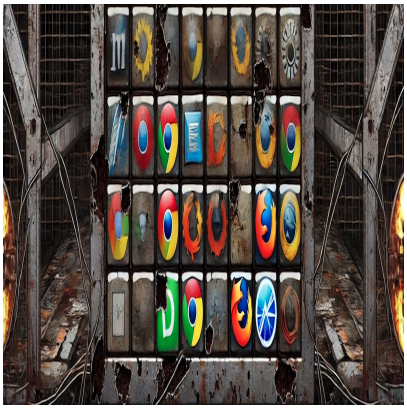

LLMs and the web

Description

Early web browsers (e.g. *Mosaic* in the 1990s and its successors) relied on centralised servers that pre-processed (â??crawledâ?•) web pages to identify key terms, words and groups of word. The servers would then add these terms to an index optimised for quick lookup, linking the terms to the URLs of the pages in which these terms appeared. As far as users were concerned, web â??crawlingâ?• operated invisibly and continuously in the background â?? traversing links to find new pages and continue the indexing.

What we browser-users thought of as â??searching the web,â?• was implemented as a process in which your local web browser would send key words from your query to this server-side indexing system that would efficiently match key words to the URL addresses of relevant web pages. The server-side search engine would then apply various filtering and ranking methods to return to the user links to web pages most relevant to the search query.

There were limits to this style of keyword search. The web page you need might not contain the exact words of your query. Whereas your browser query might be for information about â??how to make jamâ?• and may miss entirely relevant web pages that refer to â??jellyâ?• or â??marmalade.â?•

By way of contrast, modern browsers donâ??t rely on keyword matches. Search engines such as Google since 2013 use **semantic embeddings** to process queries and web content. This enables retrieval of relevant pages even if they do not contain the exact keywords from the query. I explored semantic embeddings in the context of LLMs in an earlier post. Also see a helpful online article: Vector Embeddings Explained Simply.

As well as individual words, search engines now draw on similarities between very large numerical vectors. These vectors are derived through neural network techniques to codify word meanings based on the co-occurrence of words in blocks of text. Search engines that draw on semantic embeddings match similarities between the semantic embeddings for words, phrases, sentences, paragraphs and documents stored on the web, with semantic embeddings calculated in the user query.

Combining semantic web search with LLMs

Tools such as NotebookLM pre-process each file at upload, using high dimensional semantic embeddings at the local level. That enables the LLM model to look up relevant chunks of the source documents based on semantic content rather than searching for key words and phrases. The model then drags these semantically relevant chunks of text into the context window as it responds to user prompts. Something similar happens for platforms that combine conversational AI with web search.

Certain large language models now expand the scope of their access to source content by exploiting the resources of the web, and capitalising on their use of semantic embeddings. In response to a prompt and in real time, such Retrieval-Augmented Generation (RAG) LLMs will deploy semantic web search to identify chunks of text data pertinent to the prompt and insert that into the context window, along with the conversational history of the current session and any file content uploaded by the user. With this enhanced context window, a Transformer-based LLM model then deploys its usual semantic, positional and attentional encoding processes (see post Extending large language models) to deliver responses that are hopefully up to date and cover a broad range of topics â?? in fact everything in the ever-changing information space of the WWW.

So, a RAG platform must have access to an index of the WWW based on semantic embeddings. This assumes that the LLM platform has computational and storage resources comparable to Google. Large data companies such as Google, Microsoft and their collaborators are best placed to exploit such web enhanced LLM capability. So far Microsoftâ??s Bing Chat and Googleâ??s Bard are high profile AI-Chatbot LLMs that exploit semantic web search (RAG) into their responses to user prompts.

Reference

- TiDB_Team. (2024), â??Vector Embeddings Explained Simplyâ??. *TiDB*, 16 July. Available online: https://www.pingcap.com/article/vector-embeddings-explained-simply/ (accessed 9 December 2024).

Category

1. Artificial Intelligence

Tags

1. index
2. search
3. semantic embedding
4. WWW

Date Created
December 21, 2024
Author
rcoyne99