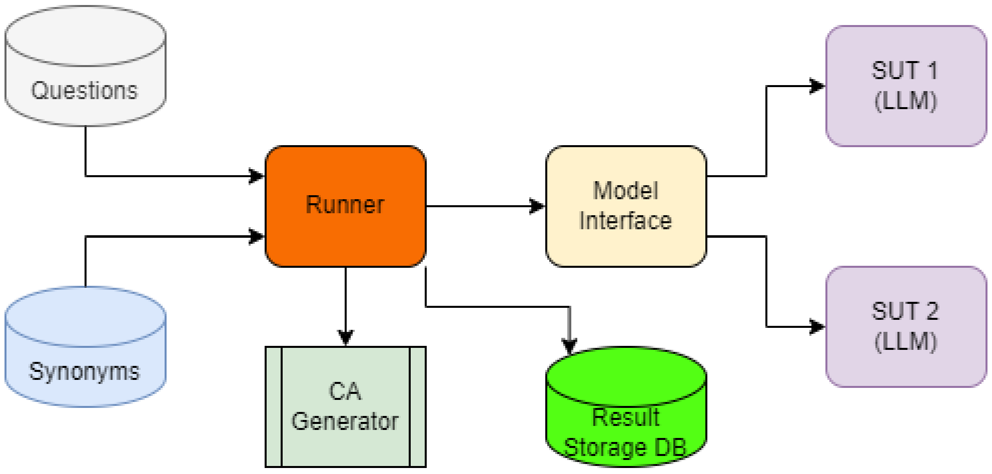
Architecture Overview

The purpose of this document is to provide an overview of the planned architecture for our Large Language Model (LLM) consistency testing framework. The diagram below visualizes the projected architecture and its components. Please note that the overall testing process (including the generation of additional inputs/queries) is not yet finalized. Minor changes to this architecture are thus possible.



We describe the role and purpose of each component below.

Runner

The purpose of the Runner is to conduct the overall LLM testing process. This will most likely include:

- Generation of queries;
- Processing of existing queries to derive their sentence structure and other natural language processing (NLP) characteristics;
- Mutation of queries to derive additional inputs used to evaluate the consistency of a LLM
    - Generation of abstract mutated queries using a CA Generator
    - Translation of abstract queries to concrete LLM inputs
- Normalization of received responses from LLMs;
- Evaluation of the LLM output's correctness;
- Storing results and evaluation data in the result storage DB;
- Creation of statistics regarding the performance of LLMs and the overall testing process.

Question Database

Commonly, the evaluation of LLMs for the ability to answer knowledge-based questions is based on publicly available data sets of questions and answers, possibly enhanced with additional tagging (e.g. linguistic information, category of question, or alternative answers).

Synonym Database

While the overall process of LLM consistency testing has not yet been finalized, we are confident that we will use a database of synonyms to create additional inputs. However, additional processing and restrictions will need to be applied in order to limit the amount of distortion our changes apply to the meaning of questions in the database.

Covering Array (CA) Generator Adapter

We plan to use Covering Arrays (CAs) for covering a well-defined, broadly distributed subset of the (very large) potential input space created by the combination of a question database and a list of synonyms. A CA is a discrete mathematical structure that offers a guaranteed degree of pseudo-exhaustive coverage of the Cartesian product of multiple sets/lists (e.g. those created by extracting the synonyms of a given word) based on a required strength. Generating CAs is computationally intensive and a wide range of algorithms and implementations exist, some of which are available for free while others are closed-source and/or commercial. We will thus create an interface that allows for this component to be exchanged if required.

Model Interface

The Runner needs a common interface to send queries to each of the LLMs and preprocess their responses. These tasks will be handled by the Model Interface, which keeps a list of connections to LLM Adapters and performs normalization of queries and responses.

System Under Test/LLM Adapter

While the diagram above shows two instances of this component, their actual number is not strictly defined. Generally, one LLM will be managed by one LLM Adapter that handles the embedding of queries in prompts and configures the LLM instance (e.g. seeds, temperature, maximum results, etc.). In some cases, the LLM instance will be started by the Adapter itself, while in other cases, an existing LLM instance will have to be connected.

Result Storage Database

Submitted queries, responses, as well as an evaluation of their correctness and consistency and statistics regarding test execution will be stored in a result storage database. We have not yet decided upon the specific database management system, but our use case does not seem to pose requirements that necessitate commercial licenses or specialized solutions.