Lost in the middle: The problem with long contexts

No matter the architecture of your model, there is a substantial performance degradation when you include 10+ retrieved documents. In brief: When models must access relevant information in the middle of long contexts, then tend to ignore the provided documents. See: https://arxiv.org/abs/2307.03172

To avoid this issue you can re-order documents after retrieval to avoid performance degradation.

```
import os
import chromadb
from langchain.vectorstores import Chroma
from langchain.embeddings import HuggingFaceEmbeddings
from langchain.document_transformers import (
    LongContextReorder,
)
from langchain.chains import StuffDocumentsChain, LLMChain
from langchain.prompts import PromptTemplate
from langchain.llms import OpenAI

# Get embeddings.
embeddings = HuggingFaceEmbeddings(model_name="all-MiniLM-L6-v2")

texts = [
    "Basquetball is a great sport.",
    "Fly me to the moon is one of my favourite songs.",
    "The Celtics are my favourite team.",
    "This is a document about the Boston Celtics",
    "I simply love going to the movies",
    "The Boston Celtics won the game by 20 points",
    "This is just a random text.",
    "Elden Ring is one of the best games in the last 15 years.",
    "L. Kornet is one of the best Celtics players.",
    "Larry Bird was an iconic NBA player.",
]

# Create a retriever
```

```
retriever = Chroma.from_texts(texts,
embedding=embeddings).as_retriever(
    search_kwargs={"k": 10}
)
query = "What can you tell me about the Celtics?"

# Get relevant documents ordered by relevance score
docs = retriever.get_relevant_documents(query)
docs
```

API Reference:

- Chroma from `langchain.vectorstores`
- HuggingFaceEmbeddings from `langchain.embeddings`
- LongContextReorder from `langchain.document_transformers`
- StuffDocumentsChain from `langchain.chains`
- LLMChain from `langchain.chains`
- PromptTemplate from `langchain.prompts`
- OpenAI from `langchain.llms`

```
    [Document(page_content='This is a document about the Boston
Celtics', metadata={}),
     Document(page_content='The Celtics are my favourite team.',
metadata={}),
     Document(page_content='L. Kornet is one of the best Celtics
players.', metadata={}),
     Document(page_content='The Boston Celtics won the game by 20
points', metadata={}),
     Document(page_content='Larry Bird was an iconic NBA player.',
metadata={}),
     Document(page_content='Elden Ring is one of the best games in the
last 15 years.', metadata={}),
     Document(page_content='Basquetball is a great sport.', metadata=
{}),
     Document(page_content='I simply love going to the movies',
metadata={}),
     Document(page_content='Fly me to the moon is one of my favourite
songs.', metadata={}),
     Document(page_content='This is just a random text.', metadata={})]
```

```
# Reorder the documents:
# Less relevant document will be at the middle of the list and more
```

```
# relevant elements at beginning / end.
reordering = LongContextReorder()
reordered_docs = reordering.transform_documents(docs)

# Confirm that the 4 relevant documents are at beginning and end.
reordered_docs
```

```
    [Document(page_content='The Celtics are my favourite team.',
metadata={}),
     Document(page_content='The Boston Celtics won the game by 20
points', metadata={}),
     Document(page_content='Elden Ring is one of the best games in the
last 15 years.', metadata={}),
     Document(page_content='I simply love going to the movies',
metadata={}),
     Document(page_content='This is just a random text.', metadata={}),
     Document(page_content='Fly me to the moon is one of my favourite
songs.', metadata={}),
     Document(page_content='Basquetball is a great sport.', metadata=
{}),
     Document(page_content='Larry Bird was an iconic NBA player.',
metadata={}),
     Document(page_content='L. Kornet is one of the best Celtics
players.', metadata={}),
     Document(page_content='This is a document about the Boston
Celtics', metadata={})]
```

```
# We prepare and run a custom Stuff chain with reordered docs as
context.

# Override prompts
document_prompt = PromptTemplate(
    input_variables=["page_content"], template="{page_content}"
)
document_variable_name = "context"
llm = OpenAI()
stuff_prompt_override = """Given this text extracts:
-----
{context}
-----
Please answer the following question:
{query}"""
prompt = PromptTemplate(
    template=stuff_prompt_override, input_variables=["context",
```

```
"query"]
)

# Instantiate the chain
llm_chain = LLMChain(llm=llm, prompt=prompt)
chain = StuffDocumentsChain(
    llm_chain=llm_chain,
    document_prompt=document_prompt,
    document_variable_name=document_variable_name,
)
chain.run(input_documents=reordered_docs, query=query)
```