Accepted at HyCHA 2024

hippoLLM: Scrutable and Robust Memory for LLMs with Hybrid Graph-Vector Databases

Adrian Valente `adrian.valente@ens.fr`

Large Language Models (LLMs) have shown a remarkable ability in solving natural language tasks, yet remain hindered by several limitations, including the unreliability of their specialized knowledge, usually encoded implicitly in the weights of the network but sometimes invented on the fly, and the lack of scrutability of the knowledge they acquired and the sources they rely on for specific affirmations. A notable technique that has helped limit these issues is Retrieval-Augmented Generation (RAG), a system in which source documents are first converted in vector embeddings and indexed in a vector database, and then retrieved when relevant to a certain query and added to the context of said query. Yet information in a RAG system remains diffusively encoded in raw textual format, and the amount of data embedded in a query is usually too large for a human to verify, which moreover renders generations costly.

On the other hand, knowledge graphs (KG) and ontologies like YAGO are considered a gold standard for organizing information in a structured way, allowing for easy retrieval and inference as well as compelling visualizations. The challenges of these systems lie rather on their rigidity and their difficulties to handle unforeseen types of information or complex natural queries. Thus, they seem perfectly complementary of LLMs, and ideally suited for the development of hybrid neuro-symbolic knowledge systems.

In this document, we describe a prototype of such a hybrid KG-LLM system called *hippoLLM*, to reference the hippocampus, a brain structure believed to handle symbolic manipulation and long-term storage of knowledge. hippoLLM is composed of a new type of structured knowledge database and an interface to connect it to an LLM. The database (Fig. 1) organizes information first in a *knowledge graph*, where nodes are *entities*, and edges are *facts*, which are n-ary relations (with $n \in \mathbb{N}^*$, making it technically a hypergraph). Entities are primarily represented by an identifying name and short description, and facts by a natural language sentence. This leads to the complementary representation of information: entity descriptions and facts are also represented as *vector embeddings*, obtained by passing them through an embedding model (like the BERT family of models) that converts an arbitrary text into a fixed-size vector. These embeddings are indexed and stored in a *vector database* such as ChromaDB, allowing for fast retrieval by vector similarity.

Like RAG, the system then runs in indexing and query mode (Fig. 2). In indexing mode, vast amounts of raw text are processed by an LLM (here Mistral-7B) in an automated set-up using 6 carefully chosen prompts to extract simple facts, as well as the entities they are related to, and verify whether these are already stored (using vector similarity). If a fact is already stored, we simply add a source for it, if not, it is added to the graph-vector database. In query mode, following a natural language question by a user, relevant facts are extracted from the database using a vector similarity search and added to the query context. This mechanism works similarly to RAG, except that the user can see which atomistic facts were used in answering their query, and from which sources these facts were extracted. The user can also visually explore the knowedge acquired by their system.

Finally, we note that bookkeeping operations could be run, based on LLM prompts or symbolic algorithms, to consolidate the knowledge base or organize it better (for example by proposing categories in which to organize relations, or by flagging suspicious facts). We also note that this system is completely compatible with already existing curated knowledge bases like YAGO or Wikidata, that can be directly used or completed in the described format.