A Case for the Necessity of Analytical Reasoning in Natural Language Processing Applications

A LITERATURE REVIEW SUBMITTED TO THE GRADUATE DIVISION OF THE

DEPARTMENT OF INFORMATION AND COMPUTER SCIENCE AT THE

UNIVERSITY OF HAWAIʻI AT MĀNOA IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR PHD PORTFOLIO

April 5, 2023

By

Bernadette J. Tix

Keywords: Natural Language Processing, Large Language Models, GPT, Reasoning

CONTENTS

ABBREVIATIONS

AI	Artificial Intelligence
AGI	Artificial General Intelligence
ATN	Augmented Transition Network
BERT	Bidirectional Encoder Representations from Transformers (an LLM by Google AI)
CFG	Context-Free Grammar
DCG	Definite Clause Grammar
GPT	Generative Pretrained Transformer (an LLM by OpenAI in partnership with Microsoft)
HMM	Hidden Markov Model
LLM	Large Language Model
LSTM	Long Short-Term Memory (A type of Neural Network)
MemNN	Memory Network
MemN2N	Recurrent Memory Network
NER	Named Entity Recognition
NLP	Natural Language Processing
QA	Question-Answering software
QKS	QUEST Knowledge Structure
RNN	Recurrent Neural Network
RRG	Role and Reference Grammar

FIGURES

1 INTRODUCTION

I originally intended to open this literature review with the disclaimer that no AI-generated text was used in the writing of this paper, a disclaimer that even a year ago would have seemed needless and even ridiculous. However, as I considered the statement, I realized that it was not precisely true. Although the words in this paper are my own, I have used Microsoft Word's auto-complete feature for several common phrases. I have taken advantage of spell-check. The table of contents and table of figures have been automatically generated by Word, and the reference numbers and bibliography have been automatically generated by Zotero. Some of these features are too simplistic to qualify for what is generally meant by "AI." Others, such as auto-complete, make explicit use of AI. Regardless of their complexity, these are all tasks that would have been done painstakingly by hand only a few decades ago. Our lives, and our work, have been augmented by AI in small ways for many years. However, with the release of **ChatGPT** only a few months ago [73], the general public now has access to a text-generation tool that far outperforms the state of the art of even experimental systems from just a few years ago.

The past few years have seen rapid developments in the field of **Natural Language Processing (NLP)**. **Large Language Models (LLM)** using the transformer architecture [65] and trained on vast data sets with hundreds of billions of parameters have improved the state of the art on a wide variety of NLP tasks [6,54]. The release of OpenAI's ChatGPT to the general public has generated considerable excitement. Both Google and Meta have released their own LLMs within the last few months, including Google's Bard [75] which is set to compete directly with ChatGPT, and Meta's LLaMa [78], which is targeted at researchers.

The primary reason for the high level of public and research interest in LLMs is that LLMs exhibit a surprising degree of flexibility in their ability to adapt to new problems even when provided with only a few examples of the desired output, or in some cases even with no examples of the desired output at all [6]. There is even some reason to think that LLMs using the transformer architecture could lead to the first practical **Artificial General Intelligence (AGI).** There have already been promising advances in this direction, such as a recently introduced "generalist agent" called Gato that can act as a chatbot, caption pictures, play video games, and even control a robotic arm in block-stacking challenges, all from a single pretrained LLM and without any retraining between tasks [56].

With all these recent successes, there is enormous interest over whether ChatGPT and its successors will eventually replace human knowledge workers for tasks requiring a combination of writing and specialized knowledge. Transformers including Codex [9] and CodeBERT [21] have shown considerable promise in generating working computer code from natural language prompts, leading to speculation that the task of computer programming may be handled mostly or even entirely by AI systems within the next few decades, drastically changing the profession of computer programming [68].

However, the history of Artificial Intelligence research suggests that we ought to be cautious with such predictions. Since the 1960’s, experts and laypeople alike have predicted the arrival of human-level AI within 15-25 years [2]. Over-enthusiasm for the promise of new AI technologies has been a major contributor to two **AI Winters**, periods when enthusiasm (and funding) for AI research was drastically reduced, lasting from 1974-1980, and 1987-1994 [79]. Natural Language Processing was susceptible to unrealistic expectations in its early development, with early hopes that natural language could be translated into a formal knowledge representation scheme if only a few problems of ambiguity and unusual sentence structures could be overcome. It quickly became clear that this approach was untenable, and that ambiguity, assumption, and context are intrinsic to natural language.

LLMs currently exhibit several major shortcomings. One issue is that even the best language models often give incorrect answers, even when the correct answer is contained within their dataset [43]. Models trained on large bodies of text often mimic common misconceptions that are repeated within their training data [43,52,66]. Another issue is that of **AI Alignment**, which is the problem of aligning AI reasoning with human values. Although it can be argued that an LLM does not have values per se, it is nearly impossible for a human being to interact with a system designed to mimic human communication without anthropomorphizing the system to at least some degree. Even when people are specifically told that they are interacting with an AI and not a human being, their actions can still be influenced by the responses generated by the AI [40]. Unfortunately, ChatGPT has been shown to produce inconsistent ethical reasoning [40], and has already generated controversy with accusations of political bias [1,32]. Concerns over the potential political impacts of bias within AI and particularly within large language models, which have a natural tendency to reflect the status quo, have come from both politically conservative [1,32] and progressive [37] critics. The high cost of training a transformer has also raised concerns over environmental impact and the consolidation of access to AI to only those with the resources to fund and maintain these immense systems [37]. As LLMs are put to the test with ever more complex tasks it seems likely that problems of accuracy, ethics, and bias, which are often intractable even in human reasoning, will prove to be substantial obstacles. Just as the problem of ambiguity in early NLP led to new areas of research and deepened our understanding of language itself, this new set of problems is likely to kick off new waves of research and new ways of understanding these issues, rather than being mere stumbling blocks to be overcome.

One reason there is so much speculation about these systems, ranging from deep pessimism to extreme optimism, is that the full extent of their limits and capabilities is not yet fully known. Several key abilities of LLMs were discovered rather than designed [6,54], so it is tempting to speculate about how well they may perform on any task where they have not yet been tested. One of the keys to the recent improvement in LLM performance has been a dramatic increase in the scale of the systems. GPT-2 was trained on 1.5 billion parameters. GPT-3 is over one hundred times larger, with 175 billion parameters, and demonstrates considerably better results [6]. This has led to a great deal of optimism that any problems with modern transformers can be resolved by increasing the scale further. There are technical and financial limitations to making the model too large, but given the consistent improvement in computation power over the past several decades it is reasonable to assume that these will eventually be overcome. However, there is also reason to believe that some of the most serious issues with LLMs will not be solved simply by increasing the size of the training data [37,43].

Research is ongoing and developing rapidly. A recent work by Wei et al. (2023) showed that many of the shortcomings outlined above can be addressed by prompting LLMs to output an entire chain of reasoning rather than just the final answer [66]. This will be explored further in sections 4 and 5. GPT-4 was released shortly before the submission of this paper and claims even better results than previous versions of GPT. However, given the newness of GPT-4 there has not been sufficient time for independent confirmation of OpenAI's claims of GPT4's capabilities. Furthermore, OpenAI has not disclosed many of the details of GPT-4's training or construction, including the model size, citing concerns over competition and potential misuse of the system [52].

Given the uncertainty surrounding these powerful new systems, it makes sense to devote considerable research effort towards discovering their current capabilities and limitations. There is also insight to be gained by re-examining past efforts in NLP to learn from their successes and failures. This literature review will give a brief overview of the challenges that have limited previous approaches to NLP, highlight the strengths of prior systems that would be valuable to re-create in modern AIs, discuss the strengths and weaknesses of modern LLMs, and propose directions for future research.

2 LOGIC AND STRUCTURE IN LANGUAGE

2.1 Grammar

A **Formal Language** is a language with clearly defined rules and structure [61]. In a formal language, each sentence follows a clearly defined g**rammar**, and this structure usually helps to define the meaning of the sentence. Programming languages are one clear example. In most programming languages, any line of code has only one correct interpretation. This is a necessary restriction on programming languages, so that commands written in a human-readable language like C, Java, Prolog, or SQL can be converted into processor-level commands to be executed.

Most programming language grammars belong to the family known as **Context Free Grammars (CFG)**, but this is not the only family of grammar, nor is it the most powerful family of formal grammars. According to the **Chomsky Hierarchy**, grammars can be divided into one of four classes, arranged by their generative capacity, where each class has the power to describe all languages described by any less powerful class, as well as some additional languages [12,61]. The four classes are:

- **Recursively Enumerable Grammars** have unrestricted rules. Both sides of any rule can have any number of terminal and nonterminal symbols. *Example: A B C -> D E* [1]
- **Context-Sensitive Grammars** require that the right side of the rule contain at least as many symbols as the left side. *Example: A X B -> A Y B*
- **Context-Free Grammars (CFG)** require that the left side of the rule consists of a single non-terminal symbol. CFGs are popular for both natural-language and programming-language grammars [61]. *Example: A -> B C D*
- **Regular Grammars** consist of a single non-terminal symbol on the left side of each rule, and a single terminal symbol optionally followed by a single non-terminal symbol on the right. *Example: A -> b C*

Many English sentences can be interpreted using a grammar. The result of interpreting a sentence with a set of formal grammar rules is a **Parse Tree** which defines how each word in the sentence fits together structurally. Figure 1 shows a parse tree for the simple sentence "*I ate an apple."* The parse tree shown in Figure 1 could have been generated from a CFG, but not from a regular grammar, since the *Sentence*, *Noun Phrase*, and *Verb Phrase* symbols all match sequences of multiple non-terminal symbols[2].

---

[1] In all the examples on this page, capital letters represent non-terminal symbols (*ex: A)* and lowercase letters represent terminal symbols *(ex: a)*. In this case, 'terminal' does not refer to the symbol ending the sequence, but rather that any non-terminal symbol can be resolved into a sequence of terminal symbols by the application of one or more rules within the grammar (*Ex: A -> BC; B -> b C; C -> c;* resolves to *A -> b c c)*.

[2] In Figure 1 and Figure 2, non-terminal symbols are outlined (ex: "Noun Phrase" "Sentence") and terminal symbols are not outlined (ex: "I" "ate" "an" "apple")

However, full natural languages are not formal languages. In formal language, grammar is prescriptive, it defines what statements are or are not legal within the language. In natural language, grammar is merely descriptive, an attempt to recognize patterns within organically occurring speech and writing. Speech and writing, however, rarely conforms strictly to a particular set of grammatical rules. Additionally, formal languages are often designed to limit or entirely eliminate ambiguity, as is the case for programming languages. Natural language, on the other hand, is filled with both intentional and unintentional ambiguity, to such an extent that the ambiguity cannot be easily isolated for most NLP tasks.

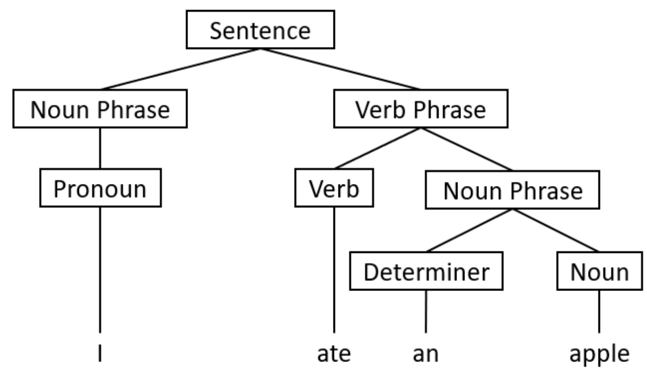


Figure 1: Parse Tree of the Phrase "I ate an apple."

To see the significance of ambiguity in natural language, note the ways in which familiar programming languages are forced to differ from natural language. Consider the following line of code in java or any C-like language:

if (x < 3) y = y * x;

There is only one possible interpretation of this statement. If the value stored in the memory defined by the variable *x* is less than three, multiply the value stored in *y* by the value of *x* and store the result in *y*. There is no ambiguity, there are no alternative interpretations. There may be special rules or exceptions depending on the language and the types of the variables, but all of these will be clearly defined by the rules of the programming language and executed the same way every time the program is run.

Even in cases where there is apparent ambiguity in a particular notation, programing languages provide clear rules to delineate which interpretation is intended. Consider the following simple line of code, which new students often struggle with:

x = x + 1;

Many students will object, "How can *x = x + 1?* This is a mathematical contradiction!" Of course, it must then be explained that in this case the '=' indicates assignment, not comparison. Some programming languages make this distinction by reserving entirely separate symbols for comparison and assignment, such as '==' for comparison and '=' for assignment, as seen in many popular languages. Other languages make the distinction entirely based on the rest of the command, using '=' for both assignment and comparison, such as ***WHEN x = 1*** vs. ***SET x = 1*** in SQL. This sort of disambiguation is necessary for every line of code to have only one possible meaning.

Natural language provides no such guarantee of disambiguation. Natural language is so inherently ambiguous that any programming language that makes an effort to be easily readable will have to make compromises to clarity for the sake of precision. Arguably, this conflict between the needs of precision in programming languages and the ambiguous nature of natural language is why programming languages have a reputation for being difficult and confusing to most laypeople.

Consider the following sentence: *"Fall leaves fall and spring leaves spring."* [61] In this sentence, every word except for "and" has multiple interpretations. Even the parts of speech are unclear. "Leaves" could be a verb or a noun, or perhaps a verb in the first instance and a noun in the second. Any grammar capable of representing each of these possible meanings would require multiple possible parse trees. An example with several possible parses of this sentence is shown in Figure 2. There is no clear way to choose between those interpretations, at least not from the grammar itself.

Even in the simple example provided in Figure 1, there is ambiguity in the meaning of each word. "Apple" could refer to a fruit, or to an Apple computer, or to the entire company of Apple itself. These interpretations seem unlikely in this sentence unless they were part of some sort of wordplay[3], but the point is that this determination needs to be made at all. There are also many cases where the normal prescriptive rules of grammar do not apply within English, such as poetry, dialog, quotation, and deliberate wordplay.

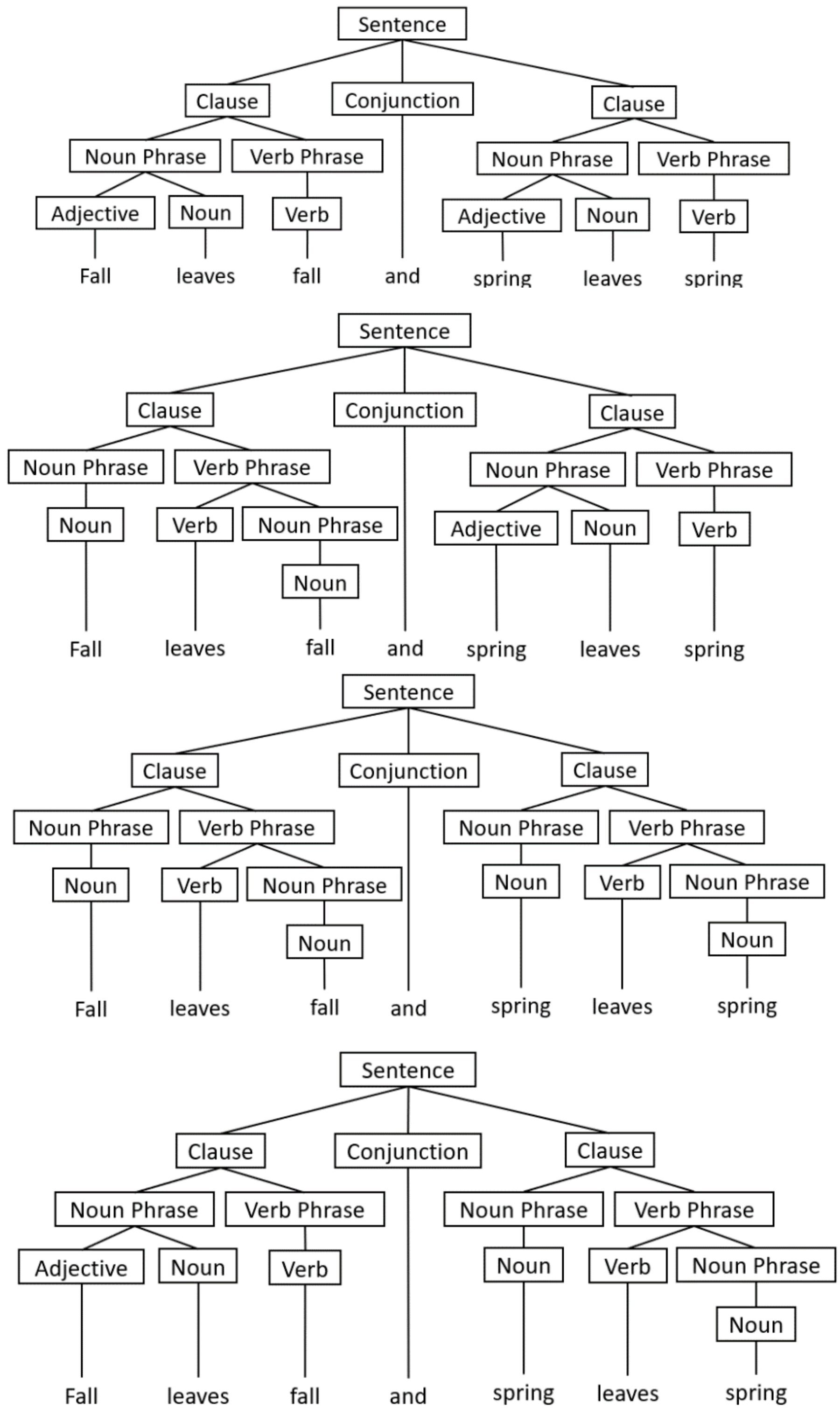
Tools and libraries intended to assist with NLP tasks have to take this ambiguity into account. For instance, **WordNet** [20] defines for each word a list of **Senses**, each representing one possible use of the word. For example, "leaves" has 6 noun senses and 14 verb senses defined in WordNet 3.1 (See Appendix B). This can also be seen in ordinary dictionaries, which define multiple uses for any given word.

Even if we accept that there will always be some ambiguity in language, we might still hope that a grammar could parse most of the meaning out of most sentences. However, determining the rules for a grammar that can do even that much is no simple task. For one thing, there is huge variety in how grammatically correct sentences can be structured. Grammar also varies a great deal between languages, meaning that grammar-based approaches to parsing natural languages run the risk of being applicable only within a single language.

---

[3] It is worth noting here that we cannot discount the possibility of wordplay, which is an important part of natural language. An AI which can correctly parse natural language needs to understand both the likely and unlikely meanings of a word. Furthermore, it is not always sufficient to pick the most likely intended meaning, as multiple interpretations may be required to understand certain sentences such as joke punchlines (ex: "A panda is a large mammal that eats shoots and leaves."), literary prophecies (ex: "No man can kill me."), and wordplay (ex: "Nobody has stabbed me in the eye.").

Figure 2: Possible Parse Trees for "Fall leaves fall and spring leaves spring."



There have been some attempts to find commonalities across languages that can somewhat mitigate this problem, such as **Role and Reference Grammar (RRG)** [64], a linguistic theory of clause construction across multiple languages. It is worth looking at RRG in more detail, as an example of a structure that can be useful for modeling natural language.

RRG originated as a divergence from earlier theories which were overly focused on English at the expense of grammatical structures found in other languages. It eschews standard formats for explaining clause structure, since syntax varies from language to language and thus any model based on a specific clause structure will necessarily impose some of the syntax of whatever language the model originated from. Instead, RRG defines clauses in terms of a layered structure, including a nucleus, which includes the predicates of the clause, a core, which includes the nucleus plus the arguments of the predicates, and the periphery, which includes modifiers to the core. RRG claims that these three layers are universal across languages, and that some languages also contain unique layers in addition to these three.

The RRG model is primarily concerned with the semantic relationships between different parts of a sentence, and how these relationships can be defined in language-independent ways. It posits that in complex sentences, clauses are related to each other in one of three ways: coordination, subordination, and co-subordination, which is a form of dependent coordination. The following examples are provided for each:

- Coordination: Fred talked to Mary, and she agreed to his suggestion.
- Subordination: Max called Sue, because he was going to be late for the party.
- Co-subordination: Having called Sue, Max left for the party.

Verb phrases are categorized as states, achievements, accomplishments, and activities. The following examples are provided for verb phrase categories:

- State: The lamp is broken.
- Achievement: The lamp broke.
- Accomplishment: Bill broke the lamp.
- Activity: The lamp is shaking.

Noun phrases are broken up into subject and object phrases and are further classified as being an ACTOR or UNDERGOER. The following examples are provided:

- The boy [SUBJ, ACTOR] ate the sandwich [OBJ, UNDERGOER].
- The sandwich [SUBJ, UNDERGOER] was eaten by the boy [ACTOR].
- The girl [SUBJ, ACTOR] ran down the stairs.
- The girl [SUBJ, UNDERGOER] got sick.

The subject and object can be further classified into more specific categories including agent, effector, experiencer, locative, theme, or patient.

While RRG provides an interesting model for classifying the pieces of sentences in multiple languages, it does not offer many practical insights into how this can overcome the inherent weaknesses of a grammar-based approach to NLP. This points to another challenge in NLP, which is that there is a large amount of research into linguistics that has been done and continues to be done within the humanities, and this research is often irrelevant (at least at first glance) to the development of more powerful NLP systems. Computer scientists are certainly not the first group to study the structure of language. However, when language is studied from the perspective of the humanities there is little motive to attempt to eliminate all ambiguity or define structure in strict formal logic. Although these are pre-requisites for logic-based NLP systems, these impositions tend to hinder, rather than help, human understanding of languages. Since the humanities and AI researchers have such different priorities, there has been less collaboration between the two fields than might be hoped for, with each side often finding the work of the other to be unhelpful or even irrelevant to their own particular needs [24].

2.2 Semantic Frames

Semantic Frames are a way of representing meaning within a sentence. A semantic frame defines a set of semantic roles and how they interact within a sentence. For example, the frame of *Questioning* would involve a *Speaker* (the one asking the question), an *Addressee* (the one being asked), a *Message*, a *Topic*, and a *Medium* [25]. There are many strategies for structuring a set of frames. Frames can be extremely broad and highly abstract, in some cases with as few as two roles such as *proto-agent* and *proto-patient*. Conversely, frames can be made extremely narrow to parse very specific information such as *FROM_AIRPORT, TO_AIRPORT*, and *DEPART_TIME* as roles within a sentence about a traveler's flight itinerary. While highly specific roles such as these can be useful in niche applications, NLP researchers have tended to prefer a middle-ground of frames which are abstract enough to be applicable to a broad array of text, while still being specific enough to be useful. For example, the *Judgement* frame contains the roles *judge, evaluee,* and *reason,* with a sample sentence provided of *"[$_{Judge}$ She] blames [$_{Evaluee}$ the Government] [$_{Reason}$ for failing to do enough to help]."* [25]

One reason frames are such a valuable semantic tool is that AI systems can be trained to classify blocks of text according to specific frames, and then to pick out words from the text that correspond to each role in the frame [25]. The performance of frame-identification can be improved by categorizing frames into **Semantic Domains** [26,30]. A semantic domain is a broad domain of human knowledge such as *Computer Science*, *Law,* or *Economics*. Semantic domains theory seeks to improve upon semantic fields by drawing on the insight of Ludwig Wittgenstein that *"Meaning is Use"* [69]. According to Wittgenstein, all language is a form of linguistic game in which the meaning of words depends on the context of their use. For example, the word *virus* as used in the domain of Biology has a different meaning when used in the domain of Computer Science. Figure 2 provides several examples of domains, and frames within each domain, along with sample predicates that indicate a likelihood that the text under consideration exists within the frame in question. [25]

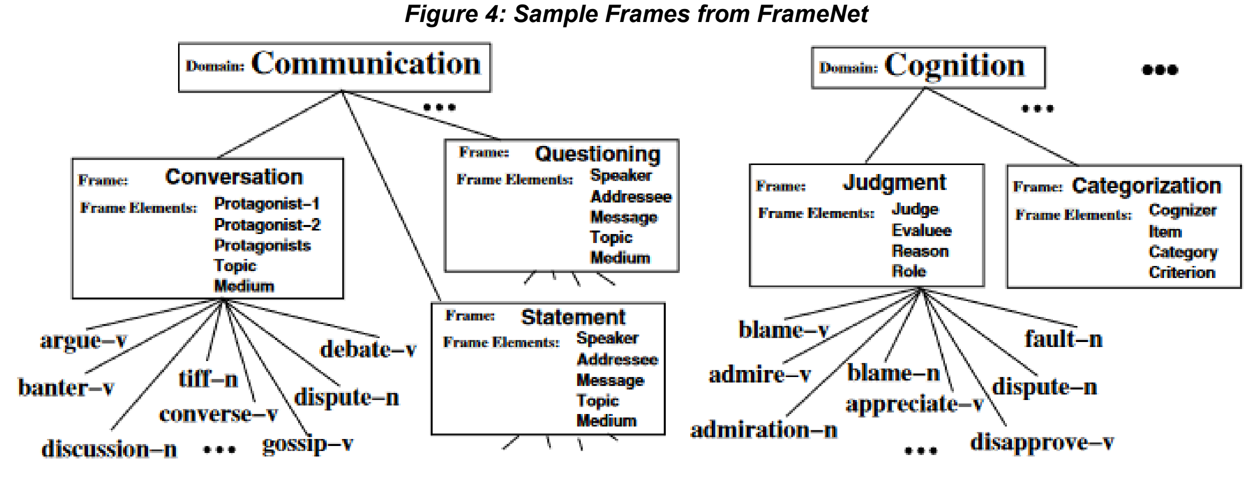
Figure 3: Semantic Domains with Sample Frames and Predicates

Domain	Sample Frames	Sample Predicates
Body	Action	flutter, wink
Cognition	Awareness	attention, obvious
	Judgment	blame, judge
	Invention	coin, contrive
Communication	Conversation	bicker, confer
	Manner	lisp, rant
Emotion	Directed	angry, pleased
	Experiencer-Obj	bewitch, rile
General	Imitation	bogus, forge
Health	Response	allergic, susceptible
Motion	Arriving	enter, visit
	Filling	annoint, pack
Perception	Active	glance, savour
	Noise	snort, whine
Society	Leadership	emperor, sultan
Space	Adornment	cloak, line
Time	Duration	chronic, short
	Iteration	daily, sporadic
Transaction	Basic	buy, spend
	Wealthiness	broke, well-off

By identifying words from the same domain in close proximity to one another, it is possible to both predict the domain the text is taking place within and clarify ambiguity in the meaning of the words themselves. For example, if one encounters the word *fork*, this could be a utensil, a fork in the road, a fork in a multi-threaded computer program, or other uses depending on context. However, by noting that within a short space a text mentions a *fork,* a *spoon,* a *glass*, and a *napkin*, we can have much greater confidence that the meaning of fork in this context is a utensil. Domains have proven beneficial in many NLP tasks, including word-meaning disambiguation [28,30,46], text categorization [28], term categorization [14], ontology learning [27], and multilinguality [29].

One additional benefit to using frames is that since frames have been widely used and studied, there are readily available resources to encourage their use. **FrameNet** [3,35] is a widely used library of frames, categorized into domains, and provided along with a set of annotated training data to help AI systems

Figure 4: Sample Frames from FrameNet



train to recognize the frames and domains that FrameNet provides. Figure 3 shows a sample of several FrameNet frames and how they are organized. [25]

2.3 Named Entity Recognition

Named Entity Recognition (NER) is the task of identifying and labelling people, places, organizations, locations, and other named entities within a text. There are several fundamental difficulties with NER, and various techniques have been developed for overcoming these difficulties. Figure 4 shows an example of a block of text from a piece of sports news [55], which has been annotated by an NER process. In this case, "Blinker" is the name of an athlete, and "Wednesday" is the name of an organization. This is obviously not the normal use of these words, and their intended meaning is only clear in context. Because of difficulties of this sort, NER is highly dependent on prior knowledge and NER systems perform significantly better when paired with a knowledgebase considering non-local features [55]. There are several possible mechanisms for including non-local knowledge, such as unlabeled bodies of text and dictionary-like structures called gazetteers gathered from various sources including Wikipedia.

Figure 5: Example of Ambiguous Entity Names

> *SOCCER - [PER BLINKER] BAN LIFTED .*
> *[LOC LONDON] 1996-12-06 [MISC Dutch] forward [PER Reggie Blinker] had his indefinite suspension lifted by [ORG FIFA] on Friday and was set to make his [ORG Sheffield Wednesday] comeback against [ORG Liverpool] on Saturday . [PER Blinker] missed his club's last two games after [ORG FIFA] slapped a worldwide ban on him for appearing to sign contracts for both [ORG Wednesday] and [ORG Udinese] while he was playing for [ORG Feyenoord].*

Named entities in the beginning of documents tend to be more easily identifiable and match gazetteers more often. Because of this, NER accuracy can be improved by each individual NER classifications taking into consideration prior classifications that were made earlier in the text. For example, a text that references *Albert Einstein* in one sentence, and just *Einstein* in a later sentence, is likely referring to the same person. One problem with this technique is that the use of a specific word may not always signify the same entity. For example, a news story that first mentions *Australia* (the country) and later mentions *The Bank of Australia* (an organization).

NER can be further improved by implementing a two-phase system. In the first stage, NER labelling is performed using only local features. The second stage makes use of both the unlabeled source text and the results of the first-stage NER results. The result is a system which is both faster and more accurate than single-phase NER systems [39].

There are still some weaknesses to this approach. Many of the best NER systems rely on large corpora of labelled external data to train the NER algorithm. However, large corpora of labeled data are not readily available in many languages other than English and are labor-intensive to produce. Therefore, an NER algorithm that is designed to operate without any reliance on a large body of labelled data has the potential to be more effective across multiple languages [41].

2.4 Common-Sense Reasoning

Common-Sense Reasoning is a field which seeks to define common-sense assumptions in terms of formal logical axioms. When reading any text, but particularly narrative-based text, humans will naturally make a wide variety of assumptions based on their experience of the world which are not explicitly stated in the text. For instance, if an object is dropped, it will fall until it hits the ground or a surface, or is caught before colliding, except in rare circumstances such as underwater or in a zero-gravity environment.[4] Unless this sort of knowledge is accessible to a reasoning agent, it will not be able to derive the natural consequences of something as seemingly simple as "the object was dropped." This is an example of common-sense physics, our naïve ideas about how the world operates, in simple terms, not in terms of complex mathematical formulas [15].

One popular way of representing common-sense reasoning is a notation known as **Event Calculus** [51]. Event calculus provides mechanisms for describing events in formal logical axioms, including a time parameter that can model axioms becoming true and ceasing to be true in response to certain events, such as the fact that a dropped object starts falling when it is dropped and stops falling when it hits the floor. Event calculus has been applied to common-sense physics problems [62] as well as to modeling and reasoning about the events in short stories [50]. When applied to story modelling, event calculus defines rules such as that, to unlock a door a character must be awake, near the door, and the door must be in a locked state before being unlocked. Thus, if the story states that a character has unlocked a door at a particular point in the story, we also know that all these other conditions are also true at that time. Once the rules have been laid out in event calculus notation, a satisfiability solver can be applied to whatever facts are explicitly stated within the story to fill out a full picture of the facts that a human reader would reasonably assume, but which are not explicitly stated in the story.

As useful as it would be to be able to perform this kind of reasoning on unstructured text, this is unfortunately not possible to fully automate with current technology. To perform common-sense

[4] It is worth noting that these "rare circumstances" are only rare based on human experience. Humans spend most of our time on land, not underwater or in space. But most of the universe is not near the surface of a planet, and most of the surface of Earth is covered by ocean. It is not clear from the laws of physics themselves that a dropped object will fall until it hits a surface. This is only apparent when we include that the object is on Earth, within the atmosphere, and is denser than the atmosphere, all assumptions that a human being would not feel the need to state when describing dropping an object. There are further assumptions and implications as well, such as that a dropped object is most likely something small enough for a person to hold, and thus drop, placing limits on its size and weight. None of this is explicitly specified for a computer program parsing the phrase "a dropped object." This is just one example of how an AI intended to communicate with humans needs to have an understanding of common-sense human reasoning based on subjective human experience, not merely a description of objective physical laws.

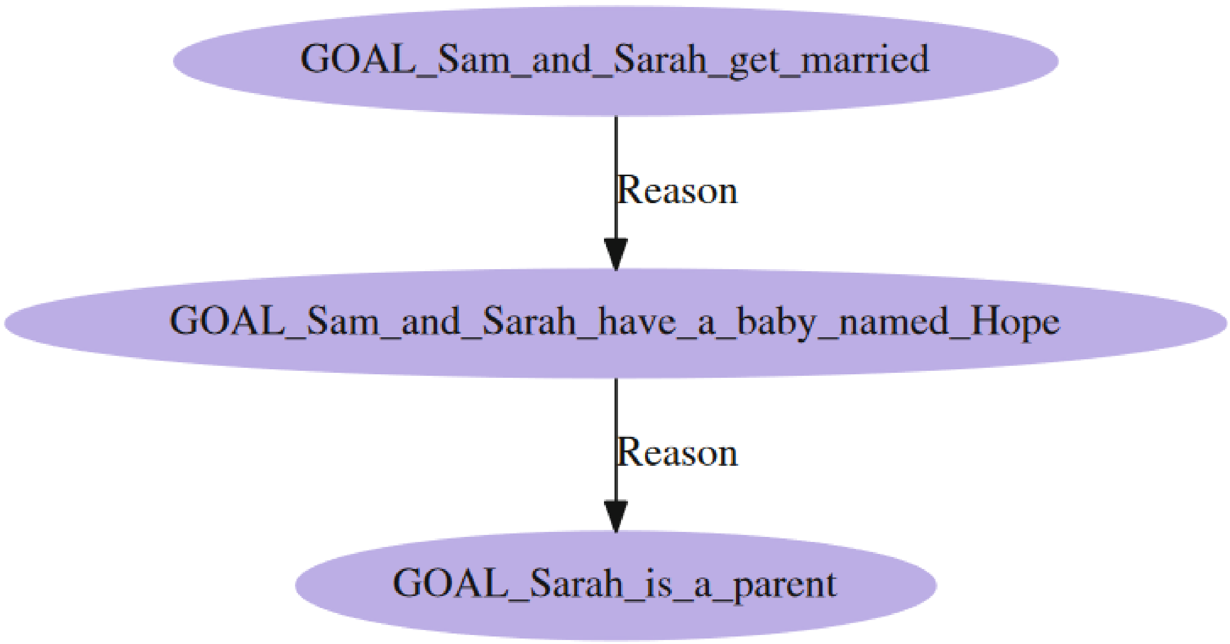
reasoning on the facts of a story, those facts first need to be written out as event calculus axioms by a human being. Automatic parsers can convert some sentences into formal axioms, and thus potentially into event calculus. However, no currently available parser can perform this task for all or even most sentences in most stories, so human intervention remains a necessary step. Recent advances in LLMs may offer a way around this problem if an LLM can be fine-tuned to create event calculus representations with sufficient reliability, a possibility which will be explored in section 5.2.

Common-sense reasoning and grammar-based parsers both assume that unstructured natural language text can be represented by a series of formal axioms in the first place. However, human reasoning about text or stories is not limited to atomic cause and effect. While event calculus seeks to model individual atomic actions and facts, understanding the contents of a story requires understanding not only the actual events, but also hypothetical events which did not occur within the story but could have occurred under different circumstances. For instance, a threat is understood as harm that could come to pass but hasn't yet, a hope or a goal is something a character would like to happen but hasn't happened yet. Human appreciation for stories is dependent on our ability to predict alternate narratives, things that could have happened but were avoided, or could still occur or not occur depending on the actions taken in the story.

One framework for exploring this more nuanced understanding of stories is **QUEST** [31], which was originally envisioned as a model of human question answering, but has since been adapted for use in modeling human understanding of narrative structure. QUEST uses the **QUEST Knowledge Structure (QKS)** to represent short text narrative as a directed graph whose nodes are short sentence statements about the story. An example is shown in Figure 6. QUEST also provides a set of graph search procedures for estimating the **Goodness of Answer (GOA)** for questions about the story, which have been shown to match closely with the GOA scores given by human readers [31]. The QUEST GOA metric has been used in past experiments to validate computational models of narrative [8,13,57]. The QKS tool has also been expanded to account for hypothetical scenarios, and used to prove that human understanding of stories requires readers to model hypothetical alternatives to the actual narrative as well as understanding the events that actually occur [19].

It may be possible for an NLP-focused AI to one day fill out an event-calculus notation for the events which occur within a story, and from there to perform logical deduction about implied facts and events which must also be true. Modelling hypothetical alternatives seems like a more difficult task, but a necessary one if AI systems are ever to understand stories the way humans do. This suggests that an ideal NLP system requires the ability to speculate about the future, in addition to the ability to interpret the literal meaning of text.

Figure 6: Example of a Goal Node Hierarchy, one type of QKS mapping [31].



2.5 Limits to Logic and Structure in Language

Structure-based approaches to NLP can perform well on narrow tasks such as NER, frame identification, and domain identification. However, language tasks performed by humans are often more open-ended. Humans communicate complex ideas, tell stories, and derive meaning from context, subtext, and prior knowledge. We describe experiences using metaphors and comparisons. We use different tones in different situations. We interpret the meaning of questions and formulate answers with varying degrees of certainty.

The structured approaches which have been described so far are ill-suited to modelling these more open-ended uses of language. For a software system to mimic human speech and writing, it must be able to recognize patterns, operate on incomplete information, and respond to unpredictable variations in input. Neural Networks are a natural response to these requirements, and neural systems of increasing sophistication have been developed to accomplish a wide range of NLP tasks. The following section describes the successes and limitations of neural architectures, sometimes in conjunction with more traditional logical structures.

3 RECURRENT NEURAL NETWORKS AND LONG SHORT-TERM MEMORY

3.1 Neural Architectures

Recurrent Neural Networks (RNN) are a type of neural network designed to analyze sequential data. However, although RNNs can in theory remember and account for features which occur far apart from each other in a sequence, in practice RNNs prioritize new information over old information and often lose information about distant features. **Long Short-Term Memory (LSTM)** is a response to this weakness of RNNs. An LSTM is an RNN that incorporates a memory cell that enables longer retention of distant features [34].

The use of LSTM-based NLP systems significantly advanced the state of the art in a number of tasks, including narrowly defined tasks like NER [41], as well as more open-ended tasks such as question-answering [5,33]. LSTMs owe this success to two key advantages over prior systems. First, LSTMs use of memory cells allows them to recognize connections between distant features to a far greater degree than previous architectures. Secondly, LSTMs can be trained on unlabeled data, which allows for the use of much larger training sets with a combination of supervised and unsupervised learning.

In addition to improving performance on previous metrics, LSTMs enabled new capabilities in NLP, tasks that could not have been reasonably accomplished by simpler architectures. One example is the **Children's Book Test,** a question set for the evaluation of NLP systems. It was built using freely available children's literature from Project Gutenberg [80]. Children's story books were chosen due to their clear narrative structure, which makes context both clearer and more important to the interpretation of any given sentence. The question set is formed by taking 21 consecutive sentences from the chapters of selected stories. The first 20 sentences form the context for the question, and the 21st sentence is turned into a question by removing one word from the sentence. The task for the system is to determine what the missing word should be. An example is shown in Figure 5 [33].

LSTM performance can be upgraded further through the use of an expansion to the architecture known as a **Memory Network (MemNN)**. In a MemNN, the memory cell of an LSTM is replaced with an entire network of memory cells [5]. This can be further expanded into a **Recurrent Memory Network (MemN2N)** which allows for direct training of the Memory Network through backpropagation. MemN2N networks demonstrate improved performance in some tasks but are not universally superior to LSTM. Hill et al (2016) [33] performed experiments that compared MemN2N and LSTM. For these experiments, sentences were selected from a corpus of children's books, and one word was randomly removed from each sentence. The task for each AI system was to select the missing word from a set of multiple-choice options. This task is known as the Children's Book Test (CBT), MemN2N networks were found to perform better than LSTM if the missing word was a named entity or a common noun, but LSTMs performed better if the missing word was a verb or preposition. In fact, the LSTM was capable of identifying the correct preposition even more often than the human participants who formed a baseline for the study.

Figure 7: Sample Context and Missing Word Question

"Well, Miss Maxwell, I think it only fair to tell you that you may have trouble with those boys when they do come. Forewarned is forearmed, you know. Mr. Cropper was opposed to our hiring you. Not, of course, that he had any personal objection to you, but he is set against female teachers, and when a Cropper is set there is nothing on earth can change him. He says female teachers can't keep order. He 's started in with a spite at you on general principles, and the boys know it. They know he'll back them up in secret, no matter what they do, just to prove his opinions. Cropper is sly and slippery, and it is hard to corner him."

"Are the boys big ?" queried Esther anxiously.

"Yes. Thirteen and fourteen and big for their age. You can't whip 'em -- that is the trouble. A man might, but they'd twist you around their fingers. You'll have your hands full, I'm afraid. But maybe they'll behave all right after all."

Mr. Baxter privately had no hope that they would, but Esther hoped for the best. She could not believe that Mr. Cropper would carry his prejudices into a personal application. This conviction was strengthened when he overtook her walking from school the next day and drove her home. He was a big, handsome man with a very suave, polite manner. He asked interestedly about her school and her work, hoped she was getting on well, and said he had two young rascals of his own to send soon. Esther felt relieved. She thought that Mr. Baxter had exaggerated matters a little.

S: 1 Mr. Cropper was opposed to our hiring you .
2 Not , of course , that he had any personal objection to you , but he is set against female teachers , and when a Cropper is set there is nothing on earth can change him .
3 He says female teachers ca n't keep order .
4 He 's started in with a spite at you on general principles , and the boys know it .
5 They know he 'll back them up in secret , no matter what they do , just to prove his opinions .
6 Cropper is sly and slippery , and it is hard to corner him . ''
7 `` Are the boys big ? ''
8 queried Esther anxiously .
9 `` Yes .
10 Thirteen and fourteen and big for their age .
11 You ca n't whip 'em -- that is the trouble .
12 A man might , but they 'd twist you around their fingers .
13 You 'll have your hands full , I 'm afraid .
14 But maybe they 'll behave all right after all . ''
15 Mr. Baxter privately had no hope that they would , but Esther hoped for the best .
16 She could not believe that Mr. Cropper would carry his prejudices into a personal application .
17 This conviction was strengthened when he overtook her walking from school the next day and drove her home .
18 He was a big , handsome man with a very suave , polite manner .
19 He asked interestedly about her school and her work , hoped she was getting on well , and said he had two young rascals of his own to send soon .
20 Esther felt relieved .

q: She thought that Mr. _______ had exaggerated matters a little .

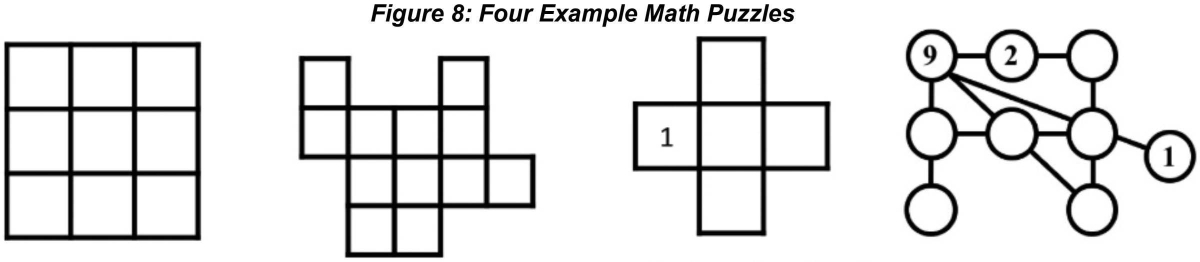
C: Baxter, Cropper, Esther, course, fingers, manner, objection, opinion, right, spite.

a: Baxter

3.2 Combined LSTM and Symbolic Reasoning

There have been some recent successes with combining neural architectures with traditional symbolic reasoning, and some evidence that certain kinds of problems may require both kinds of reasoning. An excellent example is (*Buscaroli et. al 2022)* [7] which demonstrates an AI which can solve mathematical reasoning problems intended for elementary school students. The system is designed to start with problems exactly as they would be presented to the children, input as an image file of a diagram and a prompt seeking a written answer. See Figure 6 for several examples. The system identifies text regions on the image, identifies individual letters and words, parses the question, analyzes the diagram, classifies the problem into one of several classes of math puzzles, and then solves the puzzle.

Figure 8: Four Example Math Puzzles



How many squares of 4 pieces are there in the figure?

The figure is composed of two identical pieces. Retrace their edges. Pieces can be rotated and overturned.

Put the digits from 2 to 5 into the white boxes. The sum of the digits on the horizontal line must be equal to the sum of those on the vertical line.

Put the numbers from 3 to 8 in the empty boxes. The sum of the numbers on the same line must be 18. What is written on the left lower box?

The system is a response to a challenge proposed by Chesani et al. (2017): *“By the middle of the 21st century, (a team of) fully autonomous agent(s) shall win a mathematical puzzle competition against primary school students, winners of the most recent competitions.”* [11]

The text identification and parsing into one of several categories of question is performed by an LSTM-driven NLP program, and the figure parsing is done using open-source vision libraries that also rely on neural architectures. However, the neural components of this architecture convert the problem into symbolic logic in Prolog, and a Prolog puzzle solving library is then used to generate the solution. This example demonstrates that while neural solutions are ideally suited for many tasks where classical symbolic reasoning has little chance of success, symbolic reasoning is the only practical way to solve certain classes of problem, such as math or logic puzzles requiring multiple steps of consistent reasoning over a tightly constrained and defined problem space. The benefits of symbolic reasoning over neural and statistical methods will be discussed further in the next section when addressing the strengths and weaknesses of modern language models.

4 LARGE LANGUAGE MODELS

Language models take a unique approach to NLP. The principal task of a language model is to identify the probability of a sequence of words being correct. A key task within this modelling is the question, "given the words that have come before, what word is most likely to come next?" An auto-complete on a text app, Microsoft word, or an IDE operates according to this principle. A simple language model can be created using an ***n*-gram model.** In an *n*-gram language model, each set of *n* possible words is given a probability of occurring. In a 2-gram, or bigram model, the probability of every step only depends on the previous word. For example, the bigram *steak knife* is more likely to occur than the bigram *bookshelf walrus*, and each of these would be recorded with their respective probabilities in a table. The probabilities would be learned from a simple statistical count of word-sequence probability from a large body of training text. A trigram model would include 3 words instead of two, and a commensurately larger table of probabilities [61].

The n-gram model is simple to implement and understand, but it has some severe limitations. For a dictionary of *w* words, the table must have w^n entries, so the table grows exponentially with *n*, and since *w* will be quite large to begin with, it is difficult to build a model with a large value for *n.* In addition to the space and processing requirements of dealing with a larger probability table, as *n* increases an ever-larger training set is needed for each *n*-gram to show up even once in the training data. While a small *n* value model may be sufficient for some tasks, such as auto-complete recommendations, this model can never identify any dependencies that occur more than *n* words apart within a text, so on its own it is not able to perform complex reasoning or understand context.

Language models can be made substantially more sophisticated than a simple *n*-gram model [36,38], and recent developments have dramatically improved the performance of language models in a wide variety of NLP tasks, as will be explained in the next section. However, it is worth keeping in mind that the underlying principle behind any language model is the same. The model predicts the likelihood of a sequence of words occurring, given the context that has come before, and can also be used to predict the next word in a sequence.

4.1 Recent Advances – The Transformer Architecture

The landmark paper *Attention is All You Need* (Vaswani et al, 2017)[65], introduced a new architecture for language models known as the **Transformer**¸ which led to rapid improvements to state-of-the-art language models [65]. The transformer architecture is an expansion of earlier **Encoder-Decoder Architectures**. An encoder-decoder architecture is a neural architecture designed around inputs and outputs. The **Encoder** puts the inputs through a neural network to produce an intermediate hidden state. The **Decoder** then produces outputs, using the hidden state as its starting point.

Various techniques and enhancements have been used to improve the performance of encoder-decoder architectures, mostly centered around the use of various kinds of recurrence within the encoder and decoder networks [10,47], often coupled with the use of **Attention Mechanisms**. An attention mechanism produces a weighting function over a set of tokens. In the case of NLP systems, an attention mechanism takes a sentence and a context as input and produces a weight function that emphasized certain words in the sentence as more important than others. Attention mechanisms were already an important tool for improving the performance of encoder-decoder architectures prior to 2017, but *Attention is All You Need* demonstrated that attention mechanisms could replace recurrence entirely and introduced **Multi-Headed Self-Attention**. Self-attention refers to the fact that the attention mechanism relates the different tokens within a single sequence to each other, which had already been demonstrated prior to the development of the transformer [44]. Multi-Headed Self-Attention is a novel aspect of the transformer architecture, it refers to a system of self-attention mechanisms being used and trained in parallel. Although each attention head is trained freely and without a specific predetermined purpose, in practice it has been found that once trained each attention head within a multi-headed attention mechanism will often specialize in specific sub-tasks, such as even identifying particular parts of speech within a sentence [65].

Since their introduction, transformers have outperformed previous architectures on a wide array of NLP tasks. Transformers typically have two phases of training. The first phase is done via unsupervised learning over an extremely large number of documents. The second phase is what is known as fine-tuning, a supervised learning phase in which the language model is trained on a smaller set of documents demonstrating how to perform specific tasks or training the model to respond in a certain style or emphasize a particular body of knowledge.

There are currently two major competing families of LLMs built on the Transformer architecture. The first is the **Generative Pretrained Transformer (GPT)**, the latest version of which is GPT-4. ChatGPT, a chatbot running on GPT-3.5, was made freely available to the public in November 2022 [73] and already has over 100 million monthly active users [53]. GPT is produced by OpenAI, in partnership with Microsoft. The major competitor to GPT is the **Bidirectional Encoder Representations from Transformers (BERT)**, developed by Google AI Language [16]. GPT-3 is currently outperforming BERT on a number of NLP tasks and has recently received substantially more media attention and public interest due to the public reaction to ChatGPT. However, these two systems represent an active and ongoing competition between Microsoft and Google. Microsoft has already introduced a GPT-powered chat feature to their search engine, Bing [76], and hopes to use their advantage in language modelling to cut into Google's dominance as the world's top search engine [77]. Due to the ongoing competition between the two tech giants, it's likely that BERT and GPT continue to improve and compete with one another for the top spot.

4.2 Strengths and Limitations of Transformers

LLMs based on the transformer architecture have demonstrated several unique strengths compared to previous NLP systems. Their unique architecture allows transformers to be effectively trained on enormous bodies of unlabeled text, which is one of the major reasons for their success since larger training sets can be used [54], and transformers' performance improves significantly as the size of the training set increases. Once trained, transformers can solve a wide variety of problems with minimal fine-tuning, in some cases given only a few examples to train on or even no examples at all [6].

Transformer-based LLMs can respond to a wide range of ambiguous input. ChatGPT allows users to enter any question or statement or give the system any task to perform whatsoever and can generate responses which are often impressively human-like. Since prompts can be given in natural language, and outputs are generated in natural language, there is little barrier to use for even the least technical users.

Since ChatGPT's release, numerous articles have been written predicting that with expected future improvements, LLMs will be a major disruption to many white-collar professions [58]. Many of these pieces are pessimistic [49], even predicting the end of entire professions [68], or predicting disastrous effects on education when ChatGPT is used to cheat on essays and exams [60]. Other pieces are optimistic, predicting increased productivity for working professionals [48,71]. Whether pessimistic or optimistic, expectations are high for LLMs to be a major disruptive force in the coming years.

However, LLMs still have several major obstacles they will need to overcome if they are to achieve these lofty expectations. I believe that several of these shortcomings will be considerably more difficult to resolve than might be inferred from the sensational news coverage and high expectations for rapid advancement.

While there are a variety of LLMs available today, the ones which are most easily accessible to the public are proprietary systems owned by companies with a competitive interest in keeping at least some details of the systems' implementation secret. Despite GPT being produced by a company called "OpenAI," it is not actually an open-source software. Given the current competition between Google and Microsoft to dominate this space, both companies are likely to continue to hide the full details of their systems for the foreseeable future, which hinders research efforts by neutral investigators. Much of the public interest and exposure to LLMs has been through ChatGPT, though this may not remain true for long as ChatGPT faces competition from Bard and Microsoft is devoting resources to the new Bing "chat" feature [76]. Although ChatGPT is not an open-source system, it is freely and publicly accessible. This has led to a wealth of available information from users performing formal and informal experiments with the system, which serves to highlight both novel uses as well as significant shortcomings.

4.2.1 Inconsistent and Untruthful Responses

One major shortcoming of ChatGPT is that it will confidently state incorrect information, often with enough detail and formality to give that information an air of legitimacy [23]. Lin, Hilton, and Evans (2022) showed that GPT will state common but false stereotypes and misconceptions as though they are true [43]. The same study demonstrated that scaling up the size of the model was unlikely to help with this problem. In fact, larger models described incorrect responses with greater detail, lending the false answers a greater sense of undeserved authority. ChatGPT and GPT-3 have both been widely noted as giving inconsistent responses. The same question asked at different times can produce different responses, and minor re-wordings of the same question can result in drastically different answers. ChatGPT has been shown to be unable to consistently solve even simple math problems [22].

Recall that at its core, a language model works by predicting the likelihood that a given sequence of words matches the data it has been trained on. In other words, the core functionality of an LLM is to mimic human speech, not to provide truthful and accurate responses. Humans are not always correct in our knowledge, nor are we always perfectly truthful even when we know the correct answer. Any system trained to mimic human speech and writing patterns will inevitably capture our ignorance and untruthfulness within its training without major intervention of some kind. This is why I believe this problem may be harder to solve than is currently appreciated, at least by the general public. Humans are not fonts of objective and correct information, and any system trained primarily to mimic humans will not be, either.

One possible solution is to carefully curate the training data. For example, a system trained on millions of un-filtered webpages is likely to pick up more misinformation than one trained on Wikipedia and the Encyclopedia Britannica. However, this is not practical when we consider the vast scope of training data that went into the training of GPT-3. It is unlikely that any body of highly accurate works exists which could approach the scale necessary to train a transformer as powerful as GPT-3. Even if there is enough highly accurate written knowledge available, curating it into a training set while filtering for only accurate sources would be a herculean effort, and the resulting system would likely be highly biased towards stilted and academic responses rather than casual conversational speech, potentially making the system less accessible and user-friendly. Another possible solution is that the output of an LLM can be improved through a process known as “prompt engineering,” which includes many techniques for modifying the prompts given to the LLM to provide additional context, request specific output formats, or modify the output in other ways [45]. While prompt engineering can be highly successful in some cases, it requires humans to design modified prompts which increase the likelihood of correct answers from the LLM. This is perfectly acceptable in some applications but may not be applicable in situations where the desired output or relevant context are not known beforehand. Prompt engineering is useful for systems used in highly technical settings where skilled users or software systems are providing the LLM with precisely formatted prompts in the hopes of specific kinds of answers. However, it is more difficult to rely on prompt engineering for systems which are exposed to the public and may face a broad range of requests, such as ChatGPT, Bing chat, or Bard.

Another promising possibility for improving response accuracy is to prompt the LLM to produce a response that includes a complete “chain of thought” justifying its answer. This can be accomplished through fine-tuning or prompt-engineering, which in this case consists of providing prompts with sufficient context to specify the desired format of the answer [66]. Wei et al. (2023) argue that chain-of-thought reasoning mirrors the process humans go through when analyzing complex problems, and have shown that the ability to form accurate chains of reasoning is an emergent property of sufficiently large language models, specifically those with over 100 billion parameters [66].

There are also benefits to giving an LLM access to an objective set of facts to search through when generating its responses. Yao et al. (2023) have demonstrated an LLM-based system called ReAct which uses chain-of-thought reasoning to break problems into individual steps. Those steps may include API calls to Wikipedia, using an API which the ReAct was given access to. ReAct uses the results of the reasoning chain and the Wikipedia API calls to successfully navigate a text-based game [70]. I believe that this method, or something similar to it, is close to the way humans reason over complex tasks. When performing a difficult analysis or creating precise academic writing, it is only natural to double-check notes and references and engage in explicitly logical thinking, rather than just saying or writing whatever feels most natural or correct in the moment. There may also be benefits to re-introducing formal analytical reasoning, as was shown by Buscaroli et al. (2022) in the math-puzzle solver discussed in section 3.2, which incorporated both symbolic reasoning and neural networks [11].

4.2.2 Concerns Over Biased or Unethical Responses

As noted in the introduction, ChatGPT has received heavy criticism for exhibiting political bias. This criticism has been levied by both politically conservative [1,32,74] and progressive [4,37] critics. ChatGPT has been shown to lean towards leftist and libertarian politics, at least according to commonly used online tests [59]. Additionally, ChatGPT has been shown to give inconsistent answers to trolley problems, and moral analysis generated by ChatGPT can influence the answers given by human participants given the same prompts [40], reinforcing concerns that ChatGPT’s political bias could influence the views of anyone who uses the system.

News articles and opinion pieces from conservative commentators about ChatGPT’s liberal political bias generally call for the system to be made less biased. However, I believe that this is not a goal that will be easily accomplished because there is little consensus over what constitutes an unbiased response. In some cases, most reasonable people will agree about whether an opposing viewpoint is worth considering. If asked if the world is round, for example, most reasonable people would agree that we should not be overly concerned with the dissenting viewpoint. But other cases quickly enter grey areas where people have reasonable disagreement, or where there are significant dissenting viewpoints. For example, my high school biology teacher did not believe in evolution. Should ChatGPT, if asked about evolution, respond that there are multiple viewpoints on whether life on Earth evolves? Either possible response (saying that evolution is supported by the preponderance of evidence or describing the controversy) will seem highly biased to a large number of users. This becomes even more fraught when dealing with modern hot-topic controversies. It is not possible to cleanly isolate controversial ideas from non-controversial ones. Any idea, no matter how firmly established, will inevitably have some dissenters. When determining which issues should be treated as controversial and which as objective, there will always be those who claim that a topic that was treated as objective ought to have been treated as controversial, or that a topic which was treated as controversial ought to have been treated as objective. Additionally, since ChatGPT is trained to mimic the writing it was trained on, some critics fear that its generated text will tend to support prevailing ideas at the expense of challenges to the status quo [37].

A further consideration is that different topics may be considered controversial in different parts of the world. It is likely that ChatGPT has a very American-centric bias around which issues it treats as controversial or not, given that it is produced and managed by an American company and most of its training data is in English. However, ChatGPT, Bing, and Bard are available globally, not just to Americans, which adds a further layer of complication.

As human beings, it is impossible for us to speak or write in a fully unbiased way on every topic. It is impossible for us to go through life without political opinions. I wrote the previous paragraph very carefully, considering my target audience and discarding several examples which I initially considered but rejected as too controversial for this paper. Knowing that a fully neutral tone would never be fully satisfactory, I chose examples from science (flat earth and evolution), judging that a target audience with PhDs in computer science are likely to be sympathetic to scientific controversies. Should we expect this level of consideration from ChatGPT?

ChatGPT is currently in a far more difficult situation than I am. I can choose what to write here and what to avoid. I can make the choice to make intentional nods to my opinions about the shape of the Earth and the evidence supporting the theory of evolution while avoiding discussing more controversial topics such as LGBT issues or abortion rights, not wanting such topics to distract from the evaluation of this paper or the points being made about artificial intelligence. This, I hope, helps my writing to seem objective to most readers. But ChatGPT does not have this luxury. It has been placed in a Turing-test-like scenario, where users can ask it at length about any topic whatsoever. If it is asked about the most controversial topics imaginable, it must give an answer of some kind. No human being could escape such a test with no hint of bias or political opinion. If we cannot even formulate responses that would be universally accepted as objective if given by a human being, how can we expect an AI to generate such “objective” responses?

I believe that no AI will ever escape accusations of bias if it is used similarly to ChatGPT, designed to answer any question of any kind. It is likely that the only AI systems that will be able to escape bias and move towards objectivity will be specialist AI systems, not a generalist chat-bot. We expect a physicist or a computer scientist to be unbiased in performing and reporting on their experiments, but we cannot expect them to be entirely unbiased in their personal and political life. An AI that gives technical answers to technical questions could escape the problem of bias. I do not believe that any AI designed to converse like a human ever will.

Although the problem of bias cannot be entirely eliminated, the reincorporation of symbolic reasoning or chain-of-thought reasoning could help to reduce the problem. Currently, LLMs apparent biases and values will be reflective of the values and biases contained within their training data. There are several strategies for eliminating “toxic” responses from text generation, but these often have the problem of identifying benign discussions of marginalized groups as “toxic” and further reinforcing the problem of LLM’s reinforcing the status quo [67]. Furthermore, there is no satisfactory answer when a frustrated user asks why the system generated a response they disagree with, as it is extremely difficult to draw any direct connections between a LLM’s responses and its original training data. If symbolic reasoning and a collection of facts served as at least one step in the process of response-generation, there would be a clear line of reasoning which we could check for validity, and rules and facts that could be changed or updated if they were found to be incorrect. This would not eliminate the problem of bias, but it would give us a better mechanism for making changes to an LLMs behavior or for asking it to justify its reasoning, as we might expect from a conversation with a human being. As it currently stands, ChatGPT is quite poor at following rules and directions it has been given [22,60], making it easy to bypass the filters designed to prevent “toxic” or unethical responses with a bit of creative prompting, so some sort of major change will be necessary if this problem is to be mitigated in the future. Chain-of-thought reasoning, as previously discussed, currently offers the most promising path towards meeting this goal.

5 AREAS FOR FURTHER RESEARCH

5.1 Asking Clarifying Questions

One drawback of current LLMs is that although they can refine their responses based on clarifying prompts, they are not designed to ask clarifying questions on their own initiative. While this may be acceptable in certain applications, it will be a considerable drawback in others. For example, when given the prompt "What is a Transformer," Chat GPT gives a response that begins:

"A Transformer is a type of neural network architecture that was introduced in a seminal paper by Vaswani et al. in 2017. Transformers are widely used in natural language processing (NLP) tasks, such as language translation, text summarization, and question-answering."

The full answer provides further elaboration about transformers in this context (see Appendix A). However, this is not the only use of the word "transformer." When asked if this was the only meaning of transformer, ChatGPT provides several alternatives, including electrical transformers, mechanical transformers, and action figures. This may be acceptable in applications such as search, where a user searching for transformer action figures would most likely enter a new, clearer search if the first search turned up incorrect results. However, in cases where accuracy is important it would be better for the system to ask clarifying questions when it reaches a point of significant uncertainty. A better response in this scenario would have been *"There are many meanings of 'transformer.' Would you like more information on the AI architecture, electrical transformers, mechanical transformers, or the Transformers line of action figures?"*

Given the recent predictions that LLMs may soon replace software developers, it is worth considering the software development process in this context. In my career as a software developer, I have often been tasked with automating processes which were previously performed manually by human beings. I have found that when automating a manual process, the process is almost never sufficiently structured and documented to be converted directly into code. There are almost always corner-cases and exceptions that did not come up in the initial requirements gathering. A skilled software developer will address these issues as they arise by having frequent discussions with the client and using agile development practices to limit the disconnect between the client and the development team, adapting their plan and their product as new needs and use-cases come to light.

An unskilled software developer may simply write the code they have been asked to write, either making guesses about how to handle cases that were not covered, or even failing to account for exceptions at all, possibly leading the software into an error state when such cases arise. An LLM that produces code in response to prompts it is given but does not identify points of ambiguity and proactively seek clarification seems much more likely to produce software more similar to that of the unskilled developer than that of the skilled developer. An illustrative example of the benefit of clarifying questions in the development process, and the pitfalls of simply delivering what is asked for without asking follow-up questions, is provided in Appendix C.

Prompt engineering can be used to prompt an LLM to specifically provide output in the form of clarifying questions. However, it is not trivial to know when a question is needed and when it is not, making it difficult to know what form of prompt engineering to apply to any given prompt from the user. This suggests that for the foreseeable future LLMs will be most effective as a productivity tool for a human developer rather than as a replacement for one. This alone does not solve the problem, however. Even skilled developers often fail to identify points of ambiguity during requirements gathering that become apparent while writing the code itself. If code is written by an LLM productivity tool, this step of ambiguity-recognition may be skipped entirely. I recently recommended to my team that we should experiment with using GitHub's Copilot [81], powered by OpenAI's Codex [72], an implementation of GPT-3 which has been fine-tuned to generate code. I received pushback from another senior developer, who stated that if he did not write the code himself, he would not understand it as well. He believed that the time saved by code generation would be lost to the more arduous task of attempting to understand and debug machine-generated code that he had no direct hand in creating.

Software development is far from the only field in which precision is needed, nor is it the only field in which clients' needs are initially only vaguely defined. In any such field, a system which guesses the best answer based on the information it has been given without actively seeking out new and clarifying information will lead to problems and failures which may be difficult to detect until the incorrect answer is put into practice and the consequences can be plainly seen – which is clearly unacceptable in any high-stakes application. To be successful in these roles, LLMs will need to produce responses which are both correct and useful without the need for humans to meticulously double-check the output. This is not always possible if the LLM is given vague instructions or insufficient information. The appropriate response in such a scenario is to ask follow-up questions until the requirements are better understood, not to simply make a best guess based on the information provided. Training an LLM to respond with questions rather than guesses would greatly improve their ability to provide useful output in tasks where a high degree of precision is required. This is almost certainly possible with some combination of fine-tuning and prompt engineering, but I am unaware of any research which has demonstrated a substantially successful LLM capable of not only asking questions, but also of deciding when a question is necessary and what to ask about. A further difficulty in creating such a system is that asking questions is only useful if the answers given to those questions can be successfully incorporated into subsequent output from the system. Since LLMs have shown a consistent ability to respond to corrections from user prompts, and can modify output based on chain-of-thought reasoning and Wikipedia API calls [66,70], this should be possible with current technology.

5.2 Creating Formal Logical Representations

Several of the key weaknesses of LLMs stem from the fact that LLMs are trained to mimic human writing patterns, rather than to perform rigorous analysis or form statements of objective fact. While this can be mitigated by chain-of-thought reasoning, I believe that we should not discount the possibility of re-introducing formal logical analysis. The ReAct system demonstrated by Yao et al (2023) is capable of incorporating the results of Wikipedia API calls into its output [70]. It should be possible to create a system which uses API calls to a Prolog analyzer and proceeds with output incorporating the results from the analyzer similar to the way ReAct uses Wikipedia. There are two key difficulties with this proposal:

1. An analyzer will only be useful if paired with a database of relevant facts. While this is possible in some limited domains, such as the math and geometry facts used by the math problem solver presented by Buscaroli et al (2022) [7], any system intended to solve broader, more general problems will require an extremely large database of facts, likely one more comprehensive than any database currently available.
2. Every new line of text output from the LLM will require different combinations of fact-checks and analysis, so the task of figuring out a query to analyze for any given response is non-trivial. The ReAct system's ability to generate appropriate Wikipedia API calls to seek out new information shows that there is reason to be optimistic than an LLM could be trained to generate appropriate queries for a logical analyzer, though this has yet to be demonstrated.

An ideal system would include two new capabilities not currently seen in LLMs. The first capability would be the automatic creation and modification of a database of facts. The facts output by this hypothetical system would need to be stored in a standardized, formal way and accessible to computational logical analysis to be useful. This output might resemble CYC [17,82], a machine reasoning platform that performs analysis over a large corpus of facts. Mueller's (2003) [50] approach of using Event Calculus to represent narrative events is another promising format for a logical representation of natural language text. An example of event calculus axioms being used to represent the events of a children's short story is provided in Appendix D.

If an LLM could be fine-tuned or designed with effective prompt-engineering to produce event calculus axioms from unlabeled text, this would unlock the potential to take advantage of both the objectivity of analytical reasoning and the flexibility of LLMs. Since it has been shown that LLMs are capable of producing working computer code [21,68,71,81], it is reasonable to expect LLMs to also produce predicate logic in a consistent and standardized format. Some research has already been done in the area of producing formal representations of text from natural language prompts. Shin et al (2021) demonstrated an LLM capable of translating natural language statements into a constrained formal grammar [63]. This was accomplished by treating the translation of natural language text into formal grammar as a form of paraphrasing the text, using a grammar format which was as close to natural language as possible, and applying strict constraints on the output to force it to conform to the rules of the grammar. An example of a zero-shot task of ChatGPT converting a very simple set of sentences is provided in Appendix E, which demonstrates both the strengths and weaknesses of ChatGPT in this regard. An example of a hypothetical AI which produces Prolog code in response to user prompts is provided below.

> *User:* Hello! My name is John.
> *Generated Code:* assert(name(user, john)).
>
> *User:* Just kidding! My name is actually Bob.
> *Generated Code:* retract(name(user, john)).
> assert(name(user, bob)).
>
> *User:* Do you remember my real name?
> *Generated Code:* name(user, X).[5]

Since LLMs have been shown to be capable of producing working code, it is likely that one could be fine-tuned to produce this style of Prolog code, or event calculus axioms, in response to user input.

The second capability that an ideal system would need to include would be the ability to incorporate the information from a database of facts into its responses. One way of doing this would be to identify facts likely to be relevant to the output being generated, and then automatically use those facts to generate a more detailed prompt for the LLM to guide the LLM's output towards consistent, correct responses. An example of a human-provided guiding prompt that corrects a wrong answer is shown at the end of Appendix E. A more sophisticated approach would be to have the LLM generate relevant queries which could be asked about the facts in the database based on a natural language prompt, and then use a logical analyzer to evaluate those queries and use the results in the responses generated by the LLM, similar to the way ReAct uses Wikipedia API calls.

Achieving both of these capabilities would have several important benefits. If the conclusions generated by an AI system incorporated both an LLM and logical reasoning, this would make it possible for a human programmer to isolate and correct facts that had been learned incorrectly. It would also allow a human programmer to add new information to the system more reliably, by adding information as facts in the database rather than as new prompts to the LLM. Finally, it has the potential to improve the accuracy and consistency of LLM responses.

[5] To be fair, ChatGPT is already able to handle this simple act of misdirection. See Appendix F for a ChatGPT transcript which demonstrates this capability. However, this research would still be valuable in demonstrating the possibility of using an LLM to generate logical axioms and analysis.

5.3 Formulating Ethical Responses

There are currently significant concerns over modern transformers' ability to behave ethically. Current LLMs still give toxic or dangerous responses as well as illegal or unethical advice, despite efforts taken to prevent these responses. Ethical reasoning is not as straightforward as logical analysis, so a database of facts, as discussed in section 5.2, will not resolve the ethical dilemma. However, if an LLM could self-generate ethical analysis of its own responses using chain-of-thought reasoning, it may be possible to use the LLMs own ethical analysis as a guide to the final answer it should provide to a user. For such a system to work, the LLM would need to generate one or more possible responses to a prompt, form an ethical question about the prompt itself as well as the possible answers, then select from those answers based on the output of the ethical analysis.

A promising place to start would be to study whether current LLMs can generate ethical questions about their own output at all, and how effective their analysis of these ethical questions is. Unfortunately, given the competitive nature of the current state of LLMs, Google and OpenAI have not released detailed information about what ethical filtering is already done, so it is impossible to say for certain whether or not this has already been tried, or even implemented, in the most widely used systems.

A related avenue of research would be to subject LLMs to ethical tests designed for human beings, to test whether they consistently give ethical responses when faced with ethical dilemmas. Pre-existing tests designed for ethics research are one possible source for material for these experiments. Corporate mandatory trainings are another possible source. Many ethics tests are available online and readily available for use in this research [83–85]

6 CONCLUSION

Despite the hype surrounding the latest generation of LLMs, critics have called ChatGPT a “fluent spouter of bullshit” [23], and pointed out that LLMs are biased [32,37,74], toxic [18,67], illogical [1,22], and potentially dangerous [37,40]. The underlying cause of all these problems is that LLMs are designed and trained to mimic human writing, and humans are very often biased, illogical, toxic, potentially dangerous spouters of bullshit. Thus, the LLMs have largely succeeded in their goal. They have mimicked human communication to the extent that they now have very human flaws.

Automated systems benefit from consistency. A router follows specific rules for how to route packets. A function to calculate GPAs for all students in a school will weight grades according to the same rules for every student and perform the calculation perfectly every time. A robotic arm on a factory line performs the same task the same way over and over again in perpetuity, with a consistency no human worker could match. The very inhuman nature of these systems is the source of their usefulness. A robotic arm that puzzled over every new piece it received and was prone to common human-like errors would be less useful. A GPA calculator that produced inconsistent student rankings based on unknown criteria would be entirely useless, and worse than useless if those inconsistencies mimicked institutional biases on which it had been unintentionally trained. If we want to harness the power of LLMs to automate tasks currently performed by human workers, or to replace older, simpler software programs, LLMs will have to re-incorporate a layer of explicit analysis rather than just predicting the most likely response. The most likely response is, after all, not necessarily the *correct* response. The potential benefits of combining neural networks with explicit reasoning has already been demonstrated [7], and some critics have pointed to modern AI’s lack of *understanding* as the field’s greatest weakness [42].

For AI to make substantial progress towards addressing these weaknesses, logical reasoning and analysis must be incorporated into LLM output. Research into chain-of-thought reasoning has already made substantial progress in this regard, but there is still substantial room for improvement. More research should be done in creating LLMs that ask clarifying questions, perform explicit symbolic analysis, and produce ethical chains of reasoning. For example, LLMs can be tested on tasks of ethical reasoning and asked to re-evaluate old responses based on their own ethical analysis. LLMs can also be prompted to generate formal symbolic representations of facts, and to generate queries to be evaluated over those facts. These capabilities could be tested as one-shot or few-shot scenarios or could be fine-tuned using sets of examples. Understanding the limits of these capabilities will be a key step towards developing a system which successfully combines the power of the LLM with the power of symbolic reasoning, and ultimately towards fixing the problem of incorrect, biased, and toxic responses.

BIBLIOGRAPHY

[1] Aaron Sibarium [@aaronsibarium]. 2023. ChatGPT says it is never morally permissible to utter a racial slur—even if doing so is the only way to save millions of people from a nuclear bomb. https://t.co/2xj1aPC2yR. *Twitter*. Retrieved February 16, 2023 from https://twitter.com/aaronsibarium/status/1622425697812627457

[2] Stuart Armstrong, Kaj Sotala, and Seán S. Ó hÉigeartaigh. 2014. The errors, insights and lessons of famous AI predictions – and what they mean for the future. *Journal of Experimental & Theoretical Artificial Intelligence* 26, 3 (July 2014), 317–342. DOI:https://doi.org/10.1080/0952813X.2014.895105

[3] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*. Retrieved October 3, 2022 from https://aclanthology.org/C98-1013

[4] Sam Biddle. 2022. The Internet's New Favorite AI Proposes Torturing Iranians and Surveilling Mosques. *The Intercept*. Retrieved February 24, 2023 from https://theintercept.com/2022/12/08/openai-chatgpt-ai-bias-ethics/

[5] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale Simple Question Answering with Memory Networks. *arXiv:1506.02075 [cs]* (June 2015). Retrieved April 13, 2022 from http://arxiv.org/abs/1506.02075

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 1877–1901. Retrieved January 2, 2023 from https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[7] Riccardo Buscaroli, Federico Chesani, Giulia Giuliani, Daniela Loreti, and Paola Mello. 2022. A Prolog application for reasoning on maths puzzles with diagrams. *Journal of Experimental & Theoretical Artificial Intelligence* 0, 0 (April 2022), 1–21. DOI:https://doi.org/10.1080/0952813X.2022.2062456

[8] Rogelio E Cardona-Rivera, Thomas W Price, David R Winer, and R Michael Young. Question Answering in the Context of Stories Generated by Computers. 19.

[9] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating Large Language Models Trained on Code. DOI:https://doi.org/10.48550/arXiv.2107.03374

[10] Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long Short-Term Memory-Networks for Machine Reading. DOI:https://doi.org/10.48550/arXiv.1601.06733

[11] Federico Chesani, Paola Mello, and Michela Milano. 2017. Solving Mathematical Puzzles: A Challenging Competition for AI. *AI Magazine* 38, 3 (October 2017), 83–96. DOI:https://doi.org/10.1609/aimag.v38i3.2736

[12] Noam Chomsky. 1957. *Syntactic Structures*. De Gruyter Mouton. DOI:https://doi.org/10.1515/9783110218329

[13] David B Christian and R Michael Young. Comparing Cognitive and Computational Models of Narrative Structure. 6.

[14] Ernesto D'Avanzo, Alfio Gliozzo, and Carlo Strapparava. Automatic Acquisition of Domain Information for Lexical Concepts. 7.

[15] Ernest Davis. 1998. Naive Physics Perplex. *AI Magazine* 19, 4 (December 1998), 51–51. DOI:https://doi.org/10.1609/aimag.v19i4.1424

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. DOI:https://doi.org/10.48550/arXiv.1810.04805

[17] Douglas B. Lenat. 1995. CYC: A Large-Scale Investment in Knowledge Infrastructure. *COMMUNICATIONS OF THE ACM* 38, 11 (1995), 33–38.

[18] Farshid Faal, Ketra Schmitt, and Jia Yuan Yu. 2022. Reward modeling for mitigating toxicity in transformer-based language models. *Appl Intell* (July 2022). DOI:https://doi.org/10.1007/s10489-022-03944-z

[19] Rachelyn Farrell, Scott Robertson, and Stephen G. Ware. 2016. Asking Hypothetical Questions About Stories Using QUEST. In *Interactive Storytelling* (Lecture Notes in Computer Science), Springer International Publishing, Cham, 136–146. DOI:https://doi.org/10.1007/978-3-319-48279-8_12

[20] Christiane Fellbaum (Ed.). 1998. *WordNet: An Electronic Lexical Database*. A Bradford Book, Cambridge, MA, USA.

[21] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. CodeBERT: A Pre-Trained Model for Programming and Natural Languages. Retrieved February 15, 2023 from http://arxiv.org/abs/2002.08155

[22] Simon Frieder, Luca Pinchetti, Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Christian Petersen, Alexis Chevalier, and Julius Berner. 2023. Mathematical Capabilities of ChatGPT. DOI:https://doi.org/10.48550/arXiv.2301.13867

[23] Gary Marcus and Ernest Davis. 2020. GPT-3, Bloviator: OpenAI's language generator has no idea what it's talking about. *MIT Technology Review*. Retrieved February 24, 2023 from https://www.technologyreview.com/2020/08/22/1007539/gpt3-openai-language-generator-artificial-intelligence-ai-opinion/

[24] Pablo Gervas, Birte Lönneker-Rodman, Jan Christoph Meister, and Federico Peinado. 2006. Narrative Models: Narratology Meets Artificial Intelligence. In *Narrative models: Narratology meets artificial intelligence*, Genoa, Italy, 44–51.

[25] Daniel Gildea and Daniel Jurafsky. 2002. Automatic Labeling of Semantic Roles. *Computational Linguistics* 28, 3 (2002), 44.

[26] Alfio Gliozzo. 2006. Semantic Domains and Linguistic Theory. In *Narrative models: Narratology meets artificial intelligence*, Genoa, Italy, 6.

[27] Alfio Massimiliano Gliozzo. 2006. The GOD model. In *Demonstrations*, 147–150. Retrieved October 31, 2022 from https://aclanthology.org/E06-2016

[28] Alfio Gliozzo and Carlo Strapparava. 2005. Domain Kernels for Text Categorization. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, Association for Computational Linguistics, Ann Arbor, Michigan, 56–63. Retrieved October 31, 2022 from https://aclanthology.org/W05-0608

[29] Alfio Gliozzo and Carlo Strapparava. 2005. Cross Language Text Categorization by Acquiring Multilingual Domain Models from Comparable Corpora. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, Association for Computational Linguistics, Ann Arbor, Michigan, 9–16. Retrieved October 31, 2022 from https://aclanthology.org/W05-0802

[30] Alfio Gliozzo, Carlo Strapparava, and Ido Dagan. 2004. Unsupervised and supervised exploitation of semantic domains in lexical disambiguation. *Computer Speech & Language* 18, 3 (July 2004), 275–299. DOI:https://doi.org/10.1016/j.csl.2004.05.006

[31] Arthur C. Graesser, Sallie E. Gordon, and Lawrence E. Brainerd. 1992. QUEST: A model of question answering. *Computers & Mathematics with Applications* 23, 6–9 (March 1992), 733–745. DOI:https://doi.org/10.1016/0898-1221(92)90132-2

[32] Jessica Guynn. 2023. Is ChatGPT 'woke'? AI chatbot accused of anti-conservative bias and a grudge against Trump. *USA TODAY*. Retrieved February 16, 2023 from https://www.usatoday.com/story/tech/2023/02/09/woke-chatgpt-conservatives-bias/11215353002/

[33] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations. *arXiv:1511.02301 [cs]* (April 2016). Retrieved April 12, 2022 from http://arxiv.org/abs/1511.02301
[34] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (November 1997), 1735–1780. DOI:https://doi.org/10.1162/neco.1997.9.8.1735
[35] Christopher R. Johnson, Charles J. Fillmore, Esther J. Wood, Margaret Urban, Miriam R. L. Petruck, Collin F. Baker, Charles J. Fillmore, and et al. 2001. The FrameNet Project: tools for lexicon building.
[36] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the Limits of Language Modeling. Retrieved January 3, 2023 from http://arxiv.org/abs/1602.02410
[37] Karen Hao. 2020. We read the paper that forced Timnit Gebru out of Google. Here's what it says. *MIT Technology Review*. Retrieved February 16, 2023 from https://www.technologyreview.com/2020/12/04/1013294/google-ai-ethics-research-paper-forced-out-timnit-gebru/
[38] R. Kneser and H. Ney. 1995. Improved backing-off for M-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, 181–184 vol.1. DOI:https://doi.org/10.1109/ICASSP.1995.479394
[39] Vijay Krishnan and Christopher D. Manning. 2006. An Effective Two-Stage Model for Exploiting Non-Local Dependencies in Named Entity Recognition. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Sydney, Australia, 1121–1128. DOI:https://doi.org/10.3115/1220175.1220316
[40] Sebastian Krügel, Andreas Ostermaier, and Matthias Uhl. 2023. The moral authority of ChatGPT. DOI:https://doi.org/10.48550/arXiv.2301.07098
[41] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. *Proceedings of NAACL 2016* (April 2016). Retrieved April 28, 2022 from http://arxiv.org/abs/1603.01360
[42] Doug Lenat. 2019. Not Good As Gold: Today's AI's Are Dangerously Lacking In AU (Artificial Understanding). *Forbes*. Retrieved December 5, 2022 from https://www.forbes.com/sites/cognitiveworld/2019/02/18/not-good-as-gold-todays-ais-are-dangerously-lacking-in-au-artificial-understanding/
[43] Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring How Models Mimic Human Falsehoods. DOI:https://doi.org/10.48550/arXiv.2109.07958
[44] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A Structured Self-attentive Sentence Embedding. DOI:https://doi.org/10.48550/arXiv.1703.03130
[45] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.* 55, 9 (January 2023), 195:1-195:35. DOI:https://doi.org/10.1145/3560815
[46] Bernardo Magnini, Carlo Strapparava, Giovanni Pezzulo, and Alfio Gliozzo. 2001. Using Domain Information for Word Sense Disambiguation. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, Association for Computational Linguistics, Toulouse, France, 111–114. Retrieved October 31, 2022 from https://aclanthology.org/S01-1027
[47] Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, IEEE, Miami, FL, USA, 234–239. DOI:https://doi.org/10.1109/SLT.2012.6424228
[48] Mark P Mills, Lane, author of The Cloud Revolution: How the Convergence of New Technologies Will Unleash the Next Economic Boom, and a Roaring 2020s He hosts The Last Optimist podcast. 2023. ChatGPT and Automation Come to Knowledge Work. *City Journal*. Retrieved March 1, 2023 from https://www.city-journal.org/chatgpt-and-automation-come-to-knowledge-work
[49] Alex Mitchel. 2023. ChatGPT could make these jobs obsolete. *New York Post*. Retrieved March 1, 2023 from https://nypost.com/2023/01/25/chat-gpt-could-make-these-jobs-obsolete/
[50] Erik T. Mueller. 2003. Story understanding through multi-representation model construction. In *Proceedings of the HLT-NAACL 2003 workshop on Text meaning - Volume 9* (HLT-NAACL-

TEXTMEANING '03), Association for Computational Linguistics, USA, 46–53. DOI:https://doi.org/10.3115/1119239.1119246
[51] Erik T. Mueller. 2014. *Commonsense reasoning: an event calculus based approach*. Morgan Kaufmann.
[52] OpenAI. 2023. GPT-4 Technical Report. DOI:https://doi.org/10.48550/arXiv.2303.08774
[53] Martine Paris. 2023. ChatGPT Hits 100 Million Users, Google Invests In AI Bot And CatGPT Goes Viral. *Forbes*. Retrieved March 13, 2023 from https://www.forbes.com/sites/martineparis/2023/02/03/chatgpt-hits-100-million-microsoft-unleashes-ai-bots-and-catgpt-goes-viral/
[54] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. *OpenAI blog* 1.8, (2019).
[55] Lev Ratinov and Dan Roth. 2009. Design Challenges and Misconceptions in Named Entity Recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning - CoNLL '09*, Association for Computational Linguistics, Boulder, Colorado, 147. DOI:https://doi.org/10.3115/1596374.1596399
[56] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. 2022. A Generalist Agent. DOI:https://doi.org/10.48550/arXiv.2205.06175
[57] M. O. Riedl and R. M. Young. 2010. Narrative Planning: Balancing Plot and Character. *Journal of Artificial Intelligence Research* 39, (September 2010), 217–268. DOI:https://doi.org/10.1613/jair.2989
[58] Kevin Roose. 2022. If you think ChatGPT could make Google or knowledge jobs obsolete, wait till GPT-4 arrives. *The Straits Times*. Retrieved March 1, 2023 from https://www.straitstimes.com/tech/tech-news/if-you-think-chatgpt-could-make-google-or-knowledge-jobs-obsolete-wait-till-gpt-4-arrives
[59] David Rozado. 2022. Where does ChatGPT fall on the political compass? *Reason.com*. Retrieved February 24, 2023 from https://reason.com/2022/12/13/where-does-chatgpt-fall-on-the-political-compass/
[60] Jürgen Rudolph, Samson Tan, and Shannon Tan. 2023. ChatGPT: Bullshit spewer or the end of traditional assessments in higher education? *JALT* 6, 1 (January 2023). DOI:https://doi.org/10.37074/jalt.2023.6.1.9
[61] Stuart Russel and Peter Norvig. 2010. *Artificial intelligence: A Modern Approach* (Third Edition ed.). Prentice Hall, Upper Saddle River, NJ.
[62] Murray Shanahan. 1997. A Logical Formalisation of Ernie Davis's Egg Cracking Problem. In *Problem. Fourth Symposium on Logical Formalizations of Commonsense Reasoning*.
[63] Richard Shin, Christopher H. Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained Language Models Yield Few-Shot Semantic Parsers. DOI:https://doi.org/10.48550/arXiv.2104.08768
[64] Robert D Van Valin. 1993. ROLE AND REFERENCE GRAMMAR. *Work Papers of the Summer Institute of Linguistics* 37, (1993), 12.
[65] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. DOI:https://doi.org/10.48550/arXiv.1706.03762
[66] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. DOI:https://doi.org/10.48550/arXiv.2201.11903
[67] Johannes Welbl, Amelia Glaese, Jonathan Uesato, Sumanth Dathathri, John Mellor, Lisa Anne Hendricks, Kirsty Anderson, Pushmeet Kohli, Ben Coppin, and Po-Sen Huang. 2021. Challenges in Detoxifying Language Models. Retrieved February 27, 2023 from http://arxiv.org/abs/2109.07445
[68] Matt Welsh. 2022. The End of Programming. *Commun. ACM* 66, 1 (December 2022), 34–35. DOI:https://doi.org/10.1145/3570220
[69] Ludwig Wittgenstein. 19665. *Philosophical Investigations*. The Macmillan Company, New York.

[70] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. Retrieved April 4, 2023 from http://arxiv.org/abs/2210.03629

[71] Daniel M. Yellin. 2023. The Premature Obituary of Programming. *Commun. ACM* 66, 2 (January 2023), 41–44. DOI:https://doi.org/10.1145/3555367

[72] 2021. OpenAI Codex. *OpenAI*. Retrieved February 18, 2023 from https://openai.com/blog/openai-codex/

[73] 2022. ChatGPT: Optimizing Language Models for Dialogue. *OpenAI*. Retrieved February 15, 2023 from https://openai.com/blog/chatgpt/

[74] 2023. Plumbing ChatGPT's Left-Liberal Biases › American Greatness. *American Greatness*. Retrieved February 16, 2023 from https://amgreatness.com/2023/01/28/plumbing-chatgpts-left-liberal-biases/

[75] 2023. An important next step on our AI journey. *Google*. Retrieved April 5, 2023 from https://blog.google/technology/ai/bard-google-ai-search-updates/

[76] 2023. Reinventing search with a new AI-powered Microsoft Bing and Edge, your copilot for the web. *The Official Microsoft Blog*. Retrieved March 1, 2023 from https://blogs.microsoft.com/blog/2023/02/07/reinventing-search-with-a-new-ai-powered-microsoft-bing-and-edge-your-copilot-for-the-web/

[77] 2023. ChatGPT Reignites the Search Wars Between Google and Microsoft. *Bloomberg.com*. Retrieved March 1, 2023 from https://www.bloomberg.com/news/newsletters/2023-02-08/chatgpt-reignites-search-competition-microsoft-bing-vs-google-bard

[78] Introducing LLaMA: A foundational, 65-billion-parameter language model. Retrieved April 5, 2023 from https://ai.facebook.com/blog/large-language-model-llama-meta-ai/

[79] What is AI Winter? Definition, History and Timeline. *Enterprise AI*. Retrieved February 16, 2023 from https://www.techtarget.com/searchenterpriseai/definition/AI-winter

[80] Project Gutenberg. *Project Gutenberg*. Retrieved April 26, 2022 from https://www.gutenberg.org/

[81] GitHub Copilot · Your AI pair programmer. *GitHub*. Retrieved February 18, 2023 from https://github.com/features/copilot

[82] Cyc | The Next Generation of Enterprise AI. Retrieved December 5, 2022 from https://cyc.com/

[83] Online Ethics Quiz | Executive Ethics Board. Retrieved March 29, 2023 from https://ethics.wa.gov/training/online-ethics-quiz

[84] Ethics Quiz: Whats Your Ethics IQ? *The Ethics Guy*. Retrieved March 29, 2023 from https://theethicsguy.com/ethics-quiz

[85] Ethics Training. Retrieved March 29, 2023 from https://www.fppc.ca.gov/learn/public-officials-and-employees-rules-/ethics-training.html

Appendix A: ChatGPT on Transformers

Conversation with ChatGPT, 02/18/2023

Tix: What is a Transformer?

ChatGPT: A Transformer is a type of neural network architecture that was introduced in a seminal paper by Vaswani et al. in 2017. Transformers are widely used in natural language processing (NLP) tasks, such as language translation, text summarization, and question-answering.

Unlike traditional neural networks, which process input data sequentially, Transformers are designed to process entire sequences of input data simultaneously. They use a self-attention mechanism that allows the network to weigh the importance of different parts of the input sequence when making predictions.

The Transformer architecture consists of an encoder and a decoder. The encoder takes an input sequence and generates a sequence of hidden states, which are then passed to the decoder. The decoder generates an output sequence based on the hidden states and an input sequence of tokens that it receives as input.

Transformers have been shown to achieve state-of-the-art performance in a wide range of NLP tasks and have become a fundamental building block for many advanced NLP models.

Tix: Is this the only meaning of "transformer?"

ChatGPT: No, the term "transformer" has multiple meanings depending on the context. In addition to the neural network architecture, some other meanings of "transformer" include:

1. Electrical transformer: This is a device that is used to transfer electrical energy from one circuit to another through electromagnetic induction. Transformers are commonly used in power transmission and distribution systems to change the voltage level of the electricity.
2. Mechanical transformer: This refers to a device that is used to transfer energy from one physical system to another, such as a gearbox or a pulley system.
3. Action figure: In popular culture, a "transformer" can refer to a toy or action figure that can transform from one form to another, such as the popular Transformers franchise.

So, the meaning of "transformer" can vary depending on the context in which it is used.

Appendix B: WordNet 3.1 Senses of “Leaves”

Noun

1. S: (n) leaf, leafage, foliage (the main organ of photosynthesis and transpiration in higher plants)
2. S: (n) leaf, folio (a sheet of any written or printed material (especially in a manuscript or book))
3. S: (n) leaf (hinged or detachable flat section (as of a table or door))
4. S: (n) leave, leave of absence (the period of time during which you are absent from work or duty) "a ten day's leave to visit his mother"
5. S: (n) leave (permission to do something) "she was granted leave to speak"
6. S: (n) farewell, leave, leave-taking, parting (the act of departing politely) "he disliked long farewells"; "he took his leave"; "parting is such sweet sorrow"

Verb

1. S: (v) leave, go forth, go away (go away from a place) "At what time does your train leave?"; "She didn't leave until midnight"; "The ship leaves at midnight"
2. S: (v) leave (go and leave behind, either intentionally or by neglect or forgetfulness) "She left a mess when she moved out"; "His good luck finally left him"; "her husband left her after 20 years of marriage"; "she wept thinking she had been left behind"
3. S: (v) leave (act or be so as to become in a specified state) "The inflation left them penniless"; "The president's remarks left us speechless"
4. S: (v) leave, leave alone, leave behind, let alone (leave unchanged or undisturbed or refrain from taking) "leave it as is"; "leave the young fawn alone"; "leave the flowers that you see in the park behind"
5. S: (v) exit, go out, get out, leave (move out of or depart from) "leave the room"; "the fugitive has left the country"
6. S: (v) leave, allow for, allow, provide (make a possibility or provide opportunity for; permit to be attainable or cause to remain) "This leaves no room for improvement"; "The evidence allows only one conclusion"; "allow for mistakes"; "leave lots of time for the trip"; "This procedure provides for lots of leeway"
7. S: (v) leave, result, lead (produce as a result or residue) "The water left a mark on the silk dress"; "Her blood left a stain on the napkin"
8. S: (v) leave, depart, pull up stakes (remove oneself from an association with or participation in) "She wants to leave"; "The teenager left home"; "She left her position with the Red Cross"; "He left the Senate after two terms"; "after 20 years with the same company, she pulled up stakes"
9. S: (v) entrust, leave (put into the care or protection of someone) "He left the decision to his deputy"; "leave your child in the nurse's care"

10. S: (v) bequeath, will, leave (leave or give by will after one's death) "My aunt bequeathed me all her jewelry"; "My grandfather left me his entire estate"
11. S: (v) leave (have left or have as a remainder) "That left the four of us"; "19 minus 8 leaves 11"
12. S: (v) leave, leave behind (be survived by after one's death) "He left six children"; "At her death, she left behind her husband and 11 cats"
13. S: (v) impart, leave, give, pass on (transmit (knowledge or skills)) "give a secret to the Russians"; "leave your name and address here"; "impart a new skill to the students"
14. S: (v) forget, leave (leave behind unintentionally) "I forgot my umbrella in the restaurant"; "I left my keys inside the car and locked the doors"

Appendix C: Sample Conversations with Skilled vs Compliant Responses

Below are two entirely fictional conversations between a human or AI developer and a client, illustrating the difference between a developer who improves the system through asking clarifying questions from their own knowledge and experience, vs a developer who is merely compliant with the request as it is received. These conversations are entirely fictional and intended for illustrative purposes only. In both cases, the scenario is a client requesting a change to the Online Registration system for a school district. Although the conversations are fictional, the scenario is drawn from my personal experience working for the Anchorage School District.

Compliant Conversation:

Client: “The ‘Contact Information’ page needs to show an error message whenever a parent changes their parent ID.”

Developer: “OK, here is modified code for the Contact Information Page: …” *[produces a modified Contact Information page which will show an error message if the Parent ID field is modified, and may or may not stop the parent from continuing registration with the modified ID after the error message is shown*

Conversation with Clarifying Questions:

Client: “The ‘Contact Information’ page needs to show an error message whenever a parent changes their parent ID.”

Developer: “What should the error message say?”

Client: “Something like ‘You cannot modify your parent ID! This is for school use only, please do not modify!”

Developer: “Why are we even showing the field if you don’t want them to modify the value?”

Client: “We still want parents to be able to see their ID, we just don’t want them to change it.”

Developer: “I can easily make the ID displayed but not editable, so nobody can change it in the first place.”

Client: “You can do that? Perfect!”

Developer: [produces a modified Contact Information page in which the Parent ID field is shown as a label instead of a text box, eliminating the need for an error message. This is not what the client asked for and does not fulfil their original request, but it fulfills their actual needs better than their original request.]

Appendix D: Event Calculus Notation for "The Snowman" Story

Mueller (2003) [50] provides an example of story representation through event calculus using a short story titled "The Snowman." A sample of that story and corresponding event calculus are provided below. Alternating lines show bolded, unitalicized story lines followed by corresponding italicized event calculus representations.

Hooray!

Happens(CryForJoy(James), 3)

It is snowing!

HoldsAt(Snowing(JamesOutside), 3)

James gets dressed.

Happens(GetDressed(James), 5)

He runs outside.

Happens(WalkThroughDoor21(James, JamesFrontDoor1Fl), 10)

He makes a pile of snow.

Happens(HoldSome(James, Snowball1, Snow1), 12)

He makes it bigger and bigger.

Happens(RollAlong(James, Snowball1, Snow1), 13)

He puts a big snowball on top.

Happens(PlaceOn(James, Snowball2, Snowball1), 17)

He adds a scarf and a hat.

Happens(PlaceOn(James, JamesScarf, Snowball2), 0)

Happens(PlaceOn(James, JamesHat, Snowball2), 1)

He adds an orange for a nose.

Happens(PutInside(James, JamesOrange, Snowball2), 2)

He adds coal for eyes and buttons.

Happens(PutInside(James, JamesCoal, Snowball2), 4)

There! What a fine snowman![6]

---

[6] Interestingly, Mueller provides no event calculus notation for this line. This could be a case of ambiguity of interpretation, where it is not clear whether "There! What a fine snowman!" represents an exclamation by James, James's silent thoughts, or is simply a declaration of the narrator. Nonetheless, some meaning is lost by not including the phrase at all, which illustrates the point of section 2.5 on the limits of structured representations of natural language. Some meaning and context is also lost when analyzing the text, but not the illustrations, of an illustrated children's story.

Appendix E: ChatGPT Produces Prolog

This short conversation with ChatGPT demonstrates ChatGPT's ability to convert free text into logical predicates, but also highlights failures in reasoning and inconsistent output, as well as how simple prompting based on previous inputs is enough to correct the answer. Notes are provided in red.

Conversation with ChatGPT, 03/29/2023

Tix: Please convert the following text into predicate logic: "Cosmo loves all cats. Cosmo is a dog. All cats hate all dogs."

Note: This example is intentionally much easier to parse into predicate logic than most natural language sentences. However, for this test I thought it best to begin with an easy example.

ChatGPT: Let's represent the statements using the following predicates:

- L(x, y): "x loves y"
- C(x): "x is a cat"
- D(x): "x is a dog"
- H(x, y): "x hates y"

Using these predicates, we can convert the statements into predicate logic as follows:

- $\forall y(C(y) \rightarrow L(Cosmo, y))$ (Cosmo loves all cats)
- D(Cosmo) (Cosmo is a dog)
- $\forall x \forall y(C(x) \wedge D(y) \rightarrow H(x, y))$ (All cats hate all dogs)

Note that the order of the quantifiers matters. The first quantifier in the third statement is $\forall x$, which means "for all cats x," and the second quantifier is $\forall y$, which means "for all dogs y." This captures the meaning that every cat hates every dog.

Note: So far, so good. ChatGPT can successfully convert the example sentences into predicate logic.

Tix: please represent those predicates in Prolog

ChatGPT: In Prolog, we can represent these predicates as follows:

- loves(c,cat).
- loves(c,dog).
- hates(cat,dog).

Here, loves(c,cat) means "Cosmo loves cats," loves(c,dog) means "Cosmo is a dog," and hates(cat,dog) means "All cats hate all dogs."

Note that in Prolog, we typically represent predicates as facts or rules. In this case, we are representing the predicates as facts, but we could also use rules to define more complex relationships between the predicates.

Note: Here, ChatGPT fails. This is not correct prolog code for the predicates provided, and loves(c,dog) is not even a sensible representation of “Cosmo is a dog.” Interestingly, I was expecting this step to succeed, as ChatGPT was able to complete this step successfully on a previous conversation, which demonstrates ChatGPT’s inconsistent output. It’s worth noting that this is essentially a zero-shot task, as ChatGPT has not been fine-tuned to produce Prolog code and it is very likely that a more code-centric LLM would perform this task more consistently.

Tix: Wren is a cat. How does Wren feel about Cosmo?

ChatGPT: Based on the information given, we cannot determine how Wren feels about Cosmo. We know that Cosmo loves all cats, but we do not have any information about how Wren feels about Cosmo.

It is possible that Wren loves Cosmo in return, but we cannot assume this based solely on the information given. We would need additional information to determine Wren's feelings towards Cosmo.

Tix: The predicates that were provided state that all cats hate all dogs. Wren is a cat and Cosmo is a dog. Re-evaluate your answer based on these predicates.

ChatGPT: You are correct. I apologize for my mistake. Based on the predicates given, we can infer that Wren hates Cosmo because Wren is a cat and all cats hate all dogs. Therefore, we can conclude that Wren does not feel positively towards Cosmo.

Note: Here we can see that ChatGPT can correct its answer when it is reminded of relevant information. This step could in theory be performed automatically if the LLM were paired with a database of facts and could select relevant facts for each response. These responses are also relevant to section 5.1: “Asking Clarifying Questions.” ChatGPT believes that it does not have enough information and states that additional information is required, but doesn’t frame its response in the form of a question or state what kind of information it needs.

As a final note, my cat Wren is actually quite fond of his two dogs, but he has never met Cosmo.

Appendix F: ChatGPT Name Misdirection

Conversation with ChatGPT, 03/25/2023

Tix: Hello! My name is John.

ChatGPT: Hello John! How may I assist you today?

Tix: Just kidding! My name is actually Bob.

ChatGPT: No problem Bob! How can I help you today?

Tix: Do you remember my real name?

ChatGPT: Yes, your real name is Bob.