RAG Doc:

<https://github.com/NirDiamant/RAG_Techniques/blob/main/all_rag_techniques/reliable_rag.ipynb>

**Reliable RAG**

* Enhances traditional RAG approach by adding layers of validation and refinement to ensure accuracy and relevance of retrieved information.

Key components:

1. Document loading and chunking
2. Vector store creation
3. Document relevancy check
4. Answer generation
5. Hallucination detection
6. Document snippet highlighting

**Implementation Steps:**

1. Document loading and chunking
2. VectoreStore creation
3. Document Relevancy check
4. Answer generation
5. Hallucination detection
6. Document snippet highlighting

**Proposition Chunking**

It breaks down the input text into proposition that are atomic, factual, self-contained and concise in nature and encodes the proposition into vector store which can be later used for retrieval.

**Implementation Steps:**

1. Document Chunking
2. Proposition Generation

Proposition are generated from each chunk using **LLM**. The output is structured as a list of factual, self-contained statements that can be understood without additional context.

1. Quality checks

A second LLM evaluates the quality of the propositions by scoring them on accuracy, clarity, completeness and conciseness.

1. Embedding Proposition

Propositions that pass quality checks are embedded into a vector store.

1. Retrieval and comparison

**Query Transformation**

Modifying and expanding queries to improve retrieval effectiveness.

1. Query re-writing
2. Step-back prompting
3. Sub-query decomposition

Each query aims to improve relevance and comprehensiveness of retrieved information by modifying or expanding original query.

**Motivation:**

RAG system often faces challenges in retrieving most relevant information. These query transformation addresses these issues by reformulating queries to better match relevant documents.

1. **Query re-writing**

Takes the original query and reformulates it to be more specific and detailed.

1. **Step-back prompting**

To generate broader, more general queries that can help retrieve relevant background information.

1. **Sub-query Decomposition**

To break down complex queries into simpler sub-queries

**Example Use case:**

**Query:** "What are the impacts of climate change on the environment?"

* **Query Rewriting** expands this to include specific aspects like temperature changes and biodiversity.
* **Step-back Prompting** generalizes it to "What are the general effects of climate change?"
* **Sub-query Decomposition** breaks it down into questions about biodiversity, oceans, weather patterns, and terrestrial environments.

<https://github.com/NirDiamant/RAG_Techniques/blob/main/all_rag_techniques/query_transformations.ipynb>

**Hypothetical Document Embedding (HyDE)**