methods

- Combine lexical and semantic searches
- Reciprocal rank fusion a popular fusion approach

$$RRF(d) = \Sigma(r \in R)\ 1 / (k + r(d))$$

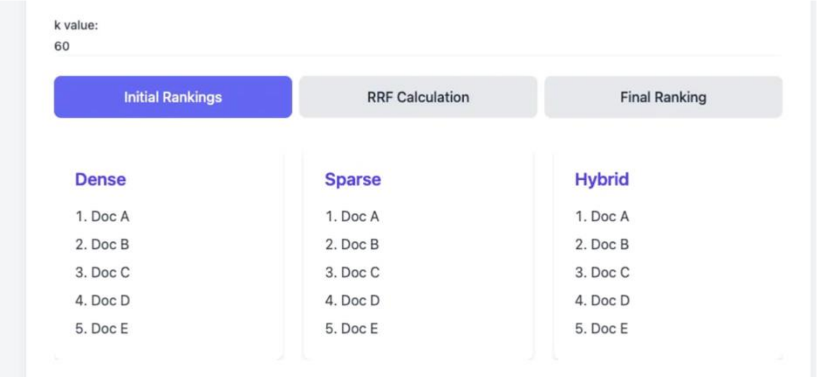
- Where:

  - d is a document

  - R is the set of rankers (retrievers)

  - k is a constant (typically 60)

  - r(d) is the rank of document d in ranker r



k value:
60

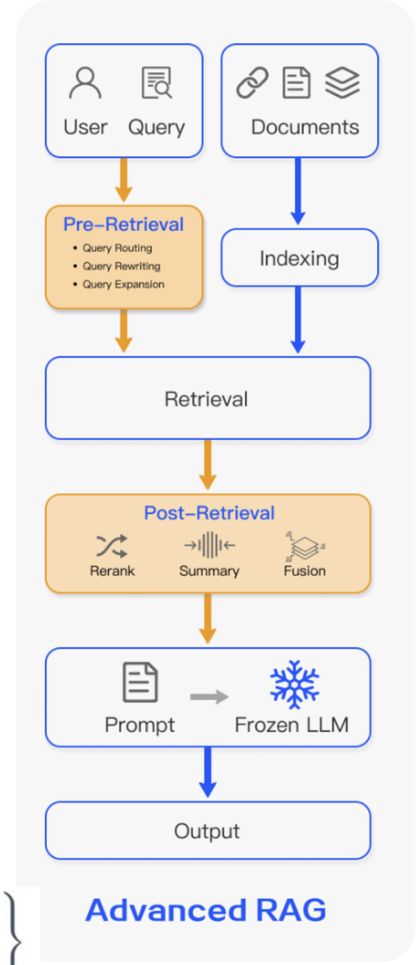
Dense	Sparse	Hybrid
1. Doc A	1. Doc A	1. Doc A
2. Doc B	2. Doc B	2. Doc B
3. Doc C	3. Doc C	3. Doc C
4. Doc D	4. Doc D	4. Doc D
5. Doc E	5. Doc E	5. Doc E

https://deval-shah.github.io/visualizations/rrf/

Advanced RAG Methods

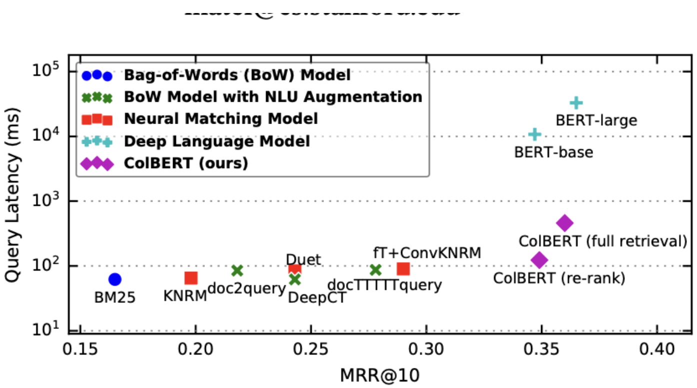
- Indexing
  - Prefiltering text, Decide what to tokenize
  - Adding metadata, Better chunking
- Retrieval
  - Fine tuning embedding (pairs of questions and answers are used to teach the retrieval engine).
    - E.g. BGE model on enterprise knowledge
  - Sparse, Dense, and Multivector models
    - E.g. COLBERT, SpladeV2
  - Dynamic embedding
    - E.g. Open AI's embeddings-ada-02, adapts to the context in which words are used
- Post-retrieval augmentation
  - Re-ranking
    - E.g. Langchain (document diversity, selection of documents)
  - Perplexity-based filtering (mutual information)

$$\mathrm{PPL}(X) = \exp\left\{-\frac{1}{t}\sum_{i}^{t}\log p_\theta(x_i|x_{<i})\right\}$$



Neural IR approaches

- Supervised approaches
  - Training data has pairs of documents and queries and ranked by relevance reflected in the loss function
  - Index time, project the documents using the learned representation
  - At search time, project query and find nearest matches
- Learn a representation of query and document such that the bes matching documents have a higher similarity
- Adapt transformer models like BERT to develop the embeddings
- Much more compute-intensive than unsupervised methods
- Many neural IR approaches can be used for re-ranking and search as well.

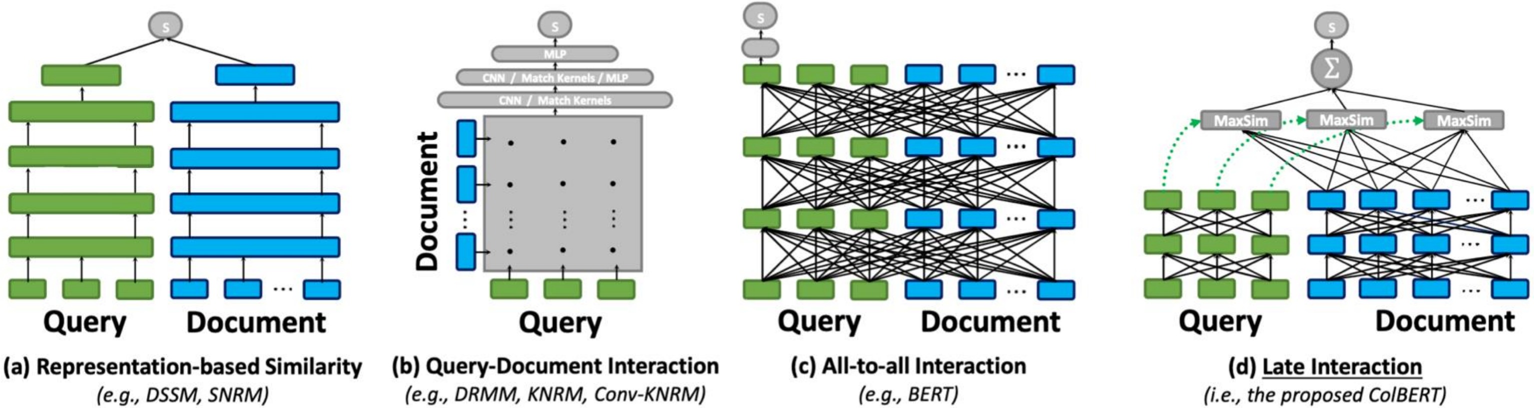


Contextualized Late Interaction over BERT, SIGIR 2020

Neural IR approaches

The supervised approach uses IR data such as labeled query-document pairs, to learn a representation that is optimized for ranking and retrieval

Transformer layers



(a) Representation-based Similarity
(e.g., DSSM, SNRM)

(b) Query-Document Interaction
(e.g., DRMM, KNRM, Conv-KNRM)

(c) All-to-all Interaction
(e.g., BERT)

(d) Late Interaction
(i.e., the proposed ColBERT)

Query and document embeddings separately computed and compared at search time using cosine similarity

An interaction matrix that reflects the similarity between
every pair of words across q and d fed to a classifier for relevance. The similarity matrix is fed to classifiers

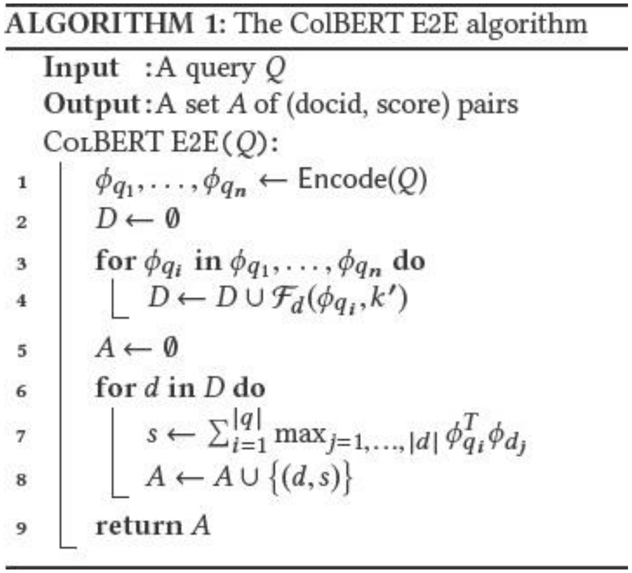
Use a transformer to match and decode to a similarity value

COLBERT

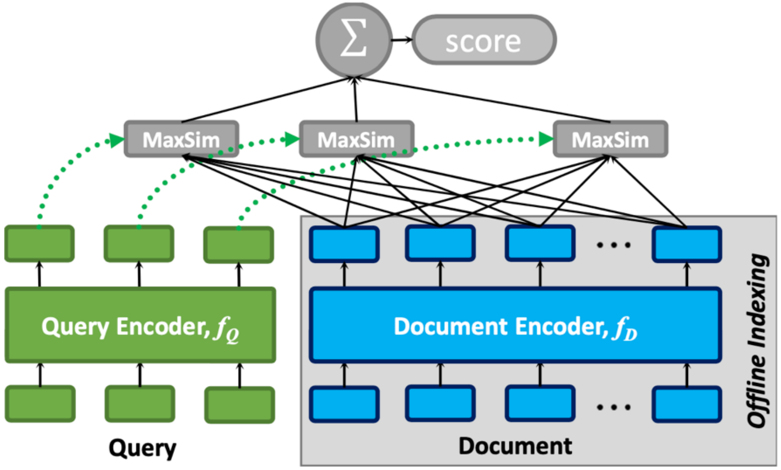
Contextualized Late Interaction over BERT, SIGIR 2020

ColBERT

- Contextualized late interaction over BERT



```
ALGORITHM 1: The ColBERT E2E algorithm
  Input  :A query $Q$
  Output :A set $A$ of (docid, score) pairs
  ColBERT E2E($Q$):
      $\phi_{q_1}, \ldots, \phi_{q_n} \leftarrow$ Encode($Q$)
      $D \leftarrow \emptyset$
      for $\phi_{q_i}$ in $\phi_{q_1}, \ldots, \phi_{q_n}$ do
          $D \leftarrow D \cup \mathcal{F}_d(\phi_{q_i}, k')$
      $A \leftarrow \emptyset$
      for $d$ in $D$ do
          $s \leftarrow \sum_{i=1}^{|q|} \max_{j=1,\ldots,|d|} \phi_{q_i}^T \phi_{d_j}$
          $A \leftarrow A \cup \{(d, s)\}$
      return $A$
```



Contextualized Late Interaction over BERT, SIGIR 2020

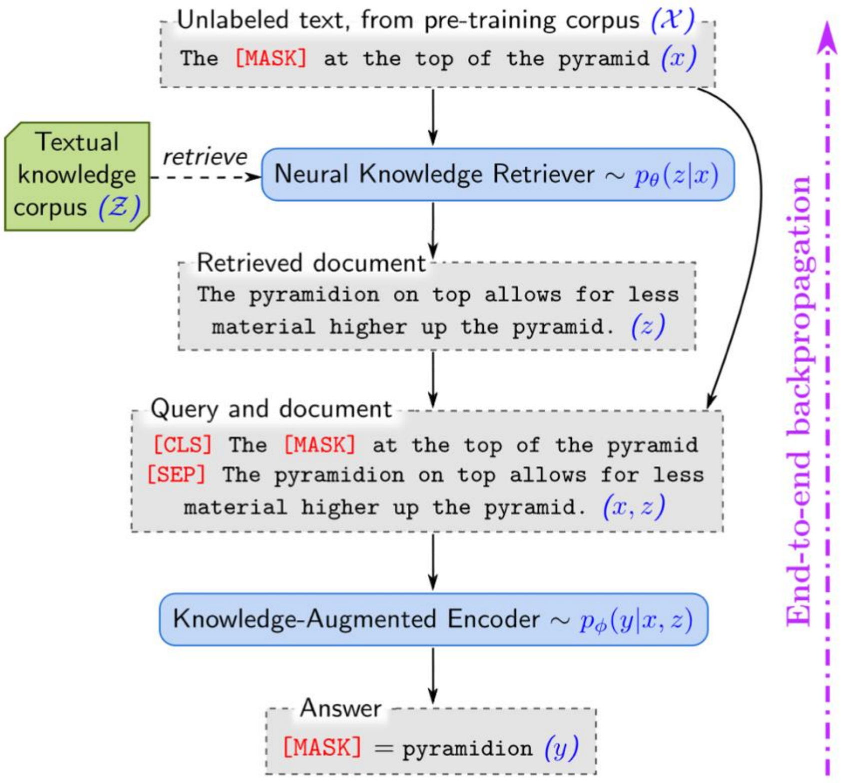
- Only the similarity scores are retained, not the term encodings that matched- Could have two different embedding collide
- Cost of storing the term embeddings of all documents – Huge storage cost!

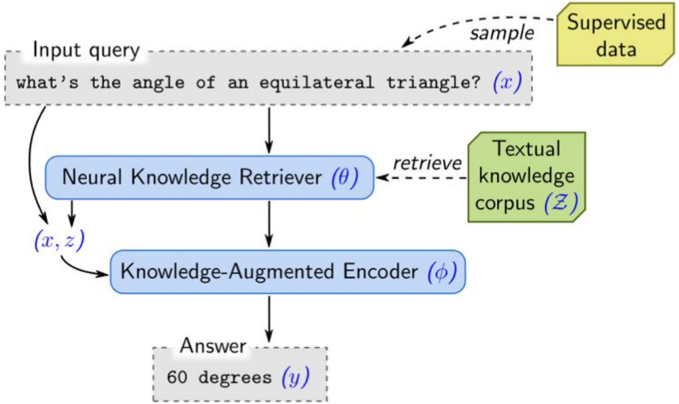
RAG for pre-training and fine-tuning

- RAG in pre-training or fine-tuning:
  - Use external knowledge as a non-parametric memory in addition to parametric seq2seq models.
- Benefits:
  - Results in smaller model sizes
  - Better explainability for model predictions
  - Better adaptability to new information without re-training

REALM

REALM: Retrieval-augmented language model pre-training


Unlabeled text, from pre-training corpus $(\mathcal{X})$
The [MASK] at the top of the pyramid (x)
Textual knowledge corpus $(\mathcal{Z})$
retrieve
Neural Knowledge Retriever $\sim p_\theta(z|x)$
Retrieved document
The pyramidion on top allows for less material higher up the pyramid. (z)
Query and document
[CLS] The [MASK] at the top of the pyramid [SEP] The pyramidion on top allows for less material higher up the pyramid. (x, z)
Knowledge-Augmented Encoder $\sim p_\phi(y|x, z)$
Answer
[MASK] = pyramidion (y)
End-to-end backpropagation


Supervised data
sample
Input query
what's the angle of an equilateral triangle? (x)
Neural Knowledge Retriever (θ)
retrieve
Textual knowledge corpus $(\mathcal{Z})$
(x, z)
Knowledge-Augmented Encoder (ϕ)
Answer
60 degrees (y)

Supervised fine-tuning.

REALM: Retrieval-augmented language model pre-training, SIGIR 2020

Language model pre-training algorithm
Learned neural document retriever using an unsupervised **fill-in-the-blank** training objective.

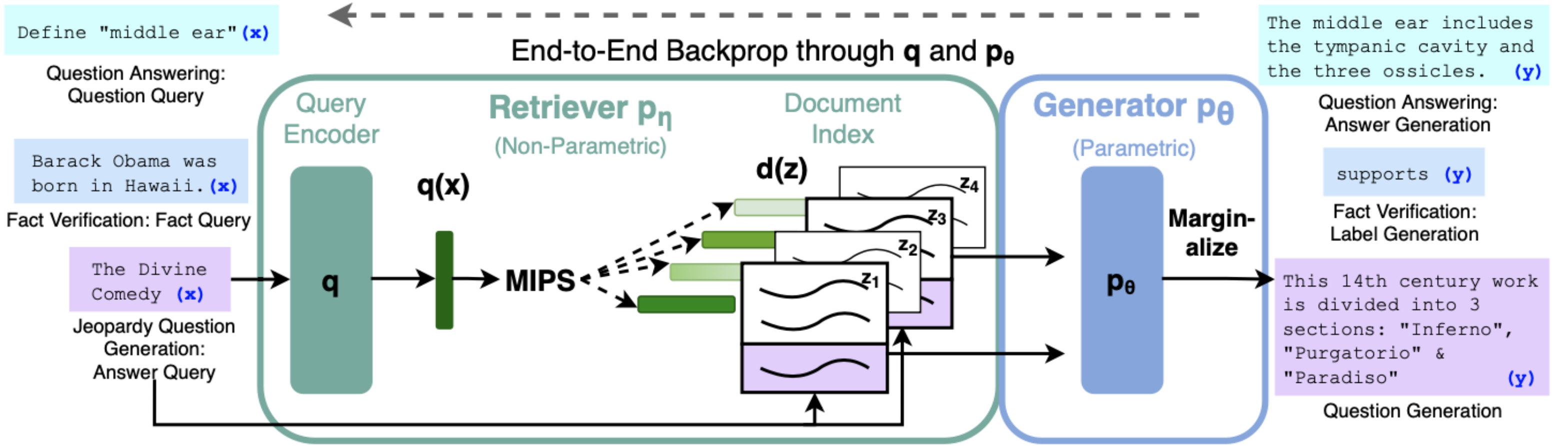
Perplexity score can be used as a loss function for training

RAG – Parametric + Non-parametric memories

- Combines a knowledge retrieval based on vector similarity search
- Once the documents are retrieved, fed along with the prompt to a decoder
- Trained in a supervised manner for downstream seq2seq translation

DPR

BART



Define "middle ear" (x)

Barack Obama was born in Hawaii. (x)

The Divine Comedy (x)

The middle ear includes the tympanic cavity and the three ossicles. (y)

supports (y)

This 14th century work is divided into 3 sections: "Inferno", "Purgatorio" & "Paradiso" (y)

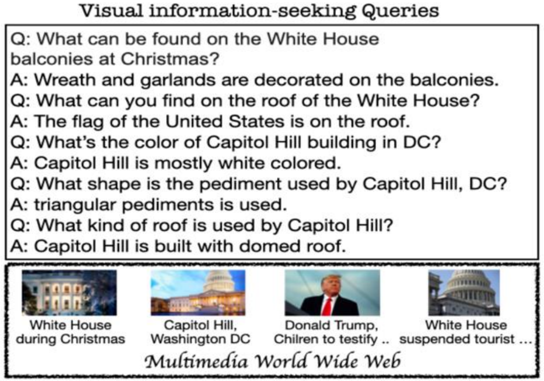
Non-parametric memory

Parametric memory

Retrieval-Augmented Generation for Knowledge-Intensive NLP Task,
NeurIPS 2020

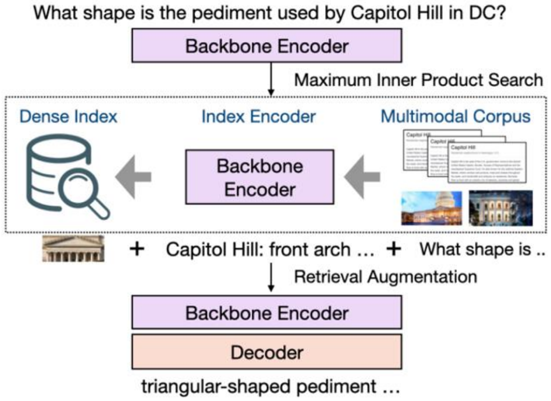
Multimodal RAG for pre-training or fine-tuning

- Uses image and text information to retrieve relevant documents
  - The knowledge to answer questions may lie within images
- Pre-training or fine-tuning with non-parametric multimodal memories
  - Generates more visually grounded output


Visual information-seeking Queries

Q: What can be found on the White House balconies at Christmas?
A: Wreath and garlands are decorated on the balconies.
Q: What can you find on the roof of the White House?
A: The flag of the United States is on the roof.
Q: What's the color of Capitol Hill building in DC?
A: Capitol Hill is mostly white colored.
Q: What shape is the pediment used by Capitol Hill, DC?
A: triangular pediments is used.
Q: What kind of roof is used by Capitol Hill?
A: Capitol Hill is built with domed roof.

White House during Christmas
Capitol Hill, Washington DC
Donald Trump, Chilren to testify ..
White House suspended tourist ...
Multimedia World Wide Web

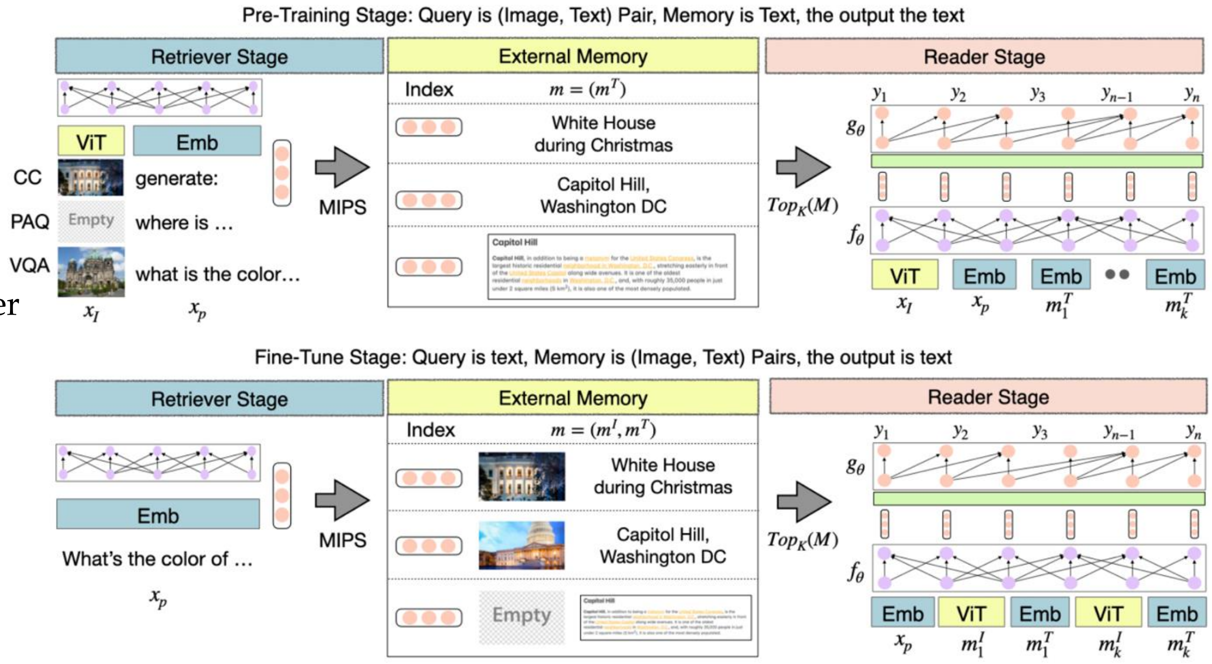

What shape is the pediment used by Capitol Hill in DC?
Backbone Encoder
Maximum Inner Product Search
Dense Index
Index Encoder
Multimodal Corpus
Backbone Encoder
+ Capitol Hill: front arch ... + What shape is ..
Retrieval Augmentation
Backbone Encoder
Decoder
triangular-shaped pediment ...

MuRAG: Multimodal Retrieval-Augmented Generator for Open Question Answering over Images and Text, EMNLP 2022

MURAG – Multimodal RAG

https://arxiv.org/abs/2210.02928

- During pre-training: Train an encoder to produce relevant captions from a textual memory bank.
- During RAG: Use the same encoder for encoding image-text pairs in a multimodal memory bank.
- Input is a textual prompt, the retrieved documents are multimodal, which are then fed to the generator to generate text



Pre-Training Stage: Query is (Image, Text) Pair, Memory is Text, the output the text

Fine-Tune Stage: Query is text, Memory is (Image, Text) Pairs, the output is text

MuRAG: Multimodal Retrieval-Augmented Generator for Open Question Answering over Images and Text, EMNLP 2022

Evaluation Benchmarks

Bier Benchmark

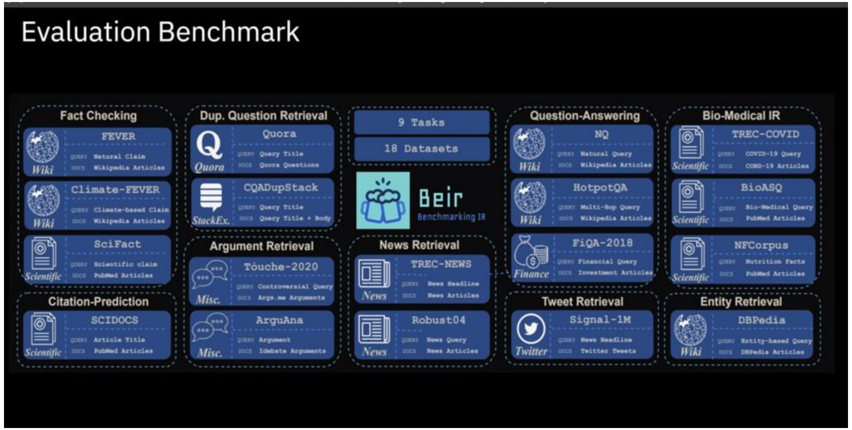
- 18 datasets, 9 tasks

The MTEB benchmark (Multi-Task Evaluation Benchmark)

- 56 datasets distributed among 8 distinct tasks.
- 2000+ outcomes on the leaderboard.

Evaluation metrics

- NDCG (normalized discounted cumulative gain),
- MRR(mean reciprocal rank)
  - It focuses on the position of the first relevant item in the ranked list
  - Weighting is stricter
- MAP
  - Works for binary relevance
- Precision, Recall, Accuracy
  - Recall@K is another popular metric
- NDCG allows for fair comparisons between lists of varying lengths and relevance distribution
- NDCG considers the entire ranking order and assigns higher weights to relevant items at higher positions
- Both binary and numerical scores are handled



Evaluation Benchmark

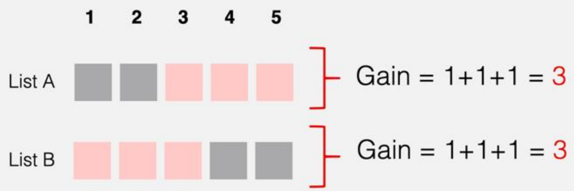
$$\mathrm{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\mathrm{rank}_i}.$$

where rank_i refers to the rank position of the *first* relevant document for the *i*-th query.

NDCG

$$\text{DCG@K} = \sum_{k=1}^{K} \frac{rel_i}{\log_2(i+1)}$$

$$\text{NDCG@K} = \frac{DCG@K}{IDCG@K}$$



Factors affecting RAG performance

- Extraction accuracy of the ingestion pipeline (structural shredders, chunkers, summarizers)
- How it got represented (a function of embedding choices and semantic expansions)
  - What representation capture semantics of documents?
  - What embedding spaces bring queries and documents closer?
- How it was searched (lexical, semantic, fusion, re-ranking)
- How it was aggregated and rolled up for the specific use case (chunk, page, document level roll-up)
- How the query was analyzed (semantic enrichment of the query)
  - Query rewriting
- Prompt engineering to prime the LLM
- Whether RAG is used only during inference or at pre-training or fine-tuning as well

RAG improvements

- Self-Query Retrieval:
- Self-Reflection Loops:
  - Introduce feedback loops where the LLM evaluates its own responses and uses the feedback to refine future retrievals and generations.
  - This process, also known as Self-RAG, helps to improve the accuracy and coherence of the generated text. iterative prompting and correction
- Modular RAG:
  - Aims to build a highly customizable and versatile RAG system.
  - It incorporates various modules for different tasks, such as search, retrieval, and generation, allowing for flexibility and adaptation to different use cases.
- RAG Fusion:
  - Combines the power of RAG with Reciprocal Rank Fusion (RRF).
  - It generates multiple queries, re-ranks the retrieved documents using RRF, and fuses them to create a more comprehensive and relevant knowledge base for the LLM.
- Agentic RAG:
  - Leverages the capabilities of LLMs as agents, granting them access to various tools and information sources. LLM-based query routing to relevant knowledge sources
  - The retrieval component becomes agentic, enabling the LLM to actively query, gather, and process information to generate more accurate and comprehensive responses.