AprèsCoT: Explaining LLM Answers with Knowledge Graphs and Chain of Thought

Moein Shirdel
University of Waterloo
Waterloo, Canada
mshirdel@uwaterloo.ca

Joel Rorseth
University of Waterloo
Waterloo, Canada
jerorset@uwaterloo.ca

Parke Godfrey
York University
Toronto, Canada
godfrey@yorku.ca

Lukasz Golab
University of Waterloo
Waterloo, Canada
lgolab@uwaterloo.ca

Divesh Srivastava
AT&T Chief Data Office
New Jersey, USA
divesh@research.att.com

Jarek Szlichta
York University
Toronto, Canada
szlichta@yorku.ca

ABSTRACT

We demonstrate AprèsCoT, a post-hoc tool for understanding how large language models (LLMs) answer questions. The idea behind AprèsCoT is to map the answer and the LLM's inference steps—obtained via chain-of-thought (CoT) prompting—onto a knowledge graph to produce a structured explanation. Conference participants will choose from several LLMs and knowledge graphs to explore AprèsCoT's ability to visualize inference paths, identify potentially incorrect answers, and find knowledge graph quality issues.

1 INTRODUCTION

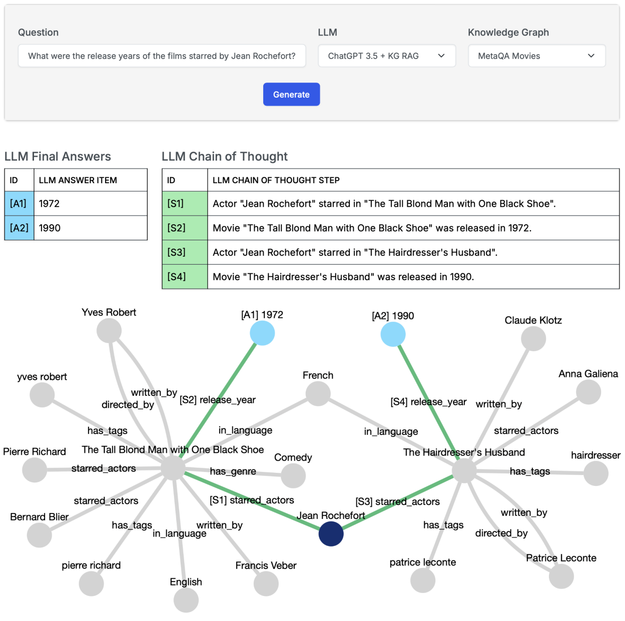
Large language models (LLMs) have revolutionized natural language processing tasks such as question-answering (QA) with their generative abilities. However, these models are prone to *hallucination*, generating confident but inaccurate outputs. This limits their adoption in high-stakes domains such as healthcare and education.

To mitigate hallucination, recent systems such as Perplexity and Microsoft Copilot employ retrieval-augmented generation (RAG), in which relevant documents are retrieved from the Web, summarized, and added to the LLM's context. The LLM can then generate an answer to a question using both its pretrained knowledge and the knowledge contained in the retrieved documents.

However, RAG does not guarantee that the LLM will use the retrieved information in its generative process. Further effort is required to trace the provenance of the generated answer, either by using another model for post-hoc citation recovery [4], which could also hallucinate, or by repeatedly querying the LLM with subsets of the retrieved sources to determine which source influences the answer [2, 5], which is expensive.

To address these issues, we demonstrate AprèsCoT, a lightweight tool for understanding LLM answers. We leverage two observations:

- that modern LLMs can generate their inference steps via chain of thought (CoT) prompting, e.g., by appending the sentence "Let's think step by step" to the query, and
- that if plausible, these steps can be mapped to facts stored in a knowledge graph (KG) to produce a data provenance trail leading to a structured explanation.



Question: What were the release years of the films starred by Jean Rochefort?
LLM: ChatGPT 3.5 + KG RAG
Knowledge Graph: MetaQA Movies

LLM Final Answers

ID	LLM ANSWER ITEM
[A1]	1972
[A2]	1990

LLM Chain of Thought

ID	LLM CHAIN OF THOUGHT STEP
[S1]	Actor "Jean Rochefort" starred in "The Tall Blond Man with One Black Shoe".
[S2]	Movie "The Tall Blond Man with One Black Shoe" was released in 1972.
[S3]	Actor "Jean Rochefort" starred in "The Hairdresser's Husband".
[S4]	Movie "The Hairdresser's Husband" was released in 1990.

Figure 1: An explanation produced by AprèsCoT.

AprèsCoT is a post-hoc tool applicable to any CoT-capable LLM, as long as a suitable KG exists or can be generated from a suitable document corpus.

Figure 1 shows AprèsCoT[1,2] in action. The user selects from the list of preloaded LLMs (ChatGPT 3.5 with RAG) and KGs (MetaQA Movies), and asks for the release years of movies starring actor Jean Rochefort. The answer (1972 and 1990) and the chain of thought produced by the LLM are displayed both in tabular form and on the underlying KG. Dark blue KG nodes correspond to the entities in the question, blue nodes correspond to answers, and green edges show the inference paths that AprèsCoT matched to the CoT. The user can then investigate any inconsistencies and gaps in the illustrated inference paths for possible hallucination or KG incompleteness instances (see Section 3 for demonstration scenarios).

We make the following contributions.

(1) **Novel LLM Explanation Approach:** On the conceptual side, we propose AprèsCoT, the first tool that leverages CoT prompting and KGs to understand LLM answers. The idea aligns with nearest neighbour LLMs [3]; i.e., looking

[1] AprèsCoT is available at http://lg-research-2.uwaterloo.ca:8050/aprescot
[2] A demo video is available at https://vimeo.com/1037583358

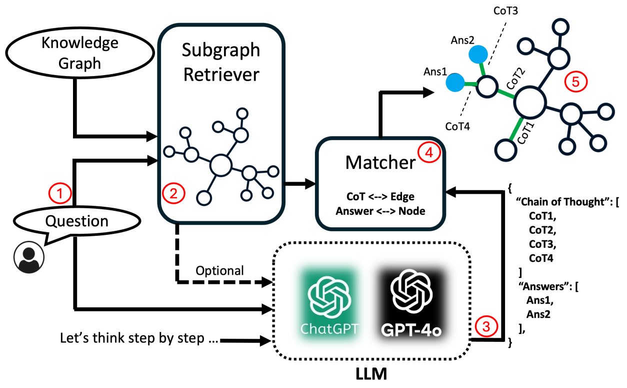
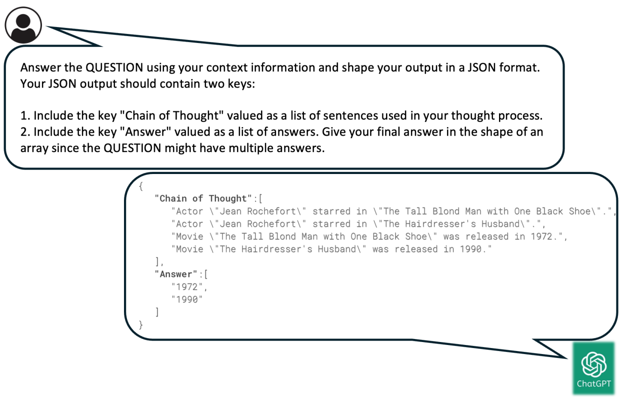


Figure 2: The architecture and workflow of AprèsCoT.



Answer the QUESTION using your context information and shape your output in a JSON format.
Your JSON output should contain two keys:

1. Include the key "Chain of Thought" valued as a list of sentences used in your thought process.
2. Include the key "Answer" valued as a list of answers. Give your final answer in the shape of an array since the QUESTION might have multiple answers.

```
{
  "Chain of Thought":[
    "Actor \"Jean Rochefort\" starred in \"The Tall Blond Man with One Black Shoe\".",
    "Actor \"Jean Rochefort\" starred in \"The Hairdresser's Husband\".",
    "Movie \"The Tall Blond Man with One Black Shoe\" was released in 1972.",
    "Movie \"The Hairdresser's Husband\" was released in 1990."
  ],
  "Answer":[
    "1972",
    "1990"
  ]
}
```

Figure 3: Example of format prompting used by AprèsCoT.

up the nearest documents to the output generated by the LLM as a form of citation. However, our use of CoT to visualize structured inference paths over a KG is novel.

(2) **Alignment Mechanism:** On the technical side, the main challenge faced by AprèsCoT is to align the CoT and the LLM's answer to the nodes and edges in the corresponding KG. To address this challenge, we convert KG elements to text and match sentences generated by the CoT prompt to facts stored in the KG using pre-trained text embedding models.

(3) **Insights:** On the application side, we present use cases on the MetaQA cinema KG and the Unified Medical Language System (UMLS) KG using ChatGPT 3.5 and GPT-4o-Mini (with and without RAG). These use cases demonstrate the value of AprèsCoT for LLM answer verification, KG completeness analysis, and performance comparisons among LLMs.

2 SYSTEM DESCRIPTION

2.1 Overview

We designed AprèsCoT to be a lightweight post-hoc tool for understanding how LLMs answer questions, relying only on API access to the LLM. The two assumptions in our design are as follows:

- that the LLM supports CoT prompting[3], and
- that a KG exists (or can be built from a suitable document corpus using existing tools [1]) with information relevant to the domain of the questions. For example, recall the MetaQA cinema KG in Figure 1, with entities such as actors, movies and their attributes, and relationships between entities such as starred or directed-by.

The current version of AprèsCoT comes pre-loaded with API access to ChatGPT 3.5 and GPT-4o-Mini, and cinema and medical KGs.

Figure 2 shows the design of AprèsCoT. The input consists of an LLM whose answers are to be analyzed, a corresponding KG, and a question (step 1). In step 2, AprèsCoT runs a subgraph retriever to identify KG facts that may be relevant to the question. Next, AprèsCoT queries the LLM. We also support an optional RAG mode, where the facts identified by the subgraph retriever are given to the LLM as context alongside the CoT prompt and formatting instructions. This allows users to compare question-answering performance of LLMs with and without RAG. In step 3, AprèsCoT parses the LLM output, i.e., the answer(s) to the question and the CoT, and runs a matcher module (step 4) that aligns these outputs to nodes and edges in the subgraph retrieved in step 2. Finally, the inference paths are illustrated on this subgraph (step 5).

The remainder of this section describes the input (prompting and subgraph retrieval) and output (matching CoT and answer to this subgraph) processing details of AprèsCoT.

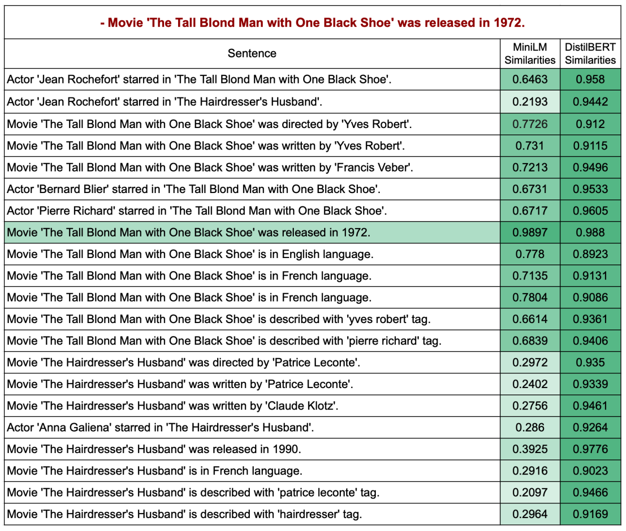
[3]In the next version of AprèsCoT, we will explore LLM explanations using tree-of-thought and graph-of-thought prompting. Furthermore, we acknowledge that CoT prompting can influence the LLM's inference process, potentially leading to a different answer compared to simple prompting.

2.2 Input Processing

Subgraph Retriever. The purpose of this module is to find a subgraph of the KG that is relevant to the question being asked and convert each fact (edge) in the subgraph to a sentence. This is important for two reasons. First is the efficiency of matching the CoT with KG facts done in step 4 in Figure 2. This matching could in principle be done over the entire KG, but most of the KG is likely to be irrelevant to the query. The matching step is much faster when limited to the relevant subgraph. The second purpose of the subgraph retriever is to enable the optional RAG mode for the LLM being explained, in which the subgraph (in sentence format) is given to the LLM as context.

Existing systems for question answering over KGs also identify relevant facts in KGs, and so AprèsCoT's subgraph retriever is based on previous work [1]. First, we run a Named Entity Recognition (NER) model over the question to identify entities mentioned therein. Next, we locate the nodes in the KG corresponding to these entities and we explore the neighbourhoods of these entities, adding nearby nodes and edges to the retrieved subgraph based on their relevance to the question.

Prompting. AprèsCoT instructs the LLM by performing two prompting steps: CoT prompting, and JSON format prompting for the answer and the CoT. This module also prompts the LLM with the question asked by the user and the optional contextual knowledge (when RAG is enabled). We show an example prompt and LLM output in Figure 3, corresponding to the demonstration scenario from Figure 1. This example uses RAG mode, in which AprèsCoT adds facts found by the subgraph retriever to the LLM context and instructs the LLM to answer the question using this information.



- Movie 'The Tall Blond Man with One Black Shoe' was released in 1972.

Sentence	MiniLM Similarities	DistilBERT Similarities
Actor 'Jean Rochefort' starred in 'The Tall Blond Man with One Black Shoe'.	0.6463	0.958
Actor 'Jean Rochefort' starred in 'The Hairdresser's Husband'.	0.2193	0.9442
Movie 'The Tall Blond Man with One Black Shoe' was directed by 'Yves Robert'.	0.7726	0.912
Movie 'The Tall Blond Man with One Black Shoe' was written by 'Yves Robert'.	0.731	0.9115
Movie 'The Tall Blond Man with One Black Shoe' was written by 'Francis Veber'.	0.7213	0.9496
Actor 'Bernard Blier' starred in 'The Tall Blond Man with One Black Shoe'.	0.6731	0.9533
Actor 'Pierre Richard' starred in 'The Tall Blond Man with One Black Shoe'.	0.6717	0.9605
Movie 'The Tall Blond Man with One Black Shoe' was released in 1972.	0.9897	0.988
Movie 'The Tall Blond Man with One Black Shoe' is in English language.	0.778	0.8923
Movie 'The Tall Blond Man with One Black Shoe' is in French language.	0.7135	0.9131
Movie 'The Tall Blond Man with One Black Shoe' is in French language.	0.7804	0.9086
Movie 'The Tall Blond Man with One Black Shoe' is described with 'yves robert' tag.	0.6614	0.9361
Movie 'The Tall Blond Man with One Black Shoe' is described with 'pierre richard' tag.	0.6839	0.9406
Movie 'The Hairdresser's Husband' was directed by 'Patrice Leconte'.	0.2972	0.935
Movie 'The Hairdresser's Husband' was written by 'Patrice Leconte'.	0.2402	0.9339
Movie 'The Hairdresser's Husband' was written by 'Claude Klotz'.	0.2756	0.9461
Actor 'Anna Galiena' starred in 'The Hairdresser's Husband'.	0.286	0.9264
Movie 'The Hairdresser's Husband' was released in 1990.	0.3925	0.9776
Movie 'The Hairdresser's Husband' is in French language.	0.2916	0.9023
Movie 'The Hairdresser's Husband' is described with 'patrice leconte' tag.	0.2097	0.9466
Movie 'The Hairdresser's Husband' is described with 'hairdresser' tag.	0.2964	0.9169

Figure 4: Comparison of MiniLM & DistilBERT embeddings.

2.3 Output Matching

This module consumes the JSON-formatted output of the LLM, containing the answer(s) and the CoT. It matches the answer(s) to the most similar node label(s), and each CoT statement to the most similar edge, in the subgraph described in Section 2.2.

We perform the matching by computing contextual word embeddings for the answer(s), the CoT, and the nodes and edges in the subgraph. Here, two design decisions are:

- which embedding model to use, and
- when calculating cosine similarity between potential matches, what similarity threshold to use when declaring a match.

The embedding model we selected is all-MiniLM-L6-v2[4], a transformer-based sentence encoder with 384-dimensional embeddings. Our model selection methodology was based on inference time (we prefer smaller models suitable for an interactive system) and effectiveness, the latter assessed by manually inspecting cosine similarities of similar and dissimilar facts.

As an example, Figure 4 compares all-MiniLM with another sentence encoder, DistilBERT, when matching the nearest facts to the statement at the top: "The Tall Blond Man with One Black Shoe was released in 1972". Each row in the table shows the cosine similarity between this statement and a fact in the KG for both models. While both models identify the same top match with the highest similarity (the sentence highlighted in green), DistilBERT produces similar embeddings for all the facts shown in the table, leading to similar cosine similarities. On the other hand, all-MiniLM generates more diverse embeddings, reflected in the wider range of similarity scores in the table, reducing the possibility of suboptimal matches having high similarity.

Finally, the similarity threshold we selected is 0.7, determined via grid search on a curated dataset of valid matches and non-matching sentences. That is, the matcher module returns the top match if its similarity is above the threshold and no match otherwise. As we will see in Section 3, failed matches correspond to gaps in inference paths, due to model hallucination or KG incompleteness.

[4]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

3 DEMONSTRATION DESCRIPTION

In our demonstration of AprèsCoT, conference participants will ask cinema and medical questions and analyze LLM answers over the aforementioned MetaQA and UMLS KGs. Furthermore, participants will be able to compare the outputs produced by several LLMs, both in standalone and RAG mode.

In the remainder of this section, we describe three use cases that will serve as our demonstration starting points, to illustrate consistent inference, incomplete inference, and KG incompleteness.

3.1 Consistent Answers and CoT

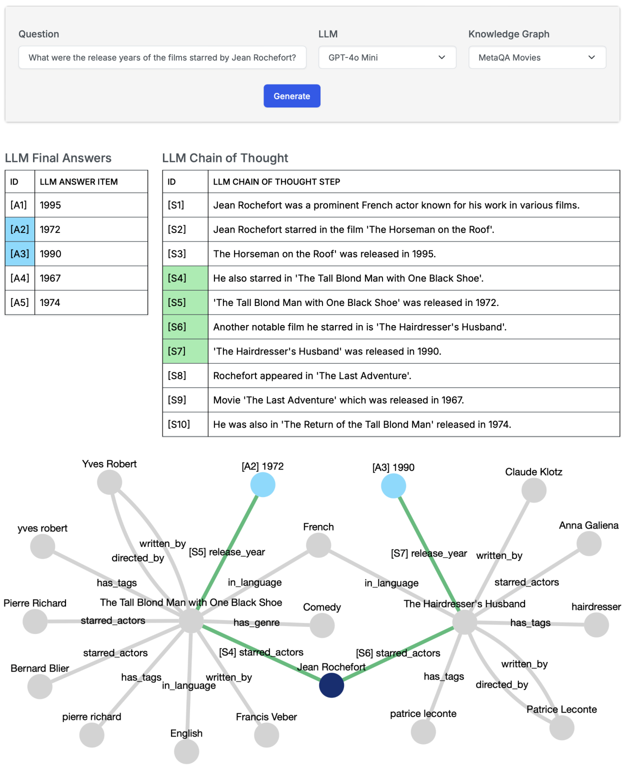
The first use case starts with the example in Figure 1, where the user selects MetaQA and ChatGPT3.5 with RAG, and asks "What were the release years of the films starred by Jean Rochefort?" The LLM produces two answers, 1972 and 1990. To verify the answers, participants can trace the two inference paths, namely [S1]-[S2] and [S3]-[S4], from the source entity, "Jean Rochefort," to the highlighted answer nodes, labelled [A1] and [A2]. Here, the answers and the chain of thought are consistent with the KG.

Next, participants can switch to GPT-4o-Mini with RAG as the backbone model, which generates the same answers. Additionally, participants can test both LLMs in standalone mode, where the models rely solely on their pretrained knowledge, without KG facts in their context, to answer the query. The next use case (Section 3.2) builds on this scenario, showing how the explanations produced by AprèsCoT can identify potential instances of incompleteness in the knowledge graph.

3.2 KG Data Quality

In this use case, we utilize AprèsCoT to identify potential KG data quality issues. The user poses the same question as in Figure 1: "What were the release years of the films starred by Jean Rochefort?" However, unlike the previous scenario, the backbone model is GPT-4o-Mini operating in standalone mode, without any contextual knowledge. The model generates an answer based on its pretrained knowledge and provides inference steps via CoT prompting. The user selects MetaQA KG to ground the LLM's answers, as the question pertains to movies.

As illustrated in Figure 5, GPT-4o-Mini generates five release years, [A1] through [A5], and ten CoT inference steps, [S1] through [S10], which is more than in the previous use case in Section 3.1. However, AprèsCoT can match only the two answers that were also produced in the previous use case along with their inference steps. The additional answers (1995, 1967, and 1974) as well as the additional inference steps (all but [S4] through [S7]) do not correspond to any KG facts. The user can inspect the unmatched CoT steps, starting with [S1], which is a general statement about the actor referenced in the question. More notably, the user realizes that [S2] and [S3] reference a movie starring Jean Rochefort, released in 1995, which does not exist in the KG. Similarly, [S8] through [S10], suggest two additional movies, released in 1967 and 1974, respectively. This inconsistency may prompt the user to investigate further (e.g., do a web search to find a complete set of Rochefort's movies). If there are no such movies released in 1995, 1967 or 1974, these additional answers may be attributed to LLM hallucination. Conversely, if these additional movies exist, they reveal KG incompleteness. In this example, it turns out that the KG is incomplete and there are in fact more movies starring Rochefort.


Question: What were the release years of the films starred by Jean Rochefort?

LLM: GPT-4o Mini

Knowledge Graph: MetaQA Movies

Generate

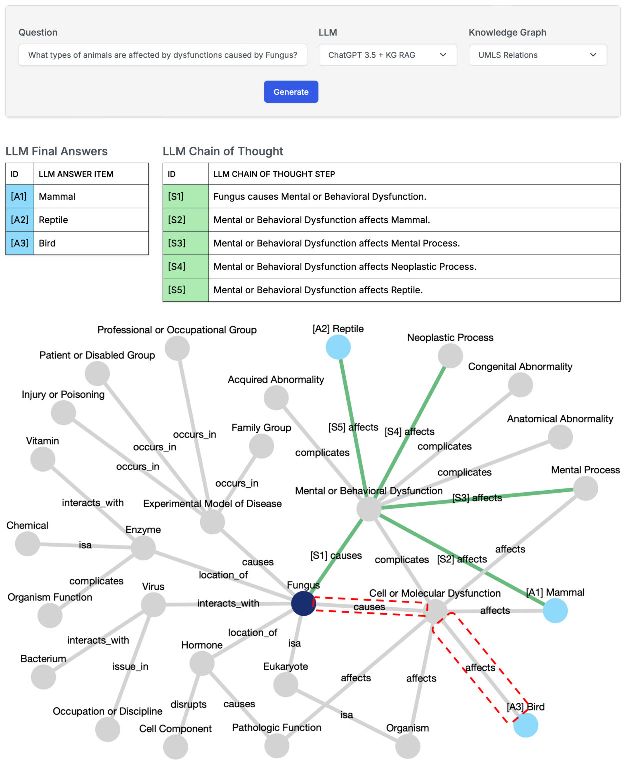
LLM Final Answers

ID	LLM ANSWER ITEM
[A1]	1995
[A2]	1972
[A3]	1990
[A4]	1967
[A5]	1974

LLM Chain of Thought

ID	LLM CHAIN OF THOUGHT STEP
[S1]	Jean Rochefort was a prominent French actor known for his work in various films.
[S2]	Jean Rochefort starred in the film 'The Horseman on the Roof'.
[S3]	The Horseman on the Roof' was released in 1995.
[S4]	He also starred in 'The Tall Blond Man with One Black Shoe'.
[S5]	'The Tall Blond Man with One Black Shoe' was released in 1972.
[S6]	Another notable film he starred in is 'The Hairdresser's Husband'.
[S7]	'The Hairdresser's Husband' was released in 1990.
[S8]	Rochefort appeared in 'The Last Adventure'.
[S9]	Movie 'The Last Adventure' which was released in 1967.
[S10]	He was also in 'The Return of the Tall Blond Man' released in 1974.

Figure 5: An explanation produced in the second use case.


Question: What types of animals are affected by dysfunctions caused by Fungus?

LLM: ChatGPT 3.5 + KG RAG

Knowledge Graph: UMLS Relations

Generate

LLM Final Answers

ID	LLM ANSWER ITEM
[A1]	Mammal
[A2]	Reptile
[A3]	Bird

LLM Chain of Thought

ID	LLM CHAIN OF THOUGHT STEP
[S1]	Fungus causes Mental or Behavioral Dysfunction.
[S2]	Mental or Behavioral Dysfunction affects Mammal.
[S3]	Mental or Behavioral Dysfunction affects Mental Process.
[S4]	Mental or Behavioral Dysfunction affects Neoplastic Process.
[S5]	Mental or Behavioral Dysfunction affects Reptile.

Figure 6: An explanation produced in the third use case.

For comparison, conference participants can repeat this scenario using ChatGPT 3.5, which also outputs an additional movie starring Rochefort, but misses some of the answers and CoT steps generated by GPT-4o-Mini. Interestingly, some of the CoT steps generated by ChatGPT 3.5 refer to general instructions for answering this question rather than KG facts, such as "Identify films starred by Jean Rochefort and check the years of each of them".

3.3 Inconsistent Answers and CoT

Finally, we turn to the UMLS medical KG to illustrate LLM inconsistencies. As shown in Figure 6, the user asks "What types of animals are affected by dysfunctions caused by Fungus?" and selects ChatGPT 3.5 with RAG. The LLM responds with three answers, [A1] through [A3], and five inference steps, [S1] through [S5], all matched with an element in the UMLS subgraph retrieved by AprèsCoT. The user can trace inference paths from the source entity, Fungus, to the first two answers, Mammal [A1] and Reptile [A2]. These paths align with CoT steps [S1], [S2] and [S5], for partial consistency between the answers and the inference process.

However, no path connects the source entity, Fungus, to the final answer, Bird [A3], highlighting an inference gap illustrated with red dashed boxes. Additionally, the user observes that CoT steps [S3] and [S4] do not correspond to any valid answers, as they lead to non-animal entities outside the scope of the query, despite being included in the LLM's inference process.

The user can also switch to the other LLM, GPT-4o-Mini with RAG, and observe that while the answers are unchanged, the inference steps are different. In this case, the inference steps align with all the final answers, filling the gaps in the graph. The path leading to the last answer (Bird) is explicitly included in the CoT, and the inference steps leading to invalid answers ([S3] and [S4] in Figure 6) do not appear in the CoT. This example again shows AprèsCoT's value in understanding the inference differences between LLMs.

REFERENCES

[1] Andrew Chai, Alireza Vezvaei, Lukasz Golab, Mehdi Kargar, Divesh Srivastava, Jaroslaw Szlichta, and Morteza Zihayat. 2023. EAGER: Explainable Question Answering Using Knowledge Graphs. In *Proceedings of the 6th Joint Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA)*. ACM, 4:1–4:5. https://doi.org/10.1145/3594778.3594877

[2] Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. 2024. ContextCite: Attributing Model Generation to Context. *CoRR* abs/2409.00729 (2024). https://doi.org/10.48550/ARXIV.2409.00729

[3] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through Memorization: Nearest Neighbor Language Models. In *8th International Conference on Learning Representations, ICLR*. https://openreview.net/forum?id=HklBjCEKvH

[4] Weitao Li, Junkai Li, Weizhi Ma, and Yang Liu. 2024. Citation-Enhanced Generation for LLM-based Chatbots. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*. 1451–1466. https://doi.org/10.18653/V1/2024.ACL-LONG.79

[5] Joel Rorseth, Parke Godfrey, Lukasz Golab, Divesh Srivastava, and Jaroslaw Szlichta. 2024. RAGE Against the Machine: Retrieval-Augmented LLM Explanations. In *40th IEEE International Conference on Data Engineering, ICDE*. IEEE, 5469–5472. https://doi.org/10.1109/ICDE60146.2024.00430