A graph of a distribution of text lengths

Description automatically generated

A graph of different colored bars

Description automatically generated

**🔍 Insights from EDA**

**1️⃣ Text Length Variability**

* The text length distribution shows that **documents vary widely in length**, ranging from **1000 to over 5000 words**.
* This suggests that some documents are **much longer than others**, which may:
  + Affect **retrieval effectiveness**, as longer documents might contain **multiple unrelated topics**.
  + Require **chunking strategies** to ensure retrieved content remains **contextually relevant**.

**2️⃣ Common Words are Mostly Stopwords**

* The **most common words** in the dataset are generic stopwords: **"the," "of," "to," "in," "and,"** etc.
* These words contribute little to **semantic retrieval** and could lead to **inefficient search queries**.
* **Possible Solution:** Stopword removal **during vectorization** (but not before retrieval) can improve search performance.

**3️⃣ Potential Data Imbalance**

* If **certain words or topics dominate the dataset**, the model might be biased towards retrieving certain types of information.
* For example, if financial terms frequently appear, queries on unrelated topics might **fail to retrieve useful results**.

**⚠️ Challenges and Their Impact on Retrieval & Generation**

**1️⃣ Document Chunking for Retrieval**

✅ **Why It Matters?**

* Since some documents are **very long**, embedding an **entire document** could lead to **irrelevant retrieval** (e.g., a query about "pricing models" retrieving a whole 5000-word article).

🔹 **Solution:**

* **Chunk documents into smaller segments** (~300–500 words per chunk).
* Store metadata like **document title, section headers, and keywords** to improve retrieval precision.

**2️⃣ Stopwords and Poor Retrieval Precision**

✅ **Why It Matters?**

* If **stopwords dominate**, keyword-based retrieval might fail to rank relevant documents correctly.
* Without proper preprocessing, queries might retrieve **high-frequency but irrelevant** documents.

🔹 **Solution:**

* **Use TF-IDF weighting or embeddings (e.g., Sentence Transformers)** to prioritize **meaningful words** over stopwords.
* Consider stopword removal **only for embeddings and retrieval, not generation**, to preserve fluency.

**3️⃣ Semantic Overlap Between Documents**

✅ **Why It Matters?**

* If many documents **cover similar content**, retrieval might return **multiple redundant results**.
* The LLM could **generate repetitive or generic responses** due to high document similarity.

🔹 **Solution:**

* Implement **diversity filters** in retrieval (e.g., **max marginal relevance (MMR)** in vector search).
* If documents are very similar, use **reranking models** to improve retrieval variety.

**4️⃣ Handling Ambiguous Queries**

✅ **Why It Matters?**

* If a query is **short or ambiguous**, retrieval might **not find the best matching chunk**.
* Example: A query like **"interest rates"** could refer to:
  + **Banking regulations**
  + **Economic indicators**
  + **Loan policies**

🔹 **Solution:**

* Improve **query expansion** (e.g., using LLMs to rewrite queries into more specific questions).
* Use **metadata filtering** to narrow down document context.

**5️⃣ Generation Challenges (LLM Stage)**

✅ **Why It Matters?**

* If retrieval returns **too much or irrelevant information**, the LLM might **generate incomplete, hallucinated, or low-quality responses**.

🔹 **Solution:**

* **Experiment with prompt engineering**:
  + Add **retrieved context** explicitly to the prompt.
  + Use **structured templates** (e.g., "Based on the document about [TOPIC], the answer is: ...").
* **Use RAG evaluation techniques**:
  + Compare **LLM-generated answers** with ground truth responses.
  + Use **LLM as a judge** to rank response quality.

**🚀 Final Recommendations**

| **Issue** | **Potential Fix** |
| --- | --- |
| **Long documents** | Chunk text into 300-500 word segments |
| **Stopwords dominate retrieval** | Remove stopwords **only during vectorization** |
| **Semantic overlap (redundant retrieval)** | Use **Max Marginal Relevance (MMR) reranking** |
| **Ambiguous queries lead to poor retrieval** | Use **query expansion** techniques |
| **LLM hallucinations** | Improve prompt engineering and filtering |

**📌 Next Steps**

* Implement **document chunking** and store metadata.
* Optimize **retrieval ranking** to ensure **highly relevant documents**.
* Experiment with **query expansion** for better search accuracy.
* Tune the **LLM prompt** to maximize **contextual accuracy**.