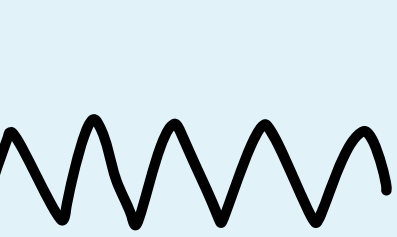
MASTERING LLM PRESENTS

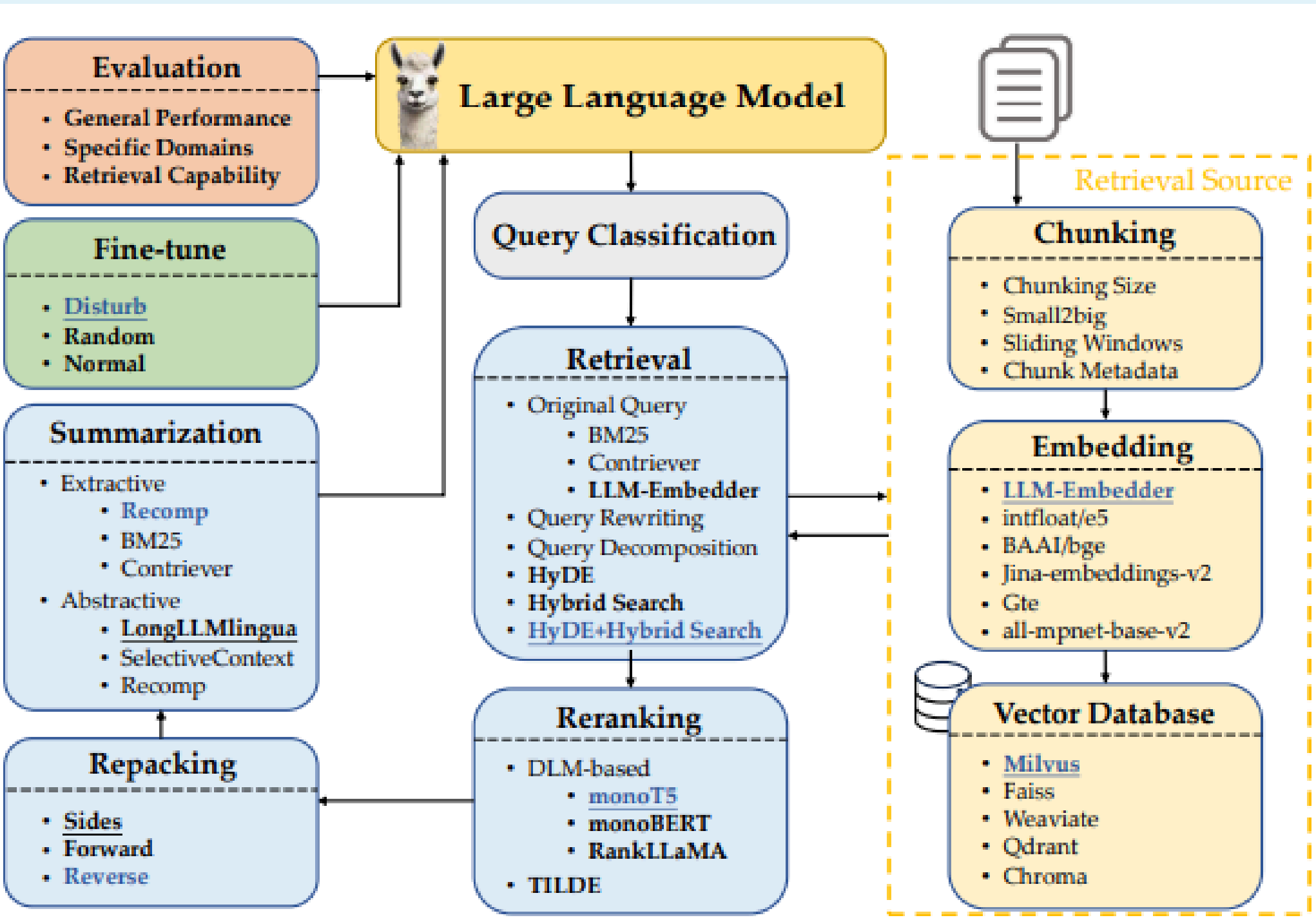
COFFEE BREAK CONCEPTS

Best Practices for RAG pipeline



Follow us on

Typical RAG Workflow



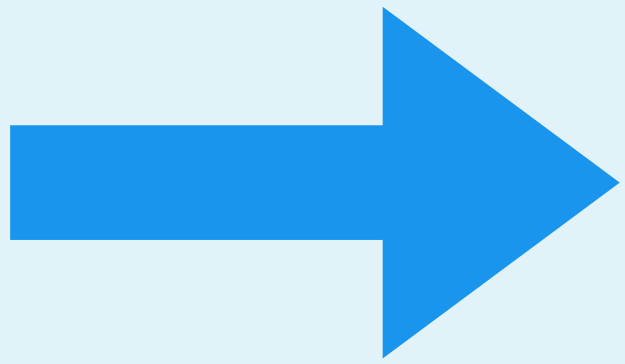
Classification of retrieval requirements for different tasks. In cases where information is not provided, we differentiate tasks based on the functions of the model.
Source: https://arxiv.org/pdf/2407.01219

Lets look into in detail

Typical RAG Workflow

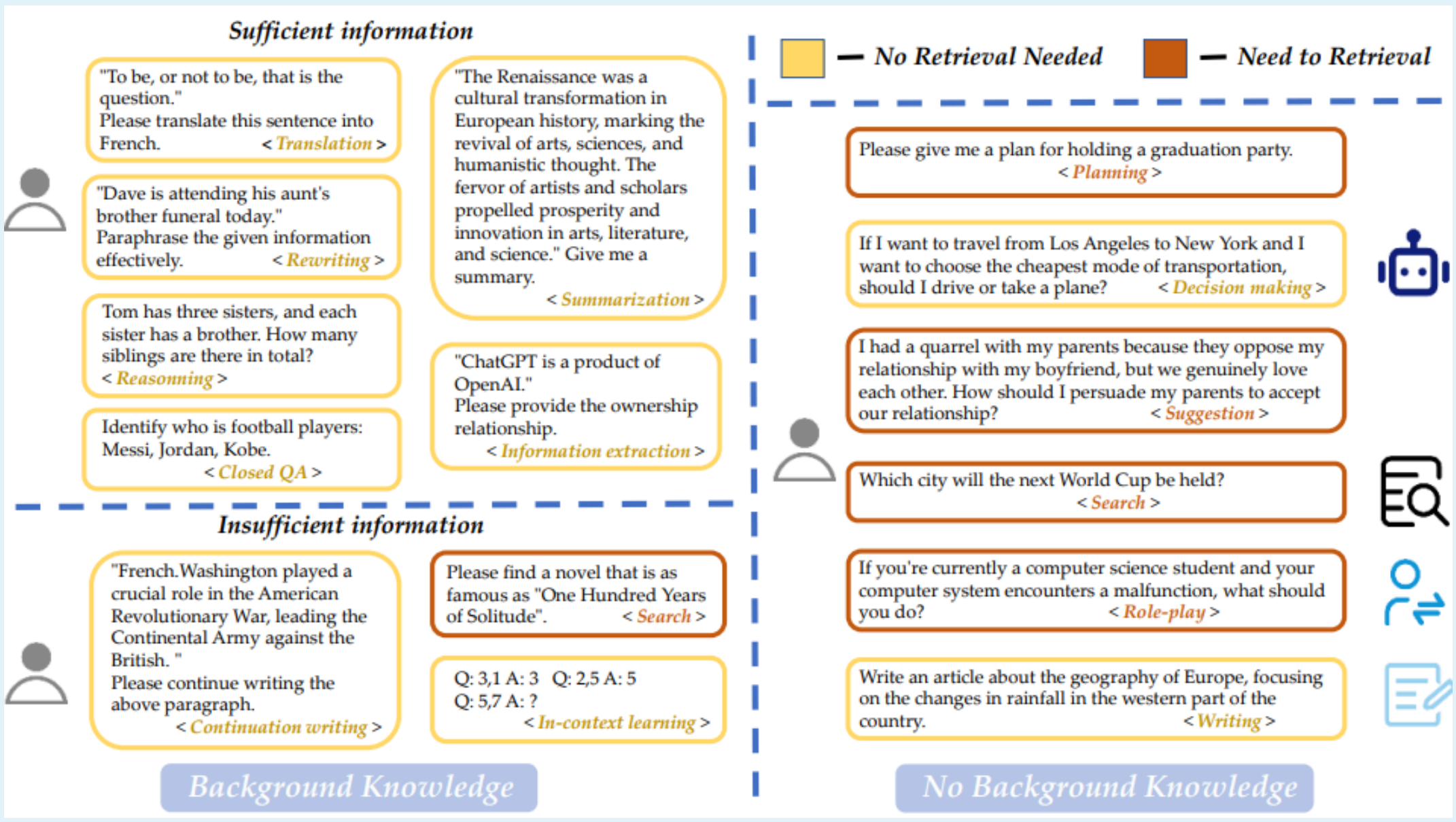
A typical RAG (Retrieval-Augmented Generation) workflow has several steps:

- **Query Classification**: Check if the user's question needs document retrieval.
- **Retrieval:** Find and get the most relevant documents quickly.
- **Re-ranking:** Arrange the retrieved documents in order of relevance.
- **Re-packing:** Organize the documents into a structured format.
- **Summarization:** Extract key points to generate clear, concise answers and avoid repetition.

Each component in detail 

Query Classification

- **LLM Knowledge**: Not all queries need retrieval since LLMs have built-in knowledge.
- **RAG Benefits**: Retrieval can increase accuracy but may slow down responses.
- **When to Retrieve**: Needed when the answer is outside the model's knowledge.
- Task Categories:
  - **Sufficient**: Enough info is already available.
  - **Insufficient**: Additional retrieval may be required.



Classification of retrieval requirements for different tasks

Chunking

- **Document Chunking:** Improves retrieval accuracy and handles length issues in LLMs.
- **Chunking Levels:**
  - **Token-level:** Simple, fixed number of tokens, but may break sentences.
  - **Semantic-level:** Identifies natural breaks, preserves context, requires more processing.
  - **Sentence-level:** Splits at sentence boundaries, balances meaning and efficiency.
- **Preferred Method:** Sentence-level chunking is often favored for its balance of meaning preservation and simplicity.

Embedding Model

- **Embedding Model Selection:** Key for balancing performance and resource use.
- **LLM-Embedder:** Comparable performance to **BAAI/bge-large-en** but one-third the size.
- **Preferred Model: LLM-Embedder** is chosen for its optimal balance of effectiveness and efficiency.
- **Metadata Addition:** Enhancing chunk blocks with metadata like titles, keywords, and hypothetical questions can improve retrieval.

Embedding Model	namespace-Pt/msmarco					
	MRR@1	MRR@10	MRR@100	R@1	R@10	R@100
BAAI/LLM-Embedder [20]	24.79	37.58	38.62	24.07	**66.45**	**90.75**
BAAI/bge-base-en-v1.5 [12]	23.34	35.80	36.94	22.63	64.12	90.13
BAAI/bge-small-en-v1.5 [12]	23.27	35.78	36.89	22.65	63.92	89.80
BAAI/bge-large-en-v1.5 [12]	24.63	37.48	38.59	23.91	65.57	90.60
BAAI/bge-large-en [12]	**24.84**	**37.66**	**38.73**	**24.13**	66.09	90.64
BAAI/bge-small-en [12]	23.28	35.79	36.91	22.62	63.96	89.67
BAAI/bge-base-en [12]	23.47	35.94	37.07	22.73	64.17	90.14
Alibaba-NLP/gte-large-en-v1.5 [21]	8.93	15.60	16.71	8.67	32.28	60.36
thenlper/gte-base [21]	7.42	13.23	14.30	7.21	28.27	56.20
thenlper/gte-small [21]	7.97	14.81	15.95	7.71	32.07	61.08
jinaai/jina-embeddings-v2-small-en [42]	8.07	15.02	16.12	7.87	32.55	60.36
intfloat/e5-small-v2 [11]	10.04	18.23	19.41	9.74	38.92	68.42
intfloat/e5-large-v2 [11]	9.58	17.94	19.03	9.35	39.00	66.11
sentence-transformers/all-mpnet-base-v2	5.80	11.26	12.26	5.66	25.57	50.94

Results for different embedding models on namespace-Pt/msmarco

Vector Databases

- **Vector Databases Comparison:** Includes Weaviate, Faiss, Chroma, Qdrant, and Milvus.
- **Top Choice: Milvus** excels in performance and meets all basic criteria better than other options.

Database	Multiple Index Type	Billion-Scale	Hybrid Search	Cloud-Native
Weaviate	✗	✗	✓	✓
Faiss	✓	✗	✗	✗
Chroma	✗	✗	✓	✓
Qdrant	✗	✓	✓	✓
Milvus	✓	✓	✓	✓

Comparison of Various Vector Databases

Retrieval

- **Query Rewriting:** Enhances query relevance by refining queries with LLMs.
- **Query Decomposition:** Retrieves documents based on sub-questions derived from the original query.
- **Pseudo-Document Generation:** Uses hypothetical documents to find similar documents; HyDE is a notable example.

Method	TREC DL19					TREC DL20				
	mAP	nDCG@10	R@50	R@1k	Latency	mAP	nDCG@10	R@50	R@1k	Latency
unsupervised										
BM25	30.13	50.58	38.32	75.01	**0.07**	28.56	47.96	46.18	78.63	**0.29**
Contriever	23.99	44.54	37.54	74.59	3.06	23.98	42.13	43.81	75.39	0.98
supervised										
LLM-Embedder	44.66	70.20	49.06	84.48	2.61	45.60	68.76	61.36	84.41	0.71
+ Query Rewriting	44.56	67.89	51.45	85.35	7.80	45.16	65.62	59.63	83.45	2.06
+ Query Decomposition	41.93	66.10	48.66	82.62	14.98	43.30	64.95	57.74	84.18	2.01
+ HyDE	50.87	**75.44**	54.93	88.76	7.21	50.94	**73.94**	63.80	88.03	2.14
+ Hybrid Search	47.14	72.50	51.13	89.08	3.20	47.72	69.80	64.32	88.04	0.77
+ HyDE + Hybrid Search	**52.13**	73.34	**55.38**	**90.42**	11.16	**53.13**	72.72	**66.14**	**90.67**	2.95

Evaluation Results (Figure above):

- **Supervised vs. Unsupervised:** Supervised methods outperform unsupervised methods.
- **Best Performance:** HyDE + Hybrid Search achieved the highest performance score.
- **Hybrid Search:** Integrates BM25 (sparse retrieval) and LLM embeddings (dense retrieval) for balanced performance and low latency.

✓ **Recommendation: Use HyDE + hybrid search as the default retrieval method.**

Re-ranking

Re-Ranking Phase: Refines document relevance following initial retrieval.

Re-Ranking Methods:

- **DLM Re-Ranking:** Utilizes Deep Language Models to classify document relevance, ranking documents based on the probability of being "true."
- **TILDE Re-Ranking:** Scores documents by summing log probabilities of query terms. TILDEv2 enhances efficiency by indexing only relevant terms and using NCE loss.

Method	MS MARCO Passage ranking						
	Base Model	# Params	MRR@1	MRR@10	MRR@1k	Hit Rate@10	Latency
w/o Reranking							
Random Ordering	-	-	0.011	0.027	0.068	0.092	-
BM25	-	-	6.52	11.65	12.59	24.63	-
DLM Reranking							
monoT5	T5-base	220M	21.62	31.78	32.40	54.07	**4.5**
monoBERT	BERT-large	340M	21.65	31.69	32.35	53.38	15.8
RankLLaMA	Llama-2-7b	7B	**22.08**	**32.35**	**32.97**	**54.53**	82.4
TILDE Reranking							
TILDEv2	BERT-base	110M	18.57	27.83	28.60	49.07	**0.02**

Evaluation Results (Figure above):

- **monoT5:** Recommended for a balance of performance and efficiency.
- **RankLLaMA:** Best for optimal performance.
- **TILDEv2:** Suitable for quick experiments with fixed sets

✓ **Recommendation: Use monoT5 for comprehensive performance and efficiency.**

Re-packing

Re-Packing Module: Ensures effective LLM response generation by optimizing document order after re-ranking.

Re-Packing Methods:

- **Forward:** Orders documents in descending relevance.
- **Reverse:** Orders documents in ascending relevance.
- **Sides:** Places relevant information at the beginning or end, based on the "**Lost in the Middle**" concept.

Evaluation: Detailed assessment of these methods will be covered in next sections.

Summarization

Importance of Summarization: Reduces redundancy and prevents long prompts from slowing down inference in LLMs.

Summarization Methods:

- **Extractive Compressors:** Segment and rank sentences based on importance.
- **Generative Compressors:** Synthesize and rephrase information from multiple documents.

Method	NQ		TQA		HotPotQA		Avg.	Avg. Token
	F1	#token	F1	#token	F1	#token		
w/o Summarization								
Origin Prompt	27.07	124	33.61	152	33.92	141	31.53	139
Extractive Method								
BM25	27.97	40	32.44	59	28.00	63	29.47	54
Contriever	23.62	42	33.79	65	23.64	60	27.02	56
Recomp (extractive)	27.84	34	35.32	60	29.46	58	30.87	51
Abstractive Method								
SelectiveContext	25.05	65	34.25	70	**34.43**	66	31.24	67
LongLLMlingua	21.32	51	32.81	56	30.79	57	28.29	55
Recomp (abstractive)	**33.68**	59	**35.87**	61	29.01	57	**32.85**	59

Comparison between different summarization methods

Evaluated Methods:

- **Recomp:** Uses both extractive and generative compressors to select and synthesize important information.
- **LongLLMLingua:** Focuses on key query-relevant information for improved summarization.
- **Selective Context:** Enhances LLM efficiency by removing redundant information.

✓ **Recommendation: Recomp is preferred, with LongLLMLingua as an alternative**

Generator Fine-tuning

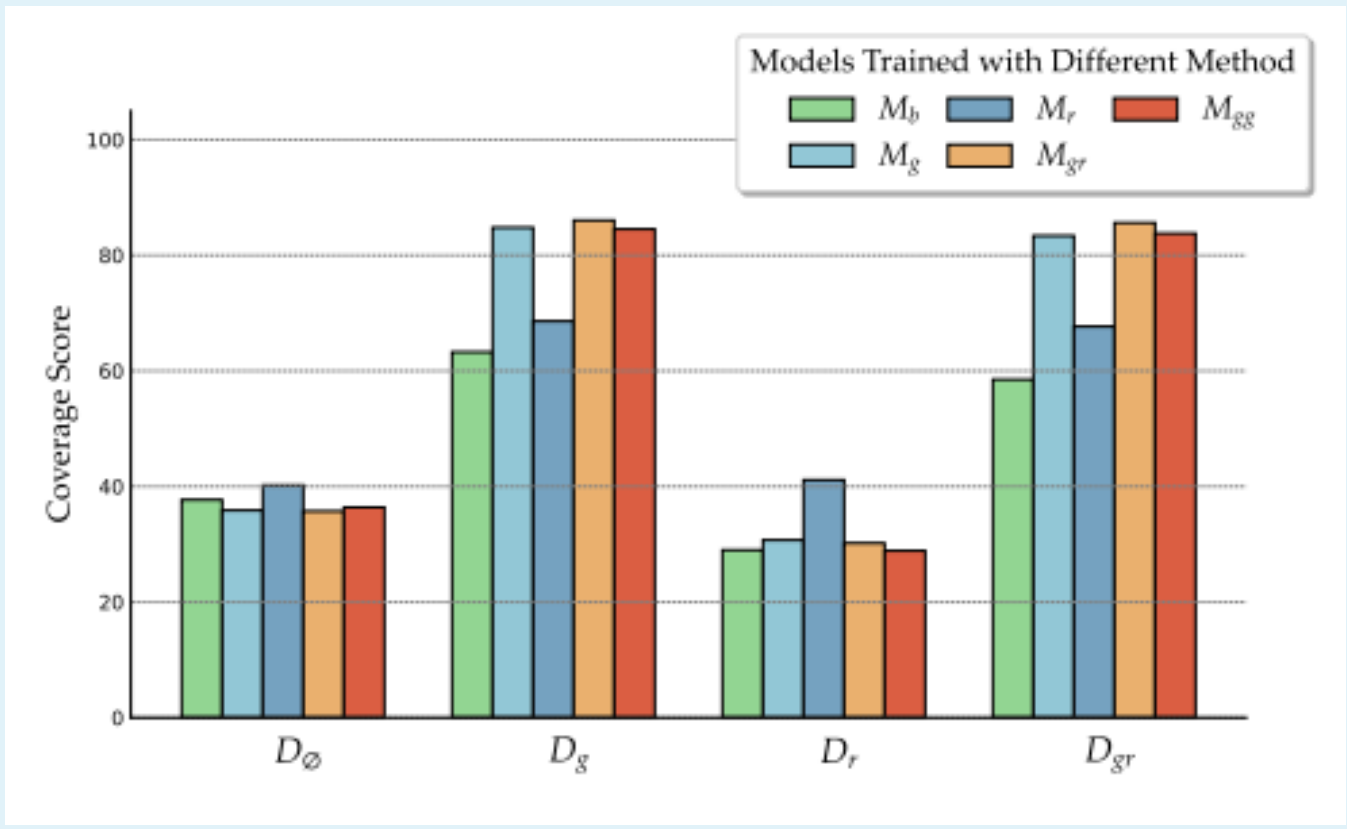
Investigates how fine-tuning with different contexts affects the generator's performance, leaving retriever fine-tuning for future work.

Training Context Variants:

- **Dg:** Only query-relevant documents.
- **Dr:** Only randomly sampled documents.
- **Dgr:** Mix of one relevant and one random document.
- **Dgg:** Two copies of a relevant document.

Model Variants:

- **Mb:** Base LM generator (not fine-tuned).
- **Mg, Mr, Mgr, Mgg:** Models fine-tuned with contexts Dg, Dr, Dgr, and Dgg respectively



✓ **Training with Mixed Documents:** The model trained with a mix of relevant and random documents (**Mgr**) shows the best performance with gold or mixed context.

Searching for Best RAG Practices

Method	Commonsense	Fact Check	ODQA		Multihop		Medical	RAG	Avg.		
	Acc	Acc	EM	F1	EM	F1	Acc	Score	Score	F1	Latency
	classification module, Hybrid with HyDE, monoT5, sides, Recomp										
w/o classification	0.719	0.505	0.391	**0.450**	**0.212**	0.255	**0.528**	0.540	0.465	**0.353**	16.58
+ **classification**	**0.727**	**0.595**	**0.393**	**0.450**	0.207	**0.257**	0.460	**0.580**	**0.478**	**0.353**	**11.71**
	with classification, *retrieval module*, monoT5, sides, Recomp										
+ HyDE	0.718	**0.595**	0.320	0.373	0.170	0.213	0.400	0.545	0.443	0.293	11.58
+ Original	0.721	0.585	0.300	0.350	0.153	0.197	0.390	0.486	0.428	0.273	**1.44**
+ Hybrid	0.718	**0.595**	0.347	0.397	0.190	0.240	**0.750**	0.498	0.477	0.318	1.45
+ **Hybrid with HyDE**	**0.727**	**0.595**	**0.393**	**0.450**	**0.207**	**0.257**	0.460	**0.580**	**0.478**	**0.353**	11.71
	with classification, Hybrid with HyDE, *reranking module*, sides, Recomp										
w/o reranking	0.720	0.591	0.365	0.429	0.211	**0.260**	**0.512**	0.530	0.470	0.334	**10.31**
+ **monoT5**	**0.727**	0.595	0.393	0.450	0.207	0.257	0.460	**0.580**	**0.478**	0.353	11.71
+ monoBERT	0.723	0.593	0.383	0.443	**0.217**	0.259	0.482	0.551	0.475	0.351	11.65
+ RankLLaMA	0.723	**0.597**	0.382	0.443	0.197	0.240	0.454	0.558	0.470	0.342	13.51
+ TILDEv2	0.725	0.588	**0.394**	**0.456**	0.209	0.255	0.486	0.536	0.476	**0.355**	11.26
	with classification, Hybrid with HyDE, monoT5, *repacking module*, Recomp										
+ sides	0.727	0.595	**0.393**	**0.450**	0.207	0.257	0.460	**0.580**	0.478	0.353	11.71
+ forward	0.722	**0.599**	0.379	0.437	0.215	0.260	0.472	0.542	0.474	0.349	**11.68**
+ **reverse**	**0.728**	0.592	0.387	0.445	**0.219**	**0.263**	**0.532**	0.560	**0.483**	**0.354**	11.70
	with classification, Hybrid with HyDE, monoT5, reverse, *summarization module*										
w/o summarization	**0.729**	0.591	**0.402**	**0.457**	0.205	0.252	0.528	0.533	0.480	**0.355**	**10.97**
+ **Recomp**	0.728	**0.592**	0.387	0.445	**0.219**	**0.263**	**0.532**	**0.560**	**0.483**	0.354	11.70
+ LongLLMLingua	0.713	0.581	0.362	0.423	0.199	0.245	0.530	0.539	0.466	0.334	16.17

- **Modules** under investigation are enclosed in a boxed module.
- **Selected implementation** is indicated with an underlined method.
- **Average Score (Avg)**: Calculated using Accuracy (Acc), Exact Match (EM), and RAG scores across all tasks.
- **Average Latency**: Measured in seconds per query.
- **Best Scores**: Highlighted in bold.

Summary

Query Classification Module:

- **Improvement:** Increases effectiveness and efficiency.
- **Performance Metrics:** Average score increased from 0.428 to 0.443.
- **Latency:** Reduced from 16.41 seconds to 11.58 seconds.

Retrieval Module:

- **Best Performance:** "**Hybrid with HyDE**" achieved the highest RAG score (0.58) but has high computational cost (11.71 seconds per query).
- **Recommendation:** Use "**Hybrid**" or "**Original**" methods to balance performance and latency.

Re-Ranking Module:

- **Impact:** Absence leads to significant performance drop.
- **Best Method: MonoT5** achieved the highest average score, highlighting the importance of re-ranking for relevance.

Summary

Re-Packing Module:

- **Best Configuration:** Reverse configuration achieved a RAG score of 0.560.
- **Insight:** Placing more relevant context closer to the query yields better results.

Summarization Module:

- **Best Method:** Recomp demonstrated superior performance.
- **Alternative:** Removing the summary module can yield comparable results with lower latency, but **Recomp** remains preferred for handling the generator's maximum length limitation.

www.masteringllm.com

LLM Interview Course

Want to Prepare yourself for an LLM Interview?

- ✓ 100+ Questions spanning 14 categories
- ✓ Curated 100+ assessments for each category
- ✓ Well-researched real-world interview questions based on FAANG & Fortune 500 companies
- ✓ Focus on Visual learning
- ✓ Real Case Studies & Certification

Coupon Code - LLM50

Coupon is valid till 30th Aug 2024

www.masteringllm.com

AgenticRAG with LlamaIndex 

Want to learn why AgenticRAG is future of RAG?

- Master **RAG fundamentals** through practical case studies
- Understand how to overcome **limitations of RA**G
- Introduction to **AgenticRAG** & techniques like **Routing Agents, Query planning agents, Structure planning agents, and React agents with human in loop**.
- **5** real-time **case studies with code walkthroughs**