PROMPT ENGINEERING AND TREE-OF-THOUGHTS MODEL

Student name: Kieu Ngoc Nguyen

Student ID: 103806243

OBJECTIVE: To utilize the Tree-of-Thoughts (ToT) model with a Depth-First-Search (DFS) algorithm to generate an AI-based solution to a given problem. The generation solution is then compared with a human-provided answer using the BART model to determine the similarity between two.

EXECUTION:

1. Abstract OpenAI Language Model:

* Introduced the foundational class OpenAILanguageModel, which sets the base structure for the optimized language model.
* This class incorporates methods and attributes for thought generation and state evaluation, establishing the groundwork for the model's operations.

1. Optimized OpenAI Language Model:

* Built upon the OpenAILanguageModel, this optimized model introduces caching mechanisms to prevent redundant calculations.
* It includes methods for parallel thought generation and state evaluation. These functions aim to concurrently generate and assess thoughts, optimizing the process.

1. Tree of Thoughts (ToT) Model:

* Described a structure called Tree of Thoughts (ToT) which can decompose thoughts based on problem properties.
* The model contains a thought generator function and a state evaluator function.
* It can choose between Breadth-First Search (BFS) or Depth-First Search (DFS) as search algorithms. In this implementation, DFS was chosen.

1. Optimized Tree of Thoughts:

* An optimized version of the ToT model was introduced, with a more streamlined solve method. It was specifically designed for the DFS search algorithm.
* Using the ToT with DFS, the best state (solution) for a given problem is identified.

1. Load Dataset and Extract Questions:

* The dataset from HuggingFace, named "MQuAD-v1", was loaded.
* A specific question (3rd in the dataset) and its associated ground-truth answer were extracted for evaluation.

1. Application of the ToT Model:

* The ToT model with the DFS algorithm was applied to the extracted question.
* The model attempted to generate an answer (solution) to the problem using the DFS approach within the ToT framework.
* The result was an AI-generated solution.

TESTING: BARTScore, BLEU, METEOR, ROUGE, FactCC

Using 5 Metrics to calculate the similarity:

* BART Model for Similarity Measurement:
* Using the transformers library, a pre-trained BART model (facebook/bart-large) and its associated tokenizer were loaded.
* The AI-generated solution and the human-provided answer were tokenized, encoded, and fed into the BART model to obtain contextual embeddings.
* Sentence embeddings for both solutions were computed by averaging the BART model's output.
* The cosine similarity between the embeddings of the AI solution and the human answer was computed, resulting in a similarity score.

RESULTS:

The AI solution and the human-provided solution were compared using the BART model, and a similarity score of 88% was obtained. This score signifies that the AI's generated solution closely matches the human's answer, indicating the effectiveness of the ToT model in generating accurate responses.

There is a CSV file named “results\_1.csv” which includes 5 columns: “Thoughts”, “Evaluate” (which is evaluate value for each thought)”, “Best Solution” (which is final thought), “Human Solution” (answer for given problem) and “Similarity Score” between Human and AI generation solution.

Prompt will be stored in separate file as “openai.logs”  
The best solution (final solution) is stored in file named “output.json”.