

## Twitter Thread by Weaviate • vector search engine



**Weaviate • vector search engine**

[@weaviate\\_io](#)



■ This week, we published a demo on combining [@LangChainAI](#) and [#Weaviate](#)

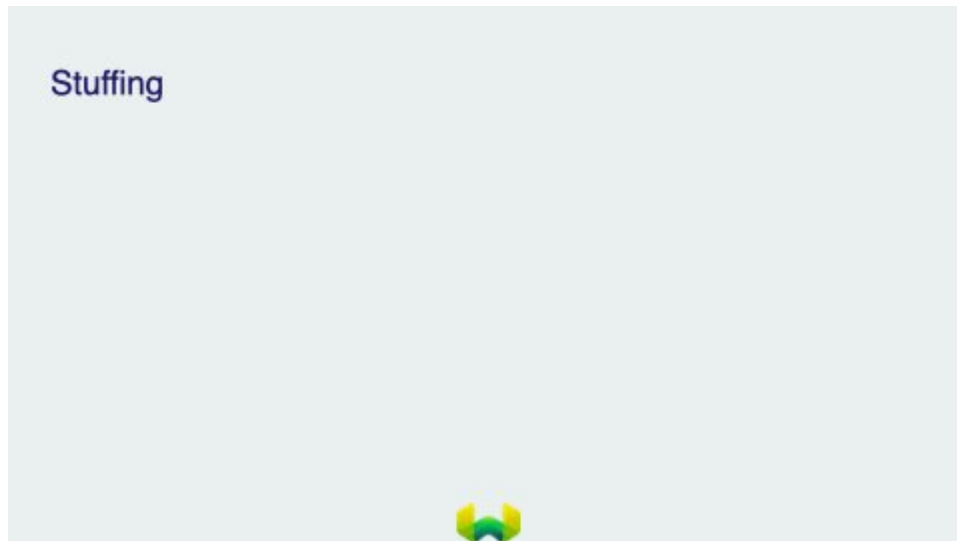
■ <https://t.co/sWAGGbvUIN>

■ Or watch [@ecardenas300](#)'s demo: <https://t.co/GVRcPhwyQU>

■ Check out the 4 CombineDocument techniques implemented in [#LangChain](#) that solve the limited input token lengths■

[@LangChainAI](#) [@ecardenas300](#) ■ Stuffing is the simplest method

■ All relevant documents from the database are stuffed into the prompt



[@LangChainAI](#) [@ecardenas300](#) ■ Map Reduce applies an initial prompt to each chunk of data, and it passed through the LLM to generate multiple responses

■ Another prompt is created to combine all of the initial outputs into one

## Map Reduce



@LangChainAI @ecardenas300 ■ Refine asks the language model to summarize the documents one by one

■ It then uses its local memory with the summaries generated so far to influence the next output

■ This is repeated until all documents have been processed

## Refine



@LangChainAI @ecardenas300 ■ Map-Rerank runs an initial prompt that asks the model to give a relevance score for each document

■ The language model then assigns a score based on the certainty of the answer and ranks them

■ The top two documents are stuffed into the model to generate a single response

Map-Rerank

