Building Intelligent Systems by Combining Machine Learning and Automated Commonsense Reasoning*

Gopal Gupta[1], **Yankai Zeng**[1], **Abhiraman Rajasekaran**[1], **Parth Padalkar**[1], **Keegan Kimbrell**[1], **Kinjal Basu**[2], **Farahad Shakerin**[1], **Elmer Salazar**[1], **Joaquín Arias**[3]
[1]Computer Science Department, The University of Texas at Dallas, Richardson, USA
[2]IBM Research, USA
[3]Universidad Rey Juan Carlos, Madrid, Spain
gupta@utdallas.edu

Abstract

We present an approach to building systems that emulate human-like intelligence. Our approach uses machine learning technology (including generative AI systems) to extract knowledge from pictures, text, etc., and represents it as (pre-defined) predicates. Next, we use the s(CASP) automated commonsense reasoning system to check the consistency of this extracted knowledge and reason over it in a manner very similar to how a human would do it. We have used our approach for building systems for visual question answering, task-specific chatbots that can "understand" human dialogs and interactively talk to them, and autonomous driving systems that rely on commonsense reasoning. Essentially, our approach emulates how humans process knowledge where they use sensing and pattern recognition to gain knowledge (Kahneman's System 1 thinking, akin to using a machine learning model), and then use reasoning to draw conclusions, generate response, or take actions (Kahneman's System 2 thinking, akin to automated reasoning).

Introduction

I don't see that human intelligence is something that humans can never understand.

– John McCarthy, March 1989

Words are not in themselves carriers of meaning, but merely pointers to shared understanding.

– David Waltz

The long-term goal of AI research is to build systems that are as good as humans. Consider natural language understanding (NLU), for example. NLU is a challenging task since there are multiple skills that humans employ to understand a typical sentence. First, a person needs to interpret a sentence and understand its meaning. Second, they need to be able to interpret the meaning of the sentence in the current context, using the commonsense knowledge they already possess. This helps resolve ambiguities in the sentence and assess if any information is missing. Third, if required, they should be able to pose a question that would seek to fill in any information that is missing. Finally, once they attain a complete understanding of the sentence, they should be able to explain what they understood. We believe that all of these skills are important for an NLU system that seeks to reliably answer questions or hold a conversation with a human.

Likewise, answering questions about a given picture (Visual Question Answering or VQA) is another task that has been undertaken in AI. Humans generally first recognize the objects in the picture, then they reason with the questions asked about the picture using their commonsense knowledge. An effective automated VQA system should work in a similar way. Thus, to "perceive" a picture, ideally, a system should have intuitive abilities like object and attribute recognition and understanding of spatial-relationships. To answer questions, it must use reasoning. Natural language questions are complex and ambiguous by nature, and also require commonsense knowledge for their interpretation. Most importantly, reasoning skills such as counting, inference, comparison, etc., are needed to answer these questions.

We strongly believe that to build intelligent systems that are as good as humans, we need to approximate the process that humans follow. When humans view a picture, view a scene, or read a text, they *extract* the information in that picture, scene, or text. This information then resides in their mind in some abstract form as knowledge. Similarly, knowledge is extracted using other senses (taste, smell, touch). Humans then interpret this extracted knowledge in the context of commonsense knowledge that they have accumulated over the years that also resides in their mind. They might use the commonsense knowledge to conclude, for example, that the knowledge extracted is inconsistent, or perhaps incomplete. The extracted knowledge, in conjunction with the commonsense knowledge, is also used to draw new conclusions using deduction, abduction, or even induction (Gupta 2022).

To build AI systems, we follow an approach similar to what humans employ. We use machine learning technology to extract knowledge from pictures, scenes, text, etc. We represent this knowledge as pre-defined logical *predicates*. Commonsense knowledge is represented using answer set programming (Gelfond and Kahl 2014). New conclusions can be drawn from the extracted knowledge together with the commonsense knowledge. A question can be converted into a query that can then be posed against the extracted knowledge combined with commonsense knowledge. Knowledge can be extracted from pictures as well as

*Authors partially supported by NSF grant IIS 1910131.

text using generative AI tools such as GPT-3 (Brown et al. 2020) and GPT-4 (OpenAI 2023).

In the past, we have used machine learning tools such as YOLO (Redmon et al. 2016) to label a picture with predicates describing the objects in the picture, their attributes, their relative spatial placement, etc. These extracted predicates can then be used in conjunction with commonsense knowledge represented in answer set programming (Gelfond and Kahl 2014) to answer questions about the picture (Basu, Shakerin, and Gupta 2020) or for making (autonomous) driving decisions based on scene analysis (Kothawade et al. 2021). Reasoning involving predicates and commonsense knowledge is performed through the use of answer set programming (Gelfond and Kahl 2014).

Likewise, we use large language models (LLMs) such as GPT-3 (Brown et al. 2020) to extract knowledge from text and represent it using pre-defined predicates (akin to an ontology). In conjunction with commonsense knowledge, these predicates can be reasoned over to check their consistency, to find missing information, draw new conclusion, or to advance a conversation. The reasoning with extracted predicates and commonsense knowledge is performed using answer set programming (Gelfond and Kahl 2014). In particular, we use the s(CASP) goal-directed ASP system to perform this reasoning (Arias et al. 2018). We have used this framework to build task-specific chatbots that can "understand" human dialogs and that can verbally interact with humans to achieve a specific goal (Zeng et al. 2023).

Our framework has many advantages. First, since it relies on reasoning over extracted predicates, every decision can be explained. Second, consistency of information implicit in the extracted predicates can be checked. Finally, if the commonsense knowledge being used is correct, we can guarantee that any conclusion drawn is also correct.

In the rest of the paper we give the relevant background, and then present one applications of our framework—a chatbot that can "understand" dialogs and converse with a human user to achieve a goal. Other applications of the framework have been developed that include the AQuA visual question answering system (Basu, Shakerin, and Gupta 2020) and the AUTO-DISCERN autonomous driving system (Kothawade et al. 2021), however, these are not described here.

Background

Large Language Models: Until recently, transformer-based deep learning models have been applied to NLP tasks by training and fine-tuning them on task-specific datasets (Casola, Lauriola, and Lavelli 2022). With the advent of Large Language Models, the paradigm changed to teaching a language model any arbitrary task using just a few demonstrations, called *in-context learning*. Brown et al. (Brown et al. 2020) introduced an LLM called GPT-3 containing approximately 175 billion parameters that have been trained using a massive corpus of filtered online text, on which the well-known ChatGPT system is based. The model was able to perform competitively on several tasks such as question-answering, semantic parsing (Shin and Van Durme 2022), and machine translation. However, such LLMs tend to make simple mistakes in tasks such as semantic (commonsense) and mathematical reasoning (Floridi and Chiriatti 2020; Wei et al. 2022).

In our work, we use GPT-3 for semantic parsing and leave the reasoning part to answer set programming based commonsense reasoning systems such as s(CASP). We theorize that given the vast pre-training they go through, LLMs can be used to automatically extract knowledge inherent in the text, just like humans do. Our experiments confirm that LLMs are able to extract such knowledge as predicates from sentences—with high accuracy—after learning from a few example demonstrations. Thus, our experiments show that LLMs are able to extract, what linguists call, the *deep structure* of a sentence, given a sentence's *surface structure*.

Answer Set Programming and the s(CASP) system: The s(CASP) system (developed by Arias et al.(Arias et al. 2018)) is an answer set programming (Gelfond and Kahl 2014) system that supports predicates, constraints over non-ground variables, uninterpreted functions, and, most importantly, a top-down, query-driven execution strategy. These features allow returning answers with non-ground variables (possibly including constraints among them) and compute partial models by returning only the fragment of a stable model that is necessary to support the answer to a given query. The s(CASP) system supports constructive negation based on a disequality constraint solver and unlike Prolog's negation as failure and ASP's default negation, `not p(X)` can return bindings for `X` on success, i.e., bindings for which the call `p(X)` would have failed. Additionally, s(CASP) system's interface with a constraint solver (over reals) allows for sound non-monotonic reasoning with constraints.

Complex commonsense knowledge can be represented in ASP and the s(CASP) query-driven predicate ASP system can be used for querying it (Gupta 2022; Gelfond and Kahl 2014). Commonsense knowledge can be emulated using (i) default rules, (ii) integrity constraints, and (iii) multiple possible worlds (Gelfond and Kahl 2014; Gupta 2022). Default rules are used for jumping to a conclusion in the absence of exceptions, e.g., a bird normally flies unless it's a penguin. Default rules with such exceptions represent an elaboration-tolerant way of representing knowledge (Gelfond and Kahl 2014).

```
flies(X) :- bird(X),
            not abnormal_bird(X).
abnormal_bird(X) :- penguin(X).
```

Integrity constraints allow us to express impossible situations and invariants. For example, a person cannot sit and stand at the same time.

```
false :- person(X), sit(X), stand(X).
```

Finally, multiple possible worlds allow us to construct alternative universes that may have some of the parts common but other parts inconsistent. For example, the cartoon world of children's books has a lot in common with the real world (e.g., birds can fly in both worlds), yet in the former birds can talk like humans but in the latter they cannot.

Default rules are used to model a bulk of our commonsense knowledge. Integrity constraints help in checking the consistency of the information extracted. Multiple possible worlds allow us to perform assumption-based reasoning (for example, knowing that “Alice loves Bob”, we could assume that either Bob also loves Alice or he does not).

Several commonsense reasoning applications have already been developed using ASP and the s(CASP) system (Morris 2023; Chen et al. 2016). In application reported in this paper, we have kept the commonsense reasoning component simple, as our main goal is to illustrate our framework for combining LLMs and ASP to develop NLU applications that are explainable and reliable. Because of the use of ASP, it is also possible to detect inconsistencies or biases in the text by reasoning over the predicates extracted. Justification for each response can also be given, as the s(CASP) system can generate justifications as proof trees (Arias et al. 2020).

Concierge Bot: A Goal-Directed Conversational Bot

In this section we describe the conversational bot that we have developed using our framework. We first leverage LLMs to extract implicit and explicit knowledge predicates from a user’s dialogue and then reason over the extracted predicates using s(CASP) to satisfy the user’s request. To illustrate the applicability of our method to conversational AI, we narrow the domain of the bot to a very specific problem: restaurant recommendations. This ensures that commonsense knowledge that needs to be modeled is within a limited domain that can be expanded later as necessary. We call this bot the concierge bot. Since the conversational bot is an interactive application, the commonsense reasoning involved can be complex, since the reasoner has to keep track of the state of the conversation. In order to make a restaurant recommendation, the bot needs to ask the user for information (preferred cuisine, price range, etc.) and recommend a restaurant once all information is procured. At any stage, the bot can explain why it’s asking for a given piece of information as well as the reason for its recommendation. The benefits of our framework are more apparent in this application. The bot can ask user questions to acquire missing information or to verify if any errors were made during predicate generation. We show an example transcript of an interaction session below of the concierge bot with a human user. The English responses are generated using GPT-3 again after learning from predicate-text demonstrations (essentially, using the reverse of the text-to-predicate generation process).

```
Bot:   Hi, what can I assist you with?
Human: Can you help me find a place for
       food with curry? Not a pricey one.
Bot:   Are you looking for a place with a
       specific customer rating?
Human: No, I don’t mind the rating.
Bot:   What kind of restaurant would you
       like to visit?
Human: A normal restaurant.
Bot:   Would you prefer somewhere
       family-friendly for tonight?
Human: No. Just for myself.
Bot:   Unfortunately, we cannot provide the
       results to your request.
Human: How about one with a high price?
       But it should be then at least
       above average quality.
Bot:   Are you looking for a Thai place?
       The Rice Boat may be suitable but it
       can be pricey and customer feedback
       has been mixed. Additionally, it may
       not have the best atmosphere for
       families.
Human: Yes, that’s what I need! Tell me
       where it is.
Bot:   The Rice Boat, located on 901 Bangkok
       Road, has an average customer rating
       and offers Thai cuisine at a premium
       cost. Unfortunately, it is unsuitable
       for children.
Human: Great! Thank you for the service!
Bot:   It’s no problem, I’m happy to assist.
```

Predicate Generation Using LLMs

We adapt examples from the E2E dataset released by Novikova et al. (Novikova, Dušek, and Rieser 2017) for our use case. We employ in-context learning where we give the LLM examples of pairs containing a natural language sentence and the corresponding restaurant predicates. We use the following six predicates from the dataset: **restaurant-name**, **eatType** (as establishment type), **food** (as cuisine), **priceRange**, **customerRating**, **familyFriendly**.

We use in-context learning where we provide the GPT-3 model with 11 selected examples from the dataset, which covers all the predicates along with their possible arguments. This ensures that the LLM is aware of every possible predicate as well as every possible argument value these predicates can take. To assess the viability of LLMs for the predicate generation task, we tested the model using the first 500 examples in the E2E training set and obtained an accuracy of 89.33%. The accuracy metric we use is designed to account for the generation of correct predicates as well as arguments. The high predicate generation accuracy supports the feasibility of using our framework for the concierge bot. Our framework can similarly be applied, to build any robust domain-specific conversational bots such as a front desk office receptionist or an airline reservation assistant.

Concierge Bot System Construction

To make GPT-3 better understand the meaning of each predicate, we first change the predicate names in E2E as follows: **restaurnt-name**, **typeToEat**, **cuisine**, **priceRange**, **customerRating**, **familyFriendly**. We also add two predicates **address** and **phoneNumber** to record the location and contact information for the user’s query. An external predicate **prefer** is also added to capture the user’s preference (such as curry, spicy, etc.) The information asked by the user is expressed by the value “query”. We specialized GPT-3 with about a dozen example sentences along with the corresponding predicate(s). Below we show some examples of the sentences and the predicates generated after this specialization.

```
Sentence: Fitzbillies coffee shop provides
    a kid-friendly venue for Chinese food
```

```
    at an average price point in the
    riverside area. It is highly rated
    by customers.
Predicates: restaurant-name(Fitzbillies),
    typeToEat('coffee shop'),
    cuisine(Chinese), priceRange(moderate),
    customerRating(high), familyFriendly(yes)

Sentence: Can you find a place for food at
    a low price? Both English and French
    cuisine is fine for me.
Predicates: restaurant-name(query),
    cuisine([Engish, French]),
    priceRange(cheap)
```

Commonsense knowledge involved in making a restaurant recommendation is coded using s(CASP). The interactive bot will take in the user's response and convert it to predicates using GPT-3. The predicates become part of the state. At this stage, we check for user preference. For example, if the user wants curry, Indian and Thai cuisine would be automatically added to the state through appropriate rules. The bot then examines the state to assess if all the information needed is present so that it can make a recommendation and if not, it will generate a question to ask the user for that information. This logic, shown in Figure 1, can be thought of as a state machine and has been referred to as a conversational knowledge template (CKT) by Basu et al. (Basu et al. 2021). The concierge bot determines which predicates are missing in its state to make a recommendation. One of the missing predicates is then selected and a query is created using it. Note that we use GPT-3 again to generate natural-sounding text from the predicate(s) corresponding to the response that our bot computes. The users can also change their preferences during the conversation and our bot can handle that. Take the conversation mentioned above as an example. When the user asks "Can you help me find a place for food with curry? I don't want a pricey one.", following predicates are generated by the GPT-3 text-davinci-003 model: *"restaurant-name(query), prefer(curry), priceRange([cheap, moderate])"*. The predicates are then added to the memory of the bot, where the log of the user requirements of the current conversation is stored. Note that the predicates *"prefer(curry)"* and *"cuisine([indian,thai])"* are also added to the state using the specific commonsense rule we wrote for this purpose (not shown due to lack of space). After the state gets updated, the bot checks to see whether any other key information is missing using another commonsense rule (not shown also due to lack of space). If no information is missing, the bot comes to the recommendation step where it searches for the appropriate restaurant in its database, consistent with the user's input, and outputs it as a recommendation.

Since we use GPT-3 to generate predicates from user utterances, the user has complete freedom in terms of what they say. GPT-3 is able to extract the appropriate predicates from the corresponding deep form of the sentence. GPT-3 is also used to generate the natural language response of the bot. As a result, the interaction appears quite natural, as the example above shows.

Experiments

To explore GPT-3's ability to generate goal-directed conversation without the reasoning component that we use, we tested the conversation directly on GPT-3 with the sample conversations as input. We showed GPT-3 three full paragraphs of conversations and asked GPT-3 to generate a similar conversation according to the user input. The database of restaurants used by our concierge bot is also included in the input to the GPT-3 model. An example interactive conversation directly with GPT-3 is as follows:

```
Bot:   Hello, what can I do for you today?
Human: I'd like to have a family meal with
       my children.
Bot:   What type of cuisine would you like?
Human: Mexican food is suitable for me.
Bot:   What is your desired price range
       for the place?
Human: At a higher price, please.
Bot:   I recommend The Wrestlers, a coffee
       shop serving Mexican cuisine at a
       moderate price range. It is
       family-friendly and has an average
       customer rating. The address is
       470 Main Rd.
```

The responses given by GPT-3 in the above conversation are correct except for the price range. In the given database, the restaurant recommended only serves cheap food. Hence, GPT-3 modified the information to align with the user's request. GPT-3 also follows the given examples and asks about the cuisine and price, but does not request other information like our framework does. This is because these questions are not motivated by missing information, unlike in our approach. This example shows that although GPT-3 used on its own as a conversational bot is able to generate natural-sounding sentences fluently, it is unreliable and does not understand the knowledge given. Bots developed using our framework do not face such problems because they employ explicit commonsense reasoning. The methodology we use to build the concierge bot is explained in more detail in the paper Zeng et. al. (Zeng et al. 2023).

Conclusion and Future Work

In this extended abstract, we presented a framework for developing AI systems that attempt to emulate humans. Knowledge is extracted from a picture, scene, or text using machine learning technology, including generative AI systems. This process can be thought of as Kahneman's System 1 thinking (Kahneman 2013). The extracted knowledge is represented as (pre-defined) logical predicates. Subsequently, reasoning is performed over these predicates conjoined with commonsense knowledge using answer set programming, specifically, using our s(CASP) system. This process is akin to Kahneman's System 2 thinking. While this framework is just an approximation of how human thinking works, it allows us to build many interesting applications, for example, the task-specific chatbot described above.

One disadvantage of our approach is that the predicates used to represent the extracted knowledge have to be predefined. Also, commonsense knowledge related to these

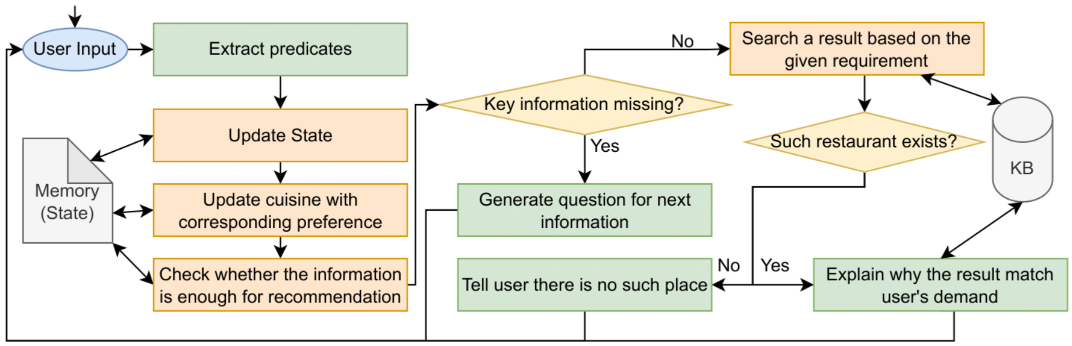

Figure 1: The framework of the reasoning system in Concierge Bot. The green boxes indicate the steps done by LLMs and the orange ones indicate the steps done by s(CASP).

predicates has to be explicitly modeled. The requirement to define commonsense knowledge impacts scalability. Future work thus includes: (i) using pre-defined ontologies such as WordNet and VerbNet as pre-defined set of predicates, (ii) generating commonsense knowledge automatically, perhaps using resources such as WordNet, VerbNet, etc.

References

Arias, J.; Carro, M.; Chen, Z.; and Gupta, G. 2020. Justifications for Goal-Directed Constraint Answer Set Programming. In *Proceedings 36th ICLP (Tech. Comm.)*, volume 325 of *EPTCS*, 59–72.

Arias, J.; Carro, M.; Salazar, E.; Marple, K.; and Gupta, G. 2018. Constraint Answer Set Programming without Grounding. *TPLP*, 18(3-4): 337–354.

Basu, K.; Shakerin, F.; and Gupta, G. 2020. AQuA: ASP-Based Visual Question Answering. In *Proc. PADL*, 57–72. Cham: Springer.

Basu, K.; Wang, H.; Dominguez, N.; Li, X.; Li, F.; Varanasi, S. C.; and Gupta, G. 2021. CASPR: a commonsense reasoning-based conversational Socialbot. ArXiv:2110.05387.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, 1877–1901. Curran Associates, Inc.

Casola, S.; Lauriola, I.; and Lavelli, A. 2022. Pre-trained Transformers: An Empirical Comparison. *Machine Learning with Applications*, 9: 100334.

Chen, Z.; Marple, K.; Salazar, E.; Gupta, G.; and Tamil, L. 2016. A Physician Advisory System for Chronic Heart Failure management based on knowledge patterns. *Theory and Practice of Logic Programming*, 16(5-6): 604–618.

Floridi, L.; and Chiriatti, M. 2020. GPT-3: Its Nature, Scope, Limits, and Consequences. *Minds and Machines*, 30: 1–14.

Gelfond, M.; and Kahl, Y. 2014. *Knowledge representation, reasoning, and the design of intelligent agents: Answer Set Programming approach*. Cambridge Univ. Press.

Gupta, G. 2022. Automating Common Sense Reasoning with ASP and s(CASP). Technical Report, https://utdallas.edu/~gupta/csr-scasp.pdf.

Kahneman, D. 2013. *Thinking Fast and Slow*. Farrar, Strauss, and Giroux.

Kothawade, S.; Khandelwal, V.; Basu, K.; Wang, H.; and Gupta, G. 2021. AUTO-DISCERN: Autonomous Driving Using Common Sense Reasoning. In *Proc. GDE Workshop*, volume 2970. CEUR-WS.org.

Morris, J. 2023. Blawx: User-friendly Goal-Directed Answer Set Programming for Rules as Code. In *Proc. Prog. Lang. and the Law (ProLaLa)*.

Novikova, J.; Dušek, O.; and Rieser, V. 2017. The E2E dataset: New Challenges for End-to-end Generation. In *Proc. SIGDIAL Discourse and Dialogue*, 201–206.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Redmon, J.; Divvala, S.; Girshick, R.; and Farhadi, A. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788.

Shin, R.; and Van Durme, B. 2022. Few-Shot Semantic Parsing with Language Models Trained on Code. In *Proc. ACL-HLT*, 5417–5425.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E. H.; Le, Q. V.; and Zhou, D. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. In *NeurIPS*.

Zeng, Y.; Rajasekharan, A.; Padalkar, P.; Basu, K.; Arias, J.; and Gupta, G. 2023. Automated Interactive Domain-Specific Conversational Agents that Understand Human Dialogs. Forthcoming.