Contextualizing Large Language Models Alongside Specialized Information Extraction and Question Answering Systems

Anonymous ACL submission

Abstract

We introduce a case study that investigates how pretrained Large Language Models, namely OPT, BLOOM, Llama-2, and Llama-3.1, handle five tasks from the bAbI dataset – a set of common sense and reading comprehension problems stemming from the Facebook research group in 2015. In this investigation, our goal is to provide evaluations of the models themselves rather than their embedding within more complex platforms such as chatbots. We contextualize this investigation with respect to specialized tools designed to solve tasks from the bAbI dataset.

1 Introduction

With recent advances in *natural language processing* (NLP), systems for *information extraction* (IE) and *question answering* (QA) have become increasingly important for organizing and reasoning over natural language data. IE is concerned with identifying structured information from unstructured text. The structured representation is then represented in a machine-readable form suitable for automated querying. QA systems build upon these extracted representations to infer answers from narratives, documents, or collections of text. Traditional IE and QA systems have often relied on specialized pipelines that combine linguistic analysis, statistical learning, symbolic reasoning, rule generation, and domain-specific engineering. These pipelines are often high performing but hyper-specialized and break upon introducing a slight syntactic or semantic change to the input domain.

Many modern QA systems heavily rely on machine learning techniques. Also, *large language models* (LLMs) have quickly become central components of modern NLP systems. These models form the backbone for powerful assistant tools such as ChatGPT (OpenAI, 2022), Gemini (Anil et al., 2024), and Claude (Anthropic, 2023), that process and produce convincingly realistic natural language. As LLMs become increasingly capable, they can be viewed as a type of general-purpose machine learning system for inference over natural language input. This flexibility over the input domain makes LLMs a natural candidate for the design of generic IE and QA systems. However, it is still uncertain to what extent LLMs can retain factual and commonsense knowledge from the information they process. Many of the impressive claims about LLMs made on media, for example, pertain to more complex systems which are often proprietary and under-documented.

This study explores two key research questions:

Q1. To what extent can pretrained LLMs themselves, separated from larger chatbot or assistant platforms built around them, function as standalone reasoning and IE/QA systems?

Q2. Where do symbolic special-purpose IE methods and specialized neural QA systems stand relative to LLM-based approaches that reframe IE and QA tasks as language generation problems?

We investigate how LLMs function as a general-purpose IE and QA tool with respect to a subset of the reading comprehension tasks from the bAbI dataset by Facebook (Weston et al., 2015). The bAbI tasks provide synthetic narratives generated to test a systems ability to extract relevant information from text and answer questions involving temporal, spatial, and common sense reasoning. We contextualize this investigation with respect to specialized IE and QA systems designed specifically for the bAbI dataset such as AM+NG+NL MemNN (Weston et al., 2015), IRLR (Mitra and Baral, 2016), and TEXT2ALM (Lierler et al., 2024). Each of these specialized systems rely on explicitly engineered pipelines taking advantage of

bAbI's syntactically simple and algorithmic narratives. Unlike these specialized systems, the LLMs are not given any examples from the dataset (aside from any potential cross-contamination in the LLM training corpus), and instead are tested out-of-the-box (OTB) in order to assess pre-trained LLMs themselves as standalone reasoning tools (not the systems built on top of them). We test two types of OTB models, namely models that have only been pretrained on large corpora, and models that have additionally been fine-tuned for general purpose use off the shelf.

We turn each of the bAbI tasks into language generation problems of two kinds: one that mimics the bAbI entries verbatim, rephrasing its questions into declarative statements and another one that adds instructions in style of *zero-shot* learning. Our study illustrates, that although number of parameters is an important factor that contributes to models performance – the particular model family and implementation details may not be underestimated.

It is worth commenting on some related work. Researchers have explored LLM evaluation in a multitude of settings. See, for instance, papers by Rajani et al. (2019), Petroni et al. (2019), Basu et al. (2021). Rajani et al. (2019) investigate how human annotated datasets help LLMs learn commonsense and reasoning tasks. Petroni et al. (2019) test pretrained LLMs on relational knowledge, claiming that LLMs, in particular BERT-large, can recall knowledge at a competitive level with knowledge bases. Basu et al. (2021) introduce a way for LLMs to map natural language to knowledge using semantics of VerbNet. The HELM leaderboard (Liang et al., 2023) is an initiative that attempts to bring extra rigor to benchmarking across LLMs by putting together various rubrics for evaluations and introducing broad selection of datasets used for these. Elements of bAbI are among the datasets used within the leaderboard settings. While we applaud that initiative we find that the following comparison is appropriate here: *what this case study is to HELM* is *what a microscope is to an eye*. Here we zoom in on one particular benchmark and attempt to inspect it from various angles to gain as deep understanding as possible. After presenting our main findings we return to the discussion of HELM results specific to bAbI dataset.

We take a stand that evaluation of LLMs performance on a task has to be accompanied by the performance of specialized systems tackling the same task (unlike monolithic leaderboards). Such an evaluation properly positions LLMs – their capabilities and shortcomings – with respect to a task at hand. In this work, we perform this effort within the realm of IE/QA. We position a contextualization of LLMs evaluation by its specialized peers as essential in claiming the progress that LLMs take to what is commonly called general AI.

2 Background: LLMs and bAbI Dataset

LLMs are *auto-regressive models* trained to predict the most likely next word given a passage of text. This task is called language generation. Some pretrained LLMs are publicly available – for instance, OPT (Zhang et al., 2022), BLOOM (Scao et al., 2022), Llama-2 (Touvron et al., 2023), and Llama-3.1(Dubey et al., 2024). These models can be used as is, "out-of-the-box" (OTB), or trained on a specific task through fine-tuning. Some of these models are already fine-tuned for tasks beyond language generation (e.g, BLOOMZ, Llama-2 Chat, and Llama-3.1 Instruct). We call such models OTB-fine-tuned. In this study, we use the mentioned OTB and OTB-fine-tuned models.

When one refers to LLM training, it is common to denote two distinct processes, namely *pretraining* and *fine-tuning*. Pretraining introduced by Collobert et al. (2011) has been shown to be widely successful (Devlin et al., 2019), (Radford and Narasimhan, 2018), (Howard and Ruder, 2018). During pretraining, a model is trained on vast corpora of text for the sole purpose of next word or next token prediction. In fine-tuning, the earlier pretrained LLMs are trained to tackle a particular NLP task. In this way, it is interesting to explore the base capabilities of pretrained LLMs in an effort to better understand the base technology's strengths and limitations. Prompt-based learning or prompt engineering is a field of study exploring the ways to take advantage of LLMs' next word prediction by conditioning text generation task with information contained in the prompt. Introduced by Palatucci et al. (2009), *zero-shot learning*, in the context of LLMs,is a prompting method in which LLMs are given a high-level overview of a question at hand. Intuitively, this is meant to help the model answer various questions without additionally fine-tuning the model on particular tasks. Liu et al. (2023) explore the notion of a *prefix*, in which a prompt is transformed into a declarative sentence with a missing token or tokens for the LLM to fill in at the end of the sentence.

Task 1: Single Supporting Fact	Task 2: Two Supporting Facts	Task 3: Three Supporting Facts
1 John travelled to the hallway.	1 Mary got the milk there.	1 Mary got the football.
2 Mary journeyed to the bathroom.	2 John moved to the bedroom.	2 Mary discarded the football.
3 Where is John? hallway 1	3 Sandra went back to the kitchen.	3 Mary picked up the football.
4 Daniel went back to the bathroom.	4 Mary travelled to the hallway.	4 John went to the kitchen.
5 John moved to the bedroom.	5 Where is the milk? hallway 1 4	5 Mary moved to the office.
6 Where is Mary? bathroom 2	6 John got the football there.	6 Mary went to the hallway.
7 John went to the hallway.	7 John went to the hallway.	7 Where was the football before the hallway? office 3 6 5
8 Sandra journeyed to the kitchen.	8 Where is the football? hallway 6 7	8 Mary picked up the apple.
9 Where is Sandra? kitchen 8	9 John put down the football.	9 Sandra moved to the office.
10 Sandra travelled to the hallway.	10 Mary went to the garden.	10 Daniel moved to the kitchen.
11 John went to the garden.	11 Where is the football? hallway 9 7	11 Daniel moved to the hallway.
12 Where is Sandra? hallway 10	12 John went to the kitchen.	12 Mary travelled to the garden.
13 Sandra went back to the bathroom.	13 Sandra travelled to the hallway.	13 Mary travelled to the kitchen.
14 Sandra moved to the kitchen.	14 Where is the football? hallway 9 7	14 Daniel moved to the bathroom.
15 Where is Sandra? kitchen 14	...	...
1 Sandra travelled to the kitchen.	1 Mary journeyed to the bathroom.	1 Daniel picked up the football.

Table 1: Excerpts from bAbI Tasks 1–3 for reasoning with one, two, and three supporting facts.

Task 4: Two Argument Relations	Task 5: Three Argument Relations
1 The hallway is east of the bathroom.	1 Fred picked up the football there.
2 The bedroom is west of the bathroom.	2 Fred gave the football to Jeff.
3 What is the bathroom east of? bedroom 2	3 What did Fred give to Jeff? football 2
1 The bedroom is west of the kitchen.	4 Bill went back to the bathroom.
2 The hallway is west of the bedroom.	5 Jeff grabbed the milk there.
3 What is west of the kitchen? bedroom 1	6 Who gave the football to Jeff? Fred 2
1 The bathroom is north of the garden.	7 Jeff gave the football to Fred.
2 The hallway is north of the bathroom.	8 Fred handed the football to Jeff.
3 What is north of the garden? bathroom 1	...
1 The garden is east of the bathroom.	1 Mary moved to the hallway.

Table 2: Excerpts from bAbI Tasks 4 and 5 for reasoning over two and three argument relations.

Task	Avg Length	Min Length	Max Length	Avg-Sl Length
1	31	10	55	5.19
2	79	10	464	5.11
3	251	19	1120	4.94
4	14	14	14	7.00
5	104	11	490	5.24

Table 3: Basic statistics for bAbI tasks measured in number of words.

The bAbI Dataset In this work, we use the question answering (QA) benchmark designed by Weston et al. (2015), namely, the Facebook AI Research *bAbI* dataset. It is a collection of 20 tasks meant to measure natural language understanding of NLP systems. Mainly, it aims to assess how an NLP question answering system can answer questions via chaining facts together, performing simple induction and deduction. Tables 1 and 2 present a snippet of each considered bAbI task. The tasks are simple and easy to manipulate and evaluate, making them ideal for baseline assessments.

This study focuses on five bAbI tasks, namely, *Task 1: Single Supporting Fact*, *Task 2: Two Supporting Facts*, *Task 3: Three Supporting Facts*, *Task 4: Two Argument Relations*, and *Task 5: Three Argument Relations*. Tasks 1, 2, 3, and 5 are collections of stories in which characters move back and forth between six different locations. Task 4 is a collection of stories that describe how each of the locations are positioned with respect to each other. Each story is marked by line numbers on the left hand side. Questions within the story reference all previous lines in the current story. A new story begins when the line number resets to one. For example in Task 1 seen in Table 1, line 3 is the question "Where is John?" The reported answer is "hallway" with the corresponding supporting fact found on line 1 (sentence 1 of the respective story). In this manner, questions and answers are provided, along with references to how the answer was derived. The questions are focused on where a character or object carried by a character is. Intuitive to the names of the tasks, questions within the Single Supporting Fact task rely on a single sentence to reason about the answer. The Two and Three Supporting Facts tasks require references to two and three sentences, respectively. Tasks 4 and 5 only rely on one sentence, in which the question requires identifying the correct subject and object.

In this study, we define narratives as the collection of sentences leading up to a question within the same numbered story from one of the bAbI tasks. Table 3 presents some statistics about the resulting narratives from the considered tasks including the average (Avg), shortest (Min), and longest narrative lengths (Max) in number of words, and average sentence length (Avg-Sl) in number of words.

3 Evaluation Settings

In this section, we describe the settings of our experimental analysis. To begin with we present how we turn the bAbI dataset from a baseline benchmark for question answering systems to baseline benchmark for language generation models. We then provide details on the computing infrastructure that we utilized to carry out the experiments.

Given a bAbI task under consideration, for each question in its stories, we extract the relevant narrative and transform the question using regular expressions into a declarative sentence. Each of these sentences is in the form of a prefix, where the last word is expected to be predicted by an LLM. This transformation aims to maintain the core principle behind the task, while presenting the question in a manner which plays to the strength of LLMs' next word prediction.

Let us illustrate these ideas in the context of Task 1. Consider Table 1. The question in line 3 is transformed into prefix:

$$\text{John is in the ____} \tag{1}$$

The first narrative is constructed by extracting sentences 1 and 2: *John travelled to the hallway. Mary journeyed to the bathroom.* The second narrative consists of sentences 1, 2, 4, and 5.

In our experiments, we use two prompting styles. The first style prompts LLMs with narratives followed by prefixes (formed from questions). Let us call this *standard prompting*. The second prompting style adds context to the narrative/prefix pairs of the standard prompting. This context is composed of two parts: *instruction* and *context question*. Let us call this *instructional prompting*. The instructional prompt is composed of the instruction, the narrative, the context question, and the corresponding prefix.

Let us consider Task 1. The instruction for Task 1 always has the form: *The following is a narrative with characters moving to different locations. Assume the character's most recent reference refers to their current location.* The context question for Task 1 is always in the form: *Using the locations from the narrative, where is [character] currently located?*

To illustrate instructional prompting in its complete form, let us consider the narrative composed of sentences 1 and 2 in Table 1 and the question in line 3. In this example, the prompt is composed of the instruction, the sentences in line 1 and 2, the context question, and the prefix in (1):

> The following is a narrative with characters moving to different locations. Assume the character's most recent reference refers to their current location. John travelled to the hallway. Mary journeyed to the bathroom. Using the locations from the narrative, where is John currently located? John is in the ____

The instructional prompting follows the ideas behind *zero-shot* learning, aiming to help an LLM process the narrative by providing a high-level overview of the question in the form of instructions. The question before the prefix aims to improve the model's reasoning, telling the model explicitly to use the locations from the narrative, as well as including words such as *currently* for Tasks 1. Instructional prompts for other tasks are designed in the same spirit.

Experimental Infrastructure To carry out the evaluation work, we utilized the Hugging Face transformers library (Wolf et al., 2020) to access 18 LLMs considered in this study. All considered models belong to one of the four families: BLOOM, OPT, Llama-1, and Llama-3.1. The members of these families vary significantly in their computational and space requirements. The largest model used in the experiments is BLOOM 176b. To collect the data on BLOOM 176b for Tasks 1 and 2 we utilized the Hugging Face inference API. This was the only model we tested, which was hosted on a free access API. We were unable to collect data for Task 3-5 on this model as the API calls limited the input to 1000 tokens per query and were disabled by provide with time. All other experiments in this study were performed through a high performance computing center. We grouped each considered task/model pair into a batch. For smaller models, namely, BLOOM (/Z) 3b (and less), all variants of OPT, Llama-2 (/Chat) 7b, and Llama-3.1 (/Instruct) 8b, we ran the tasks using a Tesla V100 32GB on GPU nodes, resulting in inference times

of about 30 minutes or less for each model/task pair. Due to the limitations on GPU nodes available, we ran Llama-2 (/Chat) 70b and Llama-3.1 (/Instruct) 70b, on CPU nodes with Intel Xeon Gold 6348 2.6GHz 56 core processors and up to 2000GB of RAM. This resulted in substantially longer inference times than running on GPU. For example, Llama-3.1 70b requires about 300GB of RAM and over five days to inference Task 3. BLOOM 176b requires about 700GB of RAM and over a week to inference Task 3; because the computing center caps processing time to a week per batch job, we were unable to collect the data for this model/task using the specified process.

4 Experiments and their Interpretation

Now we discuss the results of the experiments on the 18 LLMs across each of the bAbI tasks considered as seen in Table 4. The 18 LLMs that we tested come from four families: BLOOM, OPT, Llama-2, and Llama-3.1. Within three of these families there are variants that were fine-tuned prior to their releases, namely, BLOOMZ, Llama-2 Chat, Llama-3.1 Instruct. In the table we use letter B to denote BLOOM; letters Bz to denote BLOOMZ; letter L to denote Llama. For each model, the table reports the accuracy for both the standard prompting (columns titled *Stand*) and instructional prompting (columns titled *Inst*). Accuracy is calculated by comparing the model's predicted location with the location given in the bAbI dataset. A predicted location that matched the answer exactly is scored with 1, otherwise 0. The LLMs were restricted to generating two tokens, as some models considered compound words like "hallway" to be two tokens: "hall" and "way". For LLMs that considered compound words to be one token, the second token was often a period or a new line token which was disregarded. In this way, {"hallway", "."}, {"hallway", "\n "}, and {"hall", "way"} are all acceptable outputs for the target answer "hallway". The score is computed from a single run. The highest score for a particular model-family across the variants of its sizes is in bold (in the table each family is separated by double lines, we placed BLOOM 176b into its own family as we did not have data on BLOOMZ 176b). Per each model and task, a box identifies higher accuracy between standard and instructional prompting.

The results show how OTB LLMs such as OPT, BLOOM, and Llama overall struggle to perform reading comprehension and logical deduction the considered bAbI Tasks. While the 70b variants of Llama score proficiently on the easier tasks, all models regardless of parameter size see a substantial drop off in accuracy on the tasks with increased complexity. Both the OTB and OTB-fine-tuned versions of Llama 3.1 saw a general improvement in performance when using instructions while prompting on almost all tasks. For the remaining models, there was no clear advantage in using the instructional prompting style for Tasks 1-3.

In Tasks 4 and 5, there is a clear improvement in performance when using instructions while prompting across all models, with Llama 3.1 70b seeing a 71.45% improvement in Task 4 when given instructions. The models generally perform better on tasks 4 and 5 than on task 3. While Task 4 requires reasoning over the relationship between entities rather than tracking changing facts, it only requires one supporting fact to answer any particular question. The results suggest that pretrained LLMs encode strong statistical representations of natural-language argument structure and semantic relations when only requiring one supporting fact; this makes sense when taking into account the large training sizes of LLMs and architectural differences between LLMs and other machine learning models. LLMs have an incredibly large training size, with models such as Llama 3.1 having seen over 15 trillion tokens.

Task 3 stresses explicit temporal state tracking requiring three steps of reasoning over evolving object locations; in contrast, answering questions in Tasks 1, 4, and 5 only requires one supporting fact. With these findings in mind, LLMs appear substantially better at leveraging latent linguistic knowledge than at maintaining explicit multi-step symbolic state representations across long reasoning chains. These findings coincide with the results from Olausson et al. in which the authors tested how LLMs, including GPT-4, can be used to reason over questions from the FOLIO (Han et al., 2024) and Proofwriter (Tafjord et al., 2021) datasets. The authors found that when answering a question required two or more logical steps, GPT-4's accuracy dropped to roughly the random baseline with naive prompting.

When comparing the base and instruct-fine tuned counterparts, we see that in tasks 1 through 3, model performance increases across the board for Llama 3.1 8b and 70b respectively. Tasks 1 and 2 in particular show the greatest improvements, with

	Task 1		Task 2		Task 3		Task 4		Task 5	
Model	**Std**	**Inst**	**Std**	**Inst**	**Std**	**Inst**	**Std**	**Inst**	**Std**	**Inst**
B 560m	0.295	0.321	0.202	0.274	0.176	0.191	0.298	0.492	0.493	0.511
B 1b1	0.364	0.330	0.223	0.221	0.158	0.155	0.220	0.379	0.315	0.508
B 3b	**0.569**	0.433	0.270	0.236	0.220	0.210	0.194	0.491	0.435	0.646
Bz 560m	0.428	0.353	0.268	0.329	0.225	0.213	0.434	0.497	**0.566**	0.586
Bz 1b1	0.379	0.42	0.226	0.253	0.166	0.164	0.281	0.481	0.442	0.570
Bz 3b	0.537	**0.535**	**0.342**	**0.391**	**0.289**	**0.282**	**0.454**	**0.568**	0.472	**0.656**
B 176b	0.536	**0.647**	0.290	**0.454**	–	–	–	–	–	–
OPT 350m	0.381	0.380	0.196	0.318	0.209	0.179	0.181	0.380	**0.451**	0.488
OPT 1.3b	0.550	0.477	**0.254**	0.323	0.212	0.222	**0.228**	0.353	0.373	**0.535**
OPT 2.7b	**0.584**	**0.542**	0.247	**0.326**	**0.246**	**0.239**	0.210	**0.404**	0.365	0.484
L-2 7b	0.471	0.551	0.316	0.313	0.230	0.262	0.261	0.452	0.524	0.643
L-2 70b	0.701	0.633	**0.590**	0.440	**0.381**	**0.323**	**0.332**	0.713	**0.694**	**0.717**
L-2 C 7b	0.371	0.819	0.284	0.493	0.276	0.262	0.166	0.561	0.511	0.670
L-2 C 70b	**0.721**	**0.951**	0.507	**0.622**	0.344	0.274	0.284	**0.811**	0.571	0.697
L-3.1 8b	0.431	0.420	0.311	0.337	0.268	0.282	0.360	0.568	0.494	0.620
L-3.1 70b	0.787	0.952	0.630	0.712	**0.398**	0.475	0.209	**0.732**	**0.720**	**0.834**
L-3.1 I 8b	0.743	0.939	0.405	0.505	0.304	0.331	0.366	0.600	0.480	0.658
L-3.1 I 70b	**0.944**	**1.000**	**0.729**	**0.804**	0.306	**0.480**	**0.376**	0.716	0.687	0.732

Table 4: LLM accuracy on standard and instructional prompting stemming from bAbI tasks.

Llama 3.1 8b performance increasing by 55.88% and 33.27% and Llama 3.1 70b increasing by 6% and 11% respectively. On tasks 4 and 5, however, Llama 3.1 70b Instruct performs marginally worse than the base counterpart. Model performance between BLOOM and BLOOMZ and Llama 2 and Llama 2 Chat fluctuates from task to task.

Figure 1 explores the relationship between the length of a narrative and the accuracy in Task 3. This task has the longest narratives and lowest scores across all models. Figure 1 presents the results on four groups of the narratives in accordance with their lengths: group 1 contains all narratives whose length ranges over 0-104 words; groups 2, 3, and 4 contain the narratives ranging over 105-263, 264-421, and 422 or more words, respectively. Each group is composed of 131 narratives, 478 narratives, 239 narratives, and 153 narratives respectively. This figure shows a clear drop-off in accuracy in groups 2 and 3 with a trend to increase accuracy in group 4. This trend upwards implies that narrative length is not the only factor decreasing performance. Additionally, each LLM in this study has a sufficient context window to encapsulate the longest prompt with approximately 1,300 words. The context windows for Llama 3.1, Llama 2, BLOOM, and OPT models are 128,000, 4,096, 2,048, and again 2,048 tokens, respectively.

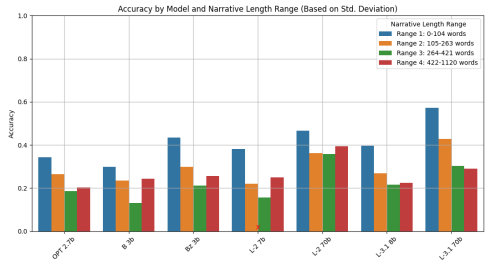

Figure 1: Accuracy of models by narrative length.

5 Comparison with Specialized IE Systems and HELM Leaderboard

We compare our findings with three specialized Information Extraction (IE) systems designed to solve instances from the bAbI dataset. The first is the machine learning approach AM+NG+NL MemNN designed by the Facebook research group when developing the bAbI dataset as a back-in-the-date state-of-the-art benchmark comparison in Weston et al. In that work, the authors compared eight different machine learning approaches on bAbI tasks, in which AM+NG+NL MemNN performed the best almost across the board. To create the AM+NG+NL MemNN tool, the elements of the bAbI dataset were used during training. In fact, each of the tasks by bAbI comes with two sets of narratives one for training and another for testing.

bAbI Task	AM+NG+NL MemNN	IRLR	TEXT2ALM	OPT 350m	L-3.1 70b I
1-Single Sup. Facts	1.00	1.00	1.00	0.38	1.00
2-Two Sup. Facts	1.00	1.00	1.00	0.32	0.80
3-Three Sup. Facts	1.00	1.00	1.00	0.18	0.48
4-Two Arg. Rels.	1.00	1.00	–	0.38	0.72
5-Three Arg. Rels.	0.98	1.00	0.22	0.49	0.73

Table 5: Accuracy on selected bAbI tasks.

	Training Questions			Pretraining Tokens	
bAbI Task	AM+NG+NL MemNN	IRLR	TEXT2ALM	OPT 350m	L-3.1 70b I
1-Single Sup. Facts	250	1000	100	180B	15T+
2-Two Sup. Facts	500	1000	100	180B	15T+
3-Three Sup. Facts	500	1000	100	180B	15T+
4-Two Arg. Rels.	500	1000	–	180B	15T+
5-Three Arg. Rels.	1000	1000	100	180B	15T+

Table 6: Comparison of training question counts for specialized QA systems and approximate pretraining token counts for LLMs.

Table 7: Comparison of other tasks using HELM submitted scores

		Task 3			Task 15			Task all
Model	Size	Std	Inst	5-Ex	Std	Inst	5-Ex	5-Ex
Llama-2	7b	0.230	0.262	0.217	0.455	0.425	0.402	0.550
Llama-2	70b	0.381	0.323	0.366	0.367	0.226	0.747	0.705
BLOOM	176b	–	–	0.320	–	–	0.339	0.447

The second column in Table 6 provides information on the portion of the training data seen by this tool. The training size is measured in the number of questions occurring within a text file comprised of bAbI entries as illustrated in Tables 1 and 2. The second system is the Inductive Rule Learning and Reasoning (IRLR) approach by Mitra and Baral. The authors used inductive machine learning techniques to produce a tool comprised of a statistical inference layer, formal reasoning layer using answer set programming, and a translation layer encoding natural language sentences into event calculus. The production of this tool required training examples provided by bAbI. Mitra and Baral report that IRLR excels at all of the bAbI tasks except for one. The third system is TEXT2ALM by Lierler et al., in which bAbI narratives are translated into the action language $\mathcal{ALM}$ which encodes facts about a given narratives. The system utilizes a number of NLP tools to extract actions and entities from these narratives. The verbs indicating a particular action are mapped with a commonsense background knowledge that augments specified actions so that formal reasoning can be performed to extract answers from the $\mathcal{ALM}$ descriptions encoding narratives. The developers of the TEXT2ALM system used some examples from training part of the bAbI dataset to guide its decisions on what kind of verbs are supplied with commonsense knowledge. Otherwise, commonsense knowledge stems from generic Core-ALM Library (INCLEZAN, 2016), whose creation has nothing to do with bAbI narratives.

Table 5 reports the accuracies of the specialized IE systems mentioned in the preceding paragraph. We augment these results with two columns obtained from Table 4, namely, the accuracies of LLMs OPT 350m and Llama 3.1 70b Instruct. We chose the former LLM for the fact that this model is of the smallest size that participated in this study. The latter is the model that was among the top performers in our experiments. System TEXT2ALM was not designed to be able to tackle narratives of the kind present in Task 4. System IRLR outperforms all other models, reporting 100% accuracy across each of the tasks. AM+NG+NL MemNN reports 100% accuracy on tasks 1-4 with a 98% accuracy on task 5. TEXT2ALM report 100% accuracy on Tasks 1 - 3 and 22% on task 5 respectively. An

interesting point for comparison which highlights the brittle nature of specialized tools is in the case of Task 5 on the TEXT2ALM system. Task 5 contained several entries of the form: "Entity1 *handed the* Object *to* Entity2". The semantic role labeling system that TEXT2ALM used to annotate input sentences incorrectly processed sentences of this type. This annotation error effectively poisoned the rest of the TEXT2ALM pipeline for these cases, resulting in incorrect interpretations of the narratives. Table 6 presents the number of questions in training sets used by each of the reported specialized approaches in considered tasks, alongside the training size in number of tokens seen during pretraining for OPT 350m and Llama 3.1 70b Instruct. This contextualizes Table 5, as IRLR saw ten times more examples than TEXT2ALM and up to four times more than AM+NG+NL MemNN. Still, these numbers pale in comparison to the training size of LLMs; while the LLMs (presumably) did not see any explicit example from the bAbI dataset, they have seen billions to trillions of tokens during pretraining.

It is also important to consider the computational cost required to deploy LLM-based approaches in practice. Systems such as TEXT2ALM demonstrate strong task-specific reasoning on a lightweight infrastructure that can be run on standard consumer-grade hardware with short inference times. In contrast, as discussed in Section 3, larger LLMs require high performance compute centers with hundreds of gigabytes of RAM, and in some cases, several days to complete a single task.

Comparison With HELM The HELM leaderboard (Liang et al., 2023) offers benchmarking data across numerous datasets and LLMs. It reports some results for the bAbI dataset for a multitude of models including, for instance, models in Llama and GPT family. The leaderboard covers Tasks 3, 15, and 19 individually, as well as the tasks conglomerated together as a single metric. The methodology used in constructing a prompt to access LLMs is different from this work; HELM prompts are constructed by including five sample narratives from the training set together with the questions and answers. Then, the LLM is given a sixth narrative followed by a question for evaluation. The LLM is not provided instructions, and the question is not transformed into a declarative statement. HELM restricts LLM generation to a maximum of five new tokens.

For comparison, we extended evaluation to Task 15 using the same methodology as Tasks 1-5. Table 7 compares the results found in this work with HELM for Task 3, Task 15, and Task All. The column labeled 5-Ex reports HELM's findings. We present results for three models – Llama 2 7b; Llama 2 70b; BLOOM 176b – these are the only models that our experiments overlapped in. It should be noted that Llama 2 70b is among the top performers for each bAbI task reported. HELM results for Task 3 provide a level of comparability with our approach; however, the results for Task 15 are severely different in the case of Llama-2 70b, highlighting the model's sensitivity to particular prompting methods.

6 Conclusions

This project showcases how OTB and OTB-fine-tuned models perform on five bAbI tasks. In reference to **Q1**, the main findings are that (i) accuracy drops across the board when increasing the number of required supporting facts, (ii) most models saw no sizable benefit from instructional prompting, and (iii) increasing parameter size generally leads to better results. We also report that LLMs generally perform better at Tasks 1, 2, 4, and 5 than Task 3. This hints that while LLMs struggle to extract chains of supporting facts, they have some latent linguistic knowledge from pretraining that helps identify argument relations in natural language. This reiterates that OTB LLMs are designed primarily for the task of realistic natural language generation, not advanced NLP tasks. We also observe that the Llama family of models, in particular the fine-tuned Llama 2 Chat and Llama 3.1 Instruct models, best take advantage of the instructional prompting. This illustrates the importance of fine-tuning and tailoring prompts for specific models and problems.

The results in reference to **Q2** help contextualize LLMs alongside specialized symbolic and neural approaches designed specifically for the bAbI dataset. While larger instruction fine-tuned models such as Llama 3.1 70B Instruct achieved competitive performance on several tasks, specialized IE systems such as AM+NG+NL MemNN, IRLR, and TEXT2ALM continued to outperform the evaluated LLMs overall. System TEXT2ALM demonstrates that strong performance can be achieved through explicitly engineered reasoning pipelines using comparatively small amounts of task-specific training data.

Limitations

Data Contamination in LLMs. Data contamination occurs when testing data appears in a model's training data. There is an ever-present risk of data contamination when working with LLMs, in which identifying and mitigating its impact in benchmarking and evaluations is a current open research question (Fu et al., 2025). LLMs are pretrained on extensively large corpora of text containing upwards of trillions of tokens from countless sources which are generally not revealed to the public. One potential way to help against data contamination is to modify the testing data, altering the narratives and questions like in the bAbI+ variant introduced in (Lierler et al., 2024).

Methodology. Aside from the processing limitations outlined in the *Experimental Infrastructure* subsection, this work has three main limitations: the models used, the tasks used, and the bAbI dataset itself. For the models used, there are only two main large models, BLOOM 176b, and Llama 70b. While the focus on publicly available OTB models is important to understand the performance of the easily accessible baseline technology, it is also important to note other state of the art models. We aim to address this limitation in future work, expanding the project to encompass GPT family evaluation. For the second limitation, this work currently only explores the first five bAbI tasks. While these tasks are convenient to explore reading comprehension through the task of language generation, each of the tasks are similar in principle. We aim to address this limitation in future work by extending the evaluations to the remaining bAbI tasks, which will help to provide a more holistic view of models' reading comprehension and problem solving abilities. The final limitation is in the bAbI dataset itself, as it is a convenient collection of "toy" tasks. This dataset is a useful tool to gauge performance, but may overestimate the capacities of models when real world problems are considered.

References

Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, and Orhan Firat ... Oriol Vinyals. 2024. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Anthropic. 2023. Introducing claude.

Kinjal Basu, Sarat Chandra Varanasi, Farhad Shakerin, Joaquín Arias, and Gopal Gupta. 2021. Knowledge-driven natural language understanding of english text and its applications. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12554–12563. AAAI Press.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and ... Kevin Stone. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Yujuan Fu, Ozlem Uzuner, Meliha Yetisgen, and Fei Xia. 2025. Does data contamination detection work (well) for LLMs? a survey and evaluation on detection assumptions. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5250–5271, Albuquerque, New Mexico. Association for Computational Linguistics.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alex Wardle-Solano, Hannah Szabo, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, and 16 others. 2024. Folio: Natural language reasoning with first-order logic. *Preprint*, arXiv:2209.00840.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 328–339. Association for Computational Linguistics.

DANIELA INCLEZAN. 2016. Corealmlib: An *ALM* library translated from the component library. *Theory and Practice of Logic Programming*, 16(5–6):800–816.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew A. Hudson, and 31 others. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*. Featured Certification, Expert Certification, Outstanding Certification.

Yuliya Lierler, Gang Ling, and Craig Olson. 2024. Information extraction tool text2alm: From narratives to action language system descriptions and query answering. *AI Commun.*, 37(1):53–81.

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):195:1–195:35.

Arindam Mitra and Chitta Baral. 2016. Addressing a question answering challenge by combining statistical methods with inductive rule learning and reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1).

Theo Olausson, Alex Gu, Benjamin Lipkin, Cedegao E. Zhang, Armando Solar-Lezama, Joshua B. Tenenbaum, and Roger Levy. 2023. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 5153–5176. Association for Computational Linguistics.

OpenAI. 2022. Openai: Introducing chatgpt.

Mark Palatucci, Dean Pomerleau, Geoffrey E. Hinton, and Tom M. Mitchell. 2009. Zero-shot learning with semantic output codes. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*, pages 1410–1418. Curran Associates, Inc.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4932–4942. Association for Computational Linguistics.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilic, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, and ... David Ifeoluwa Adelani. 2022. BLOOM: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100.

Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. 2021. Proofwriter: Generating implications, proofs, and abductive statements over natural language. *Preprint*, arXiv:2012.13048.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and ... Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards AI-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and Alexander ... Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: open pre-trained transformer language models. *CoRR*, abs/2205.01068.