Understanding Logical Reasoning - Task 5

(Auditors - Sameera Sudarshan Kashyap and Pavithra Selvaraj) | GitHub

Paper: Logic-LM: https://arxiv.org/abs/2305.12295

Understanding the core idea of the paper:
Prepared the slides summarizing the core idea and functionalities of the paper:

1. User provides a natural language input
2. Large Language Model like GPT is utilized to convert natural language into symbolic formulation
3. Utilize the Symbolic solver to infer the answer
4. Convert the result back to natural language answer
5. A self refiner to rectify the errors caused in the past

Understanding the paper's algorithm:
Cloned the codebase and installed all the required dependencies using requirements.txt and tested the code locally as per the commands in the readme file to ensure that everything is working as expected and to understand the workflow as stated in the paper.

Reimplementing the algorithm and testing using own KB:

We have re-implemented the core behaviour of the Logic-LM by creating a system of modular components for each phase. Given a natural language question as the input to the system, it first classifies the type of problem as either logical programming or constraint satisfaction problem. Based on the classification a separate module is executed for Logic programming or CSP - we call these modules as `constraint_solver` or `logic_solver`. Both solvers rely on iterative GPT-4.o prompting to refine and generate the intermediate symbolic representation of the problem. The solver modifies the problem and code using chain of thought prompting in the gpt-4.o model. The output of these solvers which are the symbolic representation of the problem and goal are then passed to the result interpreter module which converts the machine readable symbolic output to natural language response which is human understandable. To enhance the reasoning accuracy, we have implemented a self refinement module. If the solver module returns an "uncertain" outcome or an ambiguous result during the first iteration, the self-refinement module is triggered. It re-evaluates the output, incorporating feedback from the prior iteration to reformulate the symbolic representation and query. This loop can be repeated for multiple refinement steps, allowing the system to find a correct solution. With this we were able to mimic the main framework proposed in the Logic-LM paper.