CSC 244/444/2023 Machine Reasoning

http://www.cs.rochester.edu/~schubert/444/

Len Schubert, WEGMANS HALL 3003

Introduction

Consider the following little reasoning challenge:

"In the recent election of a student council president at Brighton High School, the turnout was 100% -- every Brighton student voted for some candidate. Clyde and Diane were two of the Brighton students who were candidates in the election. (Our own kids at Brighton, Alice and Bob, were not candidates.) Sadly, every candidate in the election disparaged every other candidate. Of course, no-one votes for someone they disparage." *As briefly as you can, answer these questions: Did any of Clyde, Diane, Alice, and Bob vote for themselves? If so, who, and what is your reasoning?*

Can you answer those questions after a little thought, without reading further?

The correct response would be that Clyde and Diane each voted for themselves, since everyone voted, but not for fellow-candidates, whom they disparaged. If you got this right, you've demonstrated reasoning ability. Here's another chance to test yourself:

"Suppose I have 3 cubical blocks, A, B, and C, and currently B and C are directly on the table, and A is on C. How can I build a stack where C is on top, B is in the middle, and A is at the bottom, by moving one block at a time?"

Make yourself a diagram, if you like (people often think with the aid of mental images or by drawing pictures). The problem stems from the 1970s, and is called the "Sussman anomaly" after the MIT grad student who showed that early planning algorithms had trouble with it. You'll probably find it trivial to solve – it just takes 3 moves. Again, this little problem draws on your reasoning ability. These are examples of the kinds of reasoning that any genuinely intelligent machine surely should be able to handle, and which are the focus of this course.

Current status in machine reasoning

You are probably acquainted with large language models (LLMs) such as ChatGPT, LLaMa, or PaLM, which have appeared in recent years. These represent dramatic strides in natural language understanding and dialogue, and have utterly captured the public imagination and even percolated into daily news and political discourse. The "deep neural network" (DNN) technology that underlies LLMs also has played key roles in other AI successes such as triumphs over humans in chess and Go, face recognition, image captioning, prediction of the convoluted 3-D structure of proteins from their amino acid sequence, machine translation, generation of dazzling images from text prompts, accelerated drug discovery, robot control, automated driving, and much else. LLMs and DNNs more generally have also ensnared the energies of AI researchers to such an extent that very few are paying attention to reasoning problems like those above, let alone the "classical" symbolic techniques that were traditionally employed for them. They assume that ways of enhancing LLMs to solve such problems will soon be found. After all, LLMs can already engage in philosophical discussions, summarize lengthy articles, explain photosynthesis, guide you in cooking catfish Southern-style, write sonnets in the style of Shakespeare, and so much more.

But while LLMs indeed "think" impressively by analogical abstraction from vast oceans of data, they are extremely unreliable when it comes to ***combinatorial*** or ***logistical*** reasoning – the kind of reasoning where the solution is the result of making choices in a space of *interrelated alternatives*, and is subject to *multiple constraints* imposed by the nature of the domain and of the desired solution. *ChatGPT failed on both of the simple problems above!* On the first, it concluded that none of Alice, Bob, Clyde, and Diane voted for themselves. When reminded by the user that the candidates only disparaged *other* candidates, not themselves, ChatGPT declared it had too little information to answer the question. For the block-stacking problem, ChatGPT offered the following three-move "solution":

1. Move block A from on top of C to the table.
2. Move block B from the table to on top of block C.
3. Move block A from the table to on top of block B.

This gives a stack in the reverse order of the specified one. This is very likely due to ChatGPT's "reinforcement learning from user inputs", a strategy that the developers hoped would improve its performance. No doubt other users had posed the "Sussman anomaly" to ChatGPT -- but in the usual wording, A and C are interchanged in the problem statement from what was stated above. So ChatGPT's attempt to mimic past solutions backfired. When the number of blocks in such block stacking problems is increased to 4 or 5, ChatGPT almost invariably flubs them.

Much of the public commentary on LLMs, including by some undeniably brilliant tech leaders and academics, expresses worries about imminent superintelligent machines that might sweep humanity aside in pursuit of their own purposes. In expressing such worries, they are forgetting about the vast logistical superiority over machines that their own work, and that of their organizations, has demonstrated. They have achieved exploits in industry, space, and scientific R&D that is comparable in complexity to the building of the Panama Canal, the Golden Gate Bridge, or the atom bomb. Do they really think they are threatened by machines that can't sort out who voted for whom in the "student president" puzzle, or figure out how to rearrange a few toy blocks?

In the case of some of these prognosticators, their expectation of an imminent "Singularity" (run-away machine intelligence) is based on the combinatorial reasoning shown by game-playing programs such as AlphaZero (for full-information games like chess and Go) or Pluribus (for poker). But reasoning in such radically limited worlds, where the possible situations and possible moves are easily formalized and coded, is just nothing like real-world reasoning. For example, successfully creating and leading a space technology company requires solution of not only extremely complex technical problems, but also political, financial, legal, staffing, and infrastructure problems, and all of these are tightly interconnected. This interconnectedness is also why a human-like logistical reasoner cannot be built by providing an LLM with "plug-ins" for various specialized forms of reasoning – Wolfram|Alpha for math, SimScale for engineering simulation, Vampire 4.8 for logic problems, IMPLAN for economic analytics, etc. While any logistically intelligent AI system will certainly need access to such tools, if it is going to solve large-scale, real-world problems, it will not be able to plan the use of the tools, or integrate their results into a coherent overall approach to the targeted problems, if it is not *itself* highly competent in logistical reasoning.

The frustrating issue with using LLMs as a basis for reasoning systems is that vast amounts of knowledge about technology, politics, finance, laws, and human and organizational characteristics are indeed stored in them – but in an opaque, numerical form. This "black box" knowledge isn't easily brought to the surface in a form enabling reasoning – and to the extent that an LLM can be coaxed to reason step-by-step, stating the assumptions being used, the reasoning

just isn't precise enough to solve even simple logistical problems like those above. My guess is that to enable LLMs to reason, we need to enable them to

- convert problem statements from natural language into a structured, self-contained form to which principles of sound reasoning can be applied;
- maintain a set of options for advancing the reasoning process (a kind of working memory);
- retrieve relevant pieces of knowledge that might help with the reasoning (the retrieval might just require self-queries of an appropriate sort, or maybe an entirely separate factual memory, to be filled by learning or knowledge mining);
- combine retrieved knowledge items with those in working memory in a logically sensible way, deriving new conclusions or conjectures; and
- evaluate progress towards the goals of the reasoning.

This may be a good area for formulating a CSC444 project.

Previous attempts to push DNNs towards thinking have usually consisted of setting up multiple-choice questions, to be answered from short texts. For example, the so-called Winograd Schema Challenge involves examples like the following: In the sentence variants *"The vase didn't fit into the suitcase because it was too small/too big"*, what does *"it"* refer to? Or in the sentence variants *"The city councillors refused the women a parade permit because they feared/advocated violence"*, who does *"they"* refer to? (Terry Winograd's own example, circa 1970!). The site http://www.thelowdownblog.com/2016/07/turing-tests-indicate-that-if-we-want.html provides a commentary. LLMs now solve these problems quite well.

More generally, DNNs do very well on multiple-choice questions, but I think such AI "challenges" are misguided. The problem is that just because such questions intuitively involve commonsense thinking doesn't mean they ***require*** any thinking when tackled by DNN methods. That's because DNN methods train on many thousands of test examples (often after pretraining to predict the most likely next word in millions or billions of texts), and in doing so learn to combine numerous superficial "cues" (features of words and the way they are arranged) to pick out the most likely choices. This has been called the "Clever Hans" phenomenon, after the horse that seemed to be able to do simple additions by stamping its leg, when in fact it was picking up superficial bodily cues from its unsuspecting trainer. Mind you, DNNs do seem to acquire some form of abstract knowledge in their higher layers, but their failures on simple combinatorial problems like those we saw above, and their inability to ***explain*** their "thinking", even when trained on virtually everything "out there" on the web, indicates that they don't actually think or understand. One thing is for sure: Learning by mimicry of vast collections of texts is nothing like human learning; Humans can pick up novel information and reason with it based on very little input (you are doing it right now), and this ability is almost entirely lacking in deep learning systems. An excellent, in-depth study of the strengths and weaknesses of LLMs from a cognitive science perspective is, (see https://arxiv.org/pdf/2301.06627.pdf):

> Kyle Mahowald, Anna A. Ivanova, Idan A. Blank, Nancy Kanwisher, Joshua B. Tenenbaum, and Evelina Fedorenko, "Dissociating language and thought in large language models: A cognitive perspective", arXiv:2301.06627v1 [cs.CL], Jan. 16, 2023.

Combinatory and logistical reasoning: The key challenge

Machine Reasoning (also traditionally called "Knowledge Representation and Reasoning (KR&R)" or "Logical Foundations of AI") is primarily concerned with enabling machines to think -- the emphasis is on combinatorial reasoning and planning, and how machines can internally represent the knowledge needed to automate these distinctively human activities, and to acquire such knowledge. (However, we'll not have much time for delving into knowledge acquisition. This

long-standing problem, referred to as the *knowledge acquisition (KA) bottleneck*, is currently being tackled by knowledge “mining” from LLMs; this could be another good topic for a CSC444 project.)

Quite a lot has been accomplished in automating combinatorial logical thinking over the 50 years or so since the beginnings of AI. We have some pretty good ideas about how to represent knowledge and how to reason with it (in various ways), and there are many noteworthy successes and applications. These include goal-directed planning for robots, dialogue-based problem solving systems in transportation domains, rule-based (or "description logic"-based) systems for many applications in industry, and proofs of difficult theorems in mathematics (or about the correctness of VLSI chip designs, nuclear plant shut-down protocols, or operating systems). Also, “logic programming languages” like Prolog, allowing solution of problems by formalizing the required knowledge and goals as logical formulas, have been widely used (and continue to be used) by many researchers.

Our approach to representing and using knowledge begins with "logic". Most people associate the word "logic" with a precise syntax and totally reliable deductive reasoning methods, but while we will start with that notion, we should understand that there are really two quite separate, equally important aspects of a logical symbolism: its "aboutness" and its use.

"Aboutness" refers to the fact that symbols in logic can be viewed formally as corresponding to entities, properties, and relationships in the world; in other words, they can be viewed as being ***about*** the world. As a result, logical "sentences" (formulas) can be viewed as ***representing*** factual information, events and actions in the world (or false claims, depending on whether they are ***true*** or ***false***). We'll come back to this below, in mentioning ***denotational semantics***. If we choose the symbols of our logic to be close to those of ordinary language, we can also understand intuitively what our logical sentences are saying about the world. (In fact, logical languages are just a formalization of ordinary language.) This intuitiveness is important both in our efforts to impart useful knowledge to machines, and in validating ("curating") that knowledge, and any conclusions drawn by the machine, in relation to the world.

The second aspect, "use", concerns how you ***reason*** with the represented knowledge. Certainly deduction is an important mode of reasoning, but many modes of human reasoning we'd like to emulate in computers are more like "educated guesswork". Often, such guesswork is based on regularities observed in the world. As children, when we encounter some creatures referred to as "dogs", and find that they are furry, friendly, and capable of running, barking, etc., we form the idea that dogs in general have these characteristics. This leads to more or less reliable "predictions" when we encounter another dog. Similarly, when we've gained some experience in what's involved in eating at a restaurant, we'll know not only what to expect but also what's appropriate to do when we, or others, go to a restaurant. Besides learning characteristics of things and behavioral patterns from experience, we also learn them from being told and by reading. This indicates that there's an intimate connection between language and thought, at least at an abstract level.

What is reasoning? It involves search through a space of alternatives, where individual steps are a kind of pattern transduction. For example, suppose we've learned the following general rule: If everything with property P also has property Q, and a certain entity X has property P, then conclude that X has property Q. Then how do we apply this rule to Snoopy, given that all dogs are furry and Snoopy is a dog? First, we match the claim that all dogs are furry to the pattern that all P are Q; this instantiates P to "dog" and Q to "furry". Then we match the claim that Snoopy is a dog to the pattern that X is a P, which succeeds because P is already bound to "dog". Finally, in the transduction step, we generate the conclusion that Snoopy is furry, by instantiating the

pattern saying that X has property Q, with the bindings we obtained from the "premise" matches. In our search, we keep track of alternative options, and for each step, we try to decide whether we are nearing our reasoning goal. Other types of reasoning steps are less reliable. For example, if a person X is a nonsmoker, we may well guess that a person Y whom we don't know, but who is friends with X, is also a nonsmoker. Planning, too, can be understood in terms of pattern matching/transduction processes, either instantiating familiar patterns of action for achieving given goals, or synthesizing new plans by manipulating symbolic models of how various kinds of actions change the world.

So these are the kinds of *mechanisms* we will study. But, coming back to "aboutness", we also need to understand what sorts of mechanisms lead to ***true***, or at least ***plausible***, conclusions from given premises, or lead to plans that actually work. We want our AI systems to be sound thinkers! Denotational semantics enables this kind of analysis. In our study of knowledge representation and reasoning, we will separately study denotational semantics -- what the symbols and expressions of a symbolic language can refer to or mean, and the reasoning methods that can be employed. We will see that reasoning methods can be *justified* using denotational semantics.

The future

When will machines be able to think as well as humans? And once they reach this point, won't they quickly surpass us, bringing about the Singularity (or "intelligence explosion" -- I.J. Good, 1965; Vernor Vinge's essay 1993)? My view is that this could happen within 2 or three decades, if the right questions are pursued. But expert views differ greatly, as do opinions about how wonderful or disastrous this would be. See

https://futureoflife.org/background/benefits-risks-of-artificial-intelligence/

and the related links at my 244/444 website,

http://www.cs.rochester.edu/~schubert/444/.

My own greatest worry is that civilization will collapse before the achievement of human-level AI, as a result of climate change, population growth, pandemics, and nuclear bombs as well as new means of exterminating ourselves. Can we still save ourselves? I tend to agree with Daniel Schmachtenberger that saving ourselves will require help from AI systems that can make sense of vastly more information than humans:

https://www.youtube.com/watch?v=8XCXvzQdcug

(Warning: a very lengthy video, in sophisticated sociological language). I also agree with novelist Richard Powers that we are *"pitched in a final footrace... between inventiveness and built-in insanity".*

Course goals & logistics

The main goal in Machine Reasoning is to learn about techniques for representing factual knowledge (typically, the sort of knowledge that we can easily express in ordinary language, though also some kinds of knowledge calling for specialized representations, such as 3-D entity structure and motion, and temporal & taxonomic relations), and using such knowledge for reasoning (deductive or uncertain), and for planning courses of action to achieve goals. Students will thereby gain an appreciation of the fundamental role that knowledge representation and reasoning (KR&R) play in abstract thought.

Practically speaking, students will reach the point where they can write programs for basic KR&R tasks, and make sense of the current research literature in this area; CSC 444 students will also

explore a subtopic of their choice in greater depth, by reviewing some recent papers or books and (preferably) pursuing some ideas they have.

The grade for the course will be based on homeworks (usually) assigned every other week, some Lisp programming assignments, a midterm and final test, and, for 444 students, their research/essay project. More details are on the tentative course schedule. For grading details, see the TA homepage for 244/444 (when those details become available).

Text (required for grads, but now online at https://www.cin.ufpe.br/~mtcfa/files/in1122/Knowledge%20Representation%20and%20Reasoning.pdf): Brachman & Levesque, *Knowledge Representation and Reasoning* (Elsevier/ Morgan Kaufmann 2004). Another interesting textbook, oriented towards Prolog, is David L. Poole and Alan K. Mackworth, *Artificial Intelligence 3E: Foundations of Computational Agents*, Cambridge U. Press 2023. Online at https://artint.info/3e/html/ArtInt3e.html. In my view, a lacuna in both books is a lack of acknowledgement and development of the close links between natural language and logic. In a sense, the lack of integration of language and logic has limited the application of logical methods to real-world problems in the past.

Old text (occasional technical reference): Genesereth & Nilsson, *Logical Foundations of AI*. Supplementary: Russell & Norvig, *Artificial Intelligence*, 2020 edition (used in CSC 242).

The "Course Schedule" on the course website gives tentative lecture topics, assignments, exams, weights, office hours, TAs (grad TA Yifan Zhu and UG TA Qianqian Wei).

Distinction between 244/444: As noted, 444 will include some extra technical topics and assignments. Most notably: a grad essay/project that should be thought about and started soon (handout next week); ***Abstract, with reference list: Tue. Nov. 14/23.***

How to use text and supplementary readings: For any technical material you're finding difficult, it's advantageous to read the specified (usually Brachmann & Levesque) text material to get a slightly different perspective. The Russell & Norvig book is entirely optional, but can be useful as

- a very *comprehensive*, up to date source about AI (much broader scope than this course, but not much concerned with neural nets), for those interested in the subject; it has value for other courses too (242, 247/447, 246/446)
- for a different, more chatty presentation for students who find some things in the course or in B & L (or G & N) hard to understand (but less depth)
- more thorough coverage of probabilistic inference, decision making
- a possible source of ideas for grad student essays/projects (browse and see what interests you; look at bibliographic/historical notes)

Lisp: The traditional text is Wilensky's *Common LISPcraft*, which is still a good guide to learning Lisp. Another book that students seem to like is Peter Seibel's *Practical Common Lisp.* There are also excellent online guides, which you can find on the course web pages. A standard reference manual if you know Lisp already is Steele's *Common LISP: The Language*. This is available in electronic form at http://www.cs.cmu.edu/Groups/AI/html/cltl/cltl2.html (Warning: this gives the syntactic details and meaning of the various constructs but few clues about how to *use* them!).

All undergrads in the class and MS students should already (or shortly) have access to the instructional network (csug) -- the lab staff are taking care of this. You can activate Steel Bank Common Lisp by typing SBCL. This is available via ssh. For CSC 444 students on the grad network, Allegro Common Lisp is activated by typing acl (well, you may have to define this as an alias (in .bashrc) for /p/lisp/acl/linux/latest/alisp).