Extended Abstract:
A Reliable Common-Sense Reasoning Socialbot Built Using LLMs and Goal-Directed ASP

Yankai Zeng[1], Abhiramon Rajashekharan[1], Kinjal Basu[2], Huaduo Wang[1], Joaquín Arias[3] and Gopal Gupta[1]

[1]*The University of Texas at Dallas, Richardson, USA*
[2]*IBM Research*
[3]*CETINIA, Universidad Ray Juan Carlos, Madrid, Spain*

Abstract

This is an extended abstract of [1] Y. Zeng, A. Rajashekharan, K. Basu, H. Wang, J. Arias, G. Gupta, A Reliable Common-Sense Reasoning Socialbot Built Using LLMs and Goal-Directed ASP, which is accepted at ICLP 2024.

Keywords

Answer Set Programming, s(CASP), Large Language Model, Socialbot

Conversational agents are designed to understand dialogues and generate meaningful responses to communicate with humans. After the popularity of ChatGPT, with its surprising performance and powerful conversational ability, commercial *Large Language Models* (LLMs) for general NLP tasks such as GPT-4 [2], sprung up and brought the generative AI as a solution to the public view. These LLMs work quite well in content generation tasks, but their deficiency in knowledge-and-reasoning-oriented tasks is well-established by now [3]. These models themselves cannot tell whether the text they generate is based on facts or made-up stories, and they cannot always follow the given data and rules strictly and sometimes even modify the data at will, also called *hallucination*. The reasoning that these LLMs appear to perform is also at a very shallow level. These are serious flaws that make the LLMs unsuitable for fact-based conversations such as providing correct information to a user. The reason for this flaw is that LLMs generate text that is purely based on a pattern-matching mechanism, and consequently have no understanding of the meaning of sentences thus a safe and sound way of reasoning is not guaranteed [3]. In contrast, humans understand the meaning of sentences and then use their reasoning capabilities to check for consistency and take further action. Thus, to make the machine-generated response reliable and consistent, our socialbot needs to follow a similar approach.

Following the above insights, in this work, we report on developing an elaborate conversational socialbot that can understand human dialog and respond properly according to human expectations. The goal of this socialbot is to be able to engage in multiple rounds of social conversations with users about their interests, understand the context of what the user is saying like a human, and be able to initiate new topics or make relevant recommendations at the right time to control the pace of the chat. We currently focus our topics of interest on discussions of movies, books, and related people, and use the STAR framework [4] to build a social conversational bot called AutoCompanion. AutoCompanion uses an LLM to interact with the user and translate between natural language dialogues and knowledge represented in predicates. It further uses s(CASP)[5], an ASP system to reason over this knowledge. After the user's input is parsed into predicates by the LLM, s(CASP) uses reasoning (and a bit of randomness) to decide whether to continue with the current topic of conversation, move on to a related topic, or recommend books or movies that are currently on sale based on the user's preferences. Questions asked by the user are also answered. Subsequently, the instructions for the next step from the reasoner are passed on to another LLM in the form of an open-ended free-for-all on that topic, and the generated sentence is provided to the user as a reply.

We believe that the misuse of LLM in fact-based domains should be controlled to avoid and that the

4th Workshop on Goal-directed Execution of Answer Set Programs (GDE'24), October 12, 2024

best way to utilize LLM is to use it only as an interface for parsing and presenting knowledge. Our AutoCompanion outperforms LLM-only socialbots in reliability, scalability, controllability, creativity, and many other aspects.

Vocabulary (Ontology) Design AutoCompanion leverages LLMs such as the GPT series to parse the natural language sentences to knowledge. The prompts are given to LLMs including the knowledge ontology, which is represented in the predicates used in the reasoning part, and several examples covering the use cases. The LLM-produced predicates are formed in themes, where a theme is a set of predicates centralized on one property of a topic instance. It typically contains one talk predicate: "talk(Topic, Instance_name, Property).", one attitude predicate: "attitude(positive/negative/ask/argue).", and one content predicate: "content(Property, Detailed_content).". Here the topic is defined either to be a movie, a book, or related people, and the property indicates an aspect of the topic instance to be potentially discussed, such as the plots, scenes, lines of a movie, etc. If the user asks a question, the predicate question: "question(What_is_asked)" is also captured. For instance, when the user compliments the scene of the movie Titanic where the giant ship is sinking, the predicates extracted from the user would be:

```
talk(movie, ‘Titanic’, scene). content(scene, sinking). attitude(positive).
```

The user preference is also extracted to be used later for recommendation. The preference is presented by the prefer predicate: "prefer(Topic, Property, Preference).".

Topic Control We leverage the s(CASP) ASP reasoner to guide and switch between the topics using CKT [6] and RCC [7]. Once a theme is selected, the ASP reasoner chooses whether to remain in the same topic instance or switch to a relevant one. When the ASP reasoner continues the current topic instance, CKT is used. It maintains a state with a list of properties that can be discussed for each topic instance. Once the reasoner chooses to go deeper into one instance using other properties, the CKT will be triggered to pick up one property that has not been discussed yet for this topic instance. The CKT also chooses whether to remain on the current property and the attitude towards the current property. The CKT component chooses what property to discuss under a topic. The code for CKT is shown below:

```
next_attr(Topic, Name, Attr) :- continue_attr, talk(Topic, Name, Attr),
                                response_attr(Topic, Attr).
next_attr(Topic, Name, Attr) :- response_attr(Topic, Attr),
                                not neg_next_attr(Topic, Name, Attr).
neg_next_attr(Topic, Name, Attr) :- not next_attr(Topic, Name, Attr).
neg_next_attr(Topic, Name, Attr) :- hist(I, Topic, Name, Attr, Attitude, From).
```

The first rule indicates to continue the current property if continue_attr, which is evoked by a random function, is true. The second and third rule forms an even loop, making it possible to choose any available property. The fourth rule defines that the discussed property should not be selected.

The RCC selector will be invoked if the ASP reasoner switches to a relevant topic instance. In AutoCompanion, an RCC is a new topic instance that shares some common properties (e.g., genre, actor) with the current instance. These relations are formatted and listed for the reasoner to search the RCCs automatically. The topic switching is implemented by the codes below.

```
next_topic(Topic, Name, Attr, 'None', 'None') :-
        round(I), hist(I, Topic, Name, Attr, Attitude, 'user'),
        not discussed_before(Topic, Name, Attr).
next_topic(Topic, Name, Attr, 'None', 'None') :-
        continue_topic, talk(Topic, Name, Attr1),
        next_attr(Topic, Name, Attr).
next_topic(Topic, Name, Attr, Source, Relation) :-
        len_rcc(I), random(1, I, N), rcc(N, Topic, Name, Source, Relation),
        next_attr(Topic, Name, Attr).
```

Response Generation AutoCompanion leaves the generation of these specifics to LLM, controlling only the topic and property for each round. We assume that the LLMs are experts in common topics such as famous movies and books, which makes it possible for us to leverage this advantage of LLMs in constructing AutoCompanion. The knowledge-required answer is generated with the prompt template as below:

```
What are the most interesting [property] for [topic] [instance]? [attitude] ->
```

And LLM would generate an answer for that. For example, If we ask about the positive value expressed in Batman Begin:

```
What are the most interesting value expressed for Batman Begins? positive ->
```

The LLM would generate:

```
I feel like Batman Begins nails it with the whole inner battle vibe.  It's not
just about taking down bad guys; it's Bruce Wayne wrestling with his own demons, and
that's what makes it hit home. Do you have the same feeling as me?
```

If the RCC mechanism is invoked, the response generator will also create cohesive sentences that naturally mention the next topic instance to discuss. The template for a cohesive sentence is:

```
Because you mentioned [source_topic_instance], it makes me think of the [next_topic]
[next_topic_instance], since [relation_of_them].
```

Similarly, if the reasoner answers the questions from the user, one of the following templates will be used according to the condition:

```
I remembered that the [property] of the [topic] [topic_instance] is [answer].
Sorry I could not remember the [property] of the [topic] [topic_instance].
```

The reply sentences generated by the above templates are then concatenated and processed by an LLM language modifier to make the replies more coherent, gentler, and more informal.

Knowledge Base Construction The movie data is from the IMDb dataset,[1] where we collected the information of release year, runtime, rating, countries involved, languages, genres, top 10 casts with their characters, directors, writers, editors, composers, producers, cinematographers, and the plot summary. The book data is collected from the Users-Books-Dataset,[2] including 528 books with information about their series, author, rating, language, genres, awards, setting places, characters, and plot descriptions. These are also recorded as properties along with their writing style, symbolism, emotional impact, and social background. We also collect the information of the people involved in these movies and books from the IMDb dataset.

Conclusion This work presents an approach to building a socialbot that utilizes LLM to translate between natural language and predicates while employing an ASP reasoner as the backend. We used this framework to develop the AutoCompanion socialbot to chat about movies and book-related topics. AutoCompanion parses the sentences to different themes regarding books and movies using an LLM and controls the next theme to respond by s(CASP), the ASP reasoner. We believe this organic combination of LLM and ASP is one of the most effective ways to realize the control of LLM-based socialbots.

References

[1] Y. Zeng, A. Rajashekharan, K. Basu, H. Wang, J. Arias, G. Gupta, A reliable common-sense reasoning socialbot built using llms and goal-directed asp, arXiv preprint arXiv:2407.18498 (2024).

[2] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al., Gpt-4 technical report, 2023.

[3] M. Nezhurina, L. Cipolina-Kun, M. Cherti, J. Jitsev, Alice in wonderland: Simple tasks showing complete reasoning breakdown in state-of-the-art large language models, 2024. URL: https://arxiv.org/abs/2302.03494.

[4] A. Rajasekharan, Y. Zeng, P. Padalkar, G. Gupta, Reliable natural language understanding with large language models and answer set programming, 2023.

[1] IMDb dataset is available at https://developer.imdb.com/non-commercial-datasets and https://cinemagoer.github.io.

[2] Users-Books-Dataset is available at https://data.world/divyanshj/users-books-dataset.

[5] J. Arias, M. Carro, E. Salazar, K. Marple, G. Gupta, Constraint answer set programming without grounding, 2018. URL: https://arxiv.org/abs/1804.11162. doi:10.48550/ARXIV.1804.11162.

[6] K. Basu, H. Wang, N. Dominguez, X. Li, F. Li, S. C. Varanasi, G. Gupta, CASPR: a commonsense reasoning-based conversational socialbot, 2021.

[7] F. Li, H. Wang, K. Basu, E. Salazar, G. Gupta, Discasp: A graph-based ASP system for finding relevant consistent concepts with applications to conversational socialbots, in: Proc 37th ICLP (Tech. Comm), volume 345 of *EPTCS*, 2021, pp. 205–218.