Learning Semantic Structure through First-Order-Logic Translation

Akshay Chaturvedi, Nicholas Asher

Learning Semantic Structure through First-Order-Logic Translation

Akshay Chaturvedi[†], **Nicholas Asher**[†‡]
[†]IRIT, [‡]CNRS,
Toulouse, France

Abstract

In this paper, we study whether transformer-based language models can extract predicate argument structure from simple sentences. We firstly show that language models sometimes confuse which predicates apply to which objects. To mitigate this, we explore two tasks: question answering (Q/A), and first order logic (FOL) translation, and two regimes, prompting and finetuning. In FOL translation, we finetune several large language models on synthetic datasets designed to gauge their generalization abilities. For Q/A, we finetune encoder models like BERT and RoBERTa and use prompting for LLMs. The results show that FOL translation for LLMs is better suited to learn predicate argument structure.

1 Introduction

Transformer-based language models (LMs) (Vaswani et al., 2017; Bubeck et al., 2023) have attracted enormous interest because of their language generating capacities and prowess in many NLP tasks. We are interested in LMs and their ability to exploit semantic structure for both grasping linguistic meaning and inference. In this paper, we concentrate on an elementary building block of semantic structure, i.e., predicate argument structure. For example, predicate argument structure determines that the predicate *blue* applies to *house* and not *car* in the sentence:

(1) there was a red car in front of a blue house.

Predicate argument structure also determines which arguments fill which places in two place predicates like *in front of*; in our example, it is the car that is in front of the house, not the house in front of the car.

Recovering predicate argument structure is crucial to capturing and reasoning about the meaning of natural language sentences. If an LM mixes up which object has which property in a premise, it is guaranteed to make errors in reasoning. While semantic structure eventually involves the scopes of operators and quantifiers and verbal modification using tense or adverbial phrases, in this paper, we concentrate simply on capturing the predicate argument structure of basic predicates applied to object denoting nouns like *car*. Our experiments are restricted to simple sentences involving two indefinite noun phrases (NPs) with one or more modifying predicates for each NP. An example of one such sentence is (1), containing two noun phrases *a red car* and *a blue house*. The motivation behind using such sentences is to study whether current models are able to capture predicate argument structure in relatively simple scenarios. While being able to capture semantic structure in such cases doesn't necessitate generalization to actual linguistic data, it is an important precursor towards this goal. Further, synthetic examples also help pinpoint the kinds of difficulties faced by the current models.

We investigate two approaches for analysing LLMs' ability to capture predicate argument structure: question answering (Q/A), and translation into a first-order logical (FOL) form. For Q/A, we prompt LLMs to predict a yes/no answer, where for FOL translation we finetune LLM for the task. We also look at the performance of smaller encoder models like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) using finetuning on Q/A with various predicate argument datasets. The main reason behind also focusing on smaller encoder models is to extend the study of Chaturvedi et al. (2024) which looked at these models' performance on a synthetic dataset of 5 simple templates. Their dataset consists of sentences with two objects having two different colors. This work extends the aforementioned synthetic dataset by incorporating *more properties* for the two objects, *sentence paraphrasing* and *negation*.

We find that both encoder models and LLMs are able to learn predicate argument structure for simple sentences with just one predicate for each object mentioned. However, encoder models are unable to generalize to more complex sentences involving more predicates for the two objects mentioned, and even LLMs fail to fully master the predicate argument structure of such sentences.

For smaller encoder models, finetuning results in a much higher accuracy when testing on similar patterns to training data but a low accuracy on dissimilar patterns, as a result of overfitting. For LLMs, the FOL translation gives better results in comparison with Q/A prompting, showing that LLMs can generalize from the simple predicate argument structure to more complex sentences. We find that translation approach also has the important advantage of showing when models add hallucinated content, which we argue a Q/A method cannot do.

In what follows, Section 2 provides motivation and surveys relevant prior work. Section 3 gives some preliminaries for our study, while Section 4 describes our synthetic datasets. Section 5 gives the results of our experiments. Finally, in Section 6, we discuss conclusions and potential future work.

2 Motivation and Previous Work

Chaturvedi et al. (2024) conduct experiments concerning predicate argument structure on encoder models like BERT and RoBERTa (Devlin et al., 2019; Liu et al., 2019). As mentioned in Section 1, they construct a dataset of sentences involving two objects with two different colors. They use 5 different schemas, two of which are reproduced below in (2-a) and (2-b), where *col1* and *col2* denote two distinct colors.

(2) a. A *col1* car was standing in front of a *col2* house.
b. They played with a *col1* ball and *col2* bat.

Their synthetic dataset contains 1040 questions (520 “yes” and “no" questions each) on contexts using schemas (2-a) or (2-b) with different color combinations. For each schema, they provide two questions that are semantically equivalent given the context (e.g. “Was the car *col1*?" and “Was there a *col1* car?"). They then test whether the models, finetuned on the CoQA dataset (Reddy et al., 2019), could correctly associate the properties with their relevant bearers in a simple question answering task. They find that all the models except RoBERTa-large (Liu et al., 2019) achieve low accuracy on this simple Q/A test; the encoder models also behave differently and rather strangely with regard to the original and modified questions. We provide their results in Table 6 in Appendix for sake of completeness. We extend this work with in depth analyses and experiments on a range of datasets.

Feng and Steinhardt (2023) investigate the binding problem for entity referring expressions (typically proper names) linked by predicates, though they did not investigate the predicate argument binding problem *per se* and certainly not in its full generality. They examine representations in a transformer after attention and linear normalization layers and argue that binding is done though a particular identification vector. In this paper, we look at the binding problem between simple properties conveyed by adjectives and their bearers typically introduced by indefinite noun phrases. Our preliminary experiments indicate that even substantial LLMs do not completely solve this problem.

It is known that LMs can learn features of abstract syntactic representations of natural language sentences like long distance dependencies involving subject verb argument agreement (Linzen et al., 2016; Goldberg, 2019) and the rarer object past participle agreement in languages like French (Li et al., 2023). Lakretz et al. (2022) show LMs have near perfect performance on short embedded syntactic dependencies but fail on longer distance embedded dependencies. We are interested in whether LMs can learn the mapping from syntax to semantic representations, of which predicate argument structure is the basic building block.

Dehghani et al. (2019) points out the problems of generalization for transformer models. Olausson et al. (2023) integrate LLMs with a theorem prover for natural language inference. They first prompt LLMs to translate text to FOL. The resultant FOL is passed to a theorem prover in order to predict an output. In this work, we look at two approaches to extract predicate argument structure from simple sentences: Question-Answering, and FOL translation. For question answering, we look at finetuning for encoder models and prompting for LLMs. Whereas, for FOL translation, we look at finetuning of LLMs.

Model	Org-Acc	Mod-Acc
Mistral-7B	97.1 (52.1)	92.5 (42.5)
Llama-2-7B	74.2 (55.8)	80.2 (34.4)
Llama-2-13B	99.0 (50.1)	93.3 (45.6)
Llama-3-8B	81.3 (65.6)	87.1 (58.3)

Table 1: Effect of question paraphrasing on the synthetic dataset of Chaturvedi et al. (2024) with different LLMs. Questions of type "Was the X *col1*?" are referred to as original questions (org) and question of type "Was there a *col1* X?" is referred to as modified questions (Mod). The number in brackets denote percentage of cases where the model predicted "no" as the answer.

3 Preliminaries

As mentioned in Section 2, we are interested to see whether the models can learn, through finetuning or prompting, the structure of a semantic representation. Given the successes of the models with learning syntactic structure, it is reasonable to assume a good level of performance. To this end, we first check whether the results from Chaturvedi et al. (2024) were just a limitation of smaller encoder models, like BERT and RoBERTa. We ran the same experiment on large language models (LLMs) in the Mistral (Jiang et al., 2023), Llama-2 (Touvron et al., 2023) and Llama-3 (Dubey et al., 2024) families.

The results are shown in Table 1. Given this table for large language models and Table 6 of Chaturvedi et al. (2024) for smaller encoder models in the Appendix, we hypothesize that model's training and finetuning on a generic question answering task does not force the model to recover the predicate argument structure of the context.

Since models can not master recovering even a simple predicate argument structure, we need to ask what about a string makes this recovery difficult. Formal linguistics provides a map μ from sentences to logical form and predicate argument structure either via an intermediate stage of syntactic structure or directly as in some forms of categorical grammar (Steedman, 1996). To recover predicate argument structure of a sentence, a model will have to learn an algorithm that given an input string has the same output as μ. By examining μ, we can pinpoint areas of difficulty for learning predicate argument structure.

The predicate argument structures of the sentences used in this work can be expressed with a simple, first order logical formula, consisting of a conjunction of positive atomic formulae of the form $\phi(\alpha_1)$ or $\psi(\alpha_1, \alpha_2)$ where α_i is a constant or a variable representing an individual object. Proper names introduce constants while indefinite noun phrases like *a car* introduce an existentially quantified variable. For instance, (1) has the logical form $\exists x \exists y(Red(x) \wedge Car(x) \wedge Blue(y) \wedge House(y) \wedge Infrontof(x, y))$.

Given our input strings and target predicate argument structures, there are two kinds of difficulty an algorithm must solve to find the predicate argument structure. The first has to do with the complexity of a noun phrase (NP) with multiple modifiers. In languages like English or French, a complex NP like *an old, green dirty car* is syntactically realized in terms of "modifier depth" and has the syntactic structure $[old[green[dirty[car]_n]_{np}]_{np}]_{np}]$. To get the right predicate argument structure with sentences containing such NPs, the model has to "unwind" the syntactic tree applying each predicate to the variable that is the argument of the predicate introduced by the noun. The ability to generalize from simple modification to more complex modification in a general way requires an ability to learn recursion. Without recursion, a model may learn different patterns for combinations of adjectives. However, this approach would require training on most (if not all) possible combinations of adjectives.

The other source of difficulty for learning predicate argument structure are long distance dependencies. Syntactic realizations of these involve for example relative clauses as in:

(3) the car that was next to the green house was red.

The predicate *red* in (3) is not close to its bearer. Such long distance dependencies have various syntactic representations (e.g. trace binding after movement in theories that allow movement) or require type raising in categorical grammars, which rewrites a function to take an argument of different type (Steedman, 1996), to guide μ to link the predicate with its argument.

We look at both kinds of difficulty by investigating LM behavior on a variety of synthetic datasets. These datasets enable us to narrow down what algorithms LMs use to capture predicate argument structure given these two dimensions of complexity. For instance, it is possible for a model to do well say on a Q/A dataset by finding

"short cut" algorithms instead of the true predicate argument structure. For instance, the model might guess the right answer to a question like *is there a red car?* for a sentence like (1) simply by computing the shortest distance between a color term and *car*, in the context. But this will fail for examples like (3).

We use two tests to determine whether the model has correctly grasped the predicate argument structure of the context. The first is a Q/A task that asks about which objects in the context have which properties. The answers to the questions should completely determine the logical form of the context. It turns out, as we show in Section 5, that this is difficult to do. The second way is for the model to produce the logical form directly. For this task, we give the model a sentence from our synthetic dataset as input and train it to output a correct logical form for that sentence in first order logic.

A complicating factor in the design of the Q/A experiments is what form the questions in the Q/A task should take. Chaturvedi et al. (2024) show that encoder models are sensitive to different formulations of a question in Q/A tasks. They behave quite differently when answering questions that are semantically equivalent given the input context, as we can see by comparing Org-Acc and Mod-Acc in Table 6, in the Appendix, for encoder models. We observe the same behavior, althought to a lesser extent, in LLMs as well, as shown in Table 1.

4 Datasets and Models

To probe a model's grasp of predicate argument structure in more detail, we develop several synthetic datasets of increasing complexity. The original dataset of Chaturvedi et al. (2024) consists of 5 templates, each containing two objects of different color. For each template, there were two semantically equivalent questions: original question of the type "Was the obj col1?", and modified question of the type "Was there a col1 obj?" where col1 and obj refer to color and object respectively. All the 5 templates have the objects and the corresponding color next to each other (e.g. "The red car was in front of the blue house"). As a result of this, we observed that when a model is trained on one of the templates, it learns to generalize to other templates as well. To counteract this, we extend the original dataset to 25 templates containing more complex templates with several long distance predicate argument structures as in (3) but also *Red was not the color of the car but of the house*. We refer to this dataset as $D_{1,1}$. We list down all the 25 templates along with their FOL translation in Table 7 of Appendix. Apart from $D_{1,1}$, we also modify the 5 original templates to construct more complex datasets, i.e., $D_{2,1}$, $D_{2,2}$, $D_{3,1}$, $D_{3,2}$, $D_{3,3}$; where $D_{i,j}$ refers to the two objects having i and j properties each. We use all these datasets for Q/A and FOL translation task. We do not add templates with complex long distance dependencies to these datasets, as our results indicate that this setting would be too difficult for LLMs to master. We also construct an additional dataset for question answering, D_{and}, where we ask "and" questions dataset such as "Was there a blue car and a red house?" for the 25 templates of $D_{1,1}$. All the synthetic datasets have equal number of *yes* and *no* questions.

Along with our synthetic datasets, we also work with the FOLIO dataset (Han et al., 2024). This dataset is an NLI dataset but also provides the first-order logic form for premises and hypotheses. We make use of these FOL forms along with a portion of our synthetic dataset to finetune LLMs to generate first order logic from simple sentences. We experiment with four LLMs, namely, Llama-2-7b, Llama-2-13b (Touvron et al., 2023), Mistral-7b (Jiang et al., 2023), and Llama-3-8b (Dubey et al., 2024). For finetuning these models, we use the QLoRA method (Dettmers et al., 2023). All the models are finetuned for 10 epochs. We provide details of computing infrastructure and hyperparameters in Table 8 of the Appendix. For question-answering prompting experiments, we give the prompt in the format "The blue car was standing in front of a red house.$\backslash n\backslash n$Q: Was there a red car?$\backslash n$A:" where $\backslash n$ is the newline character. We observe that adding in-context examples in the prompt does not lead to any improvement in terms of accuracy. This might be because we are prompting LLMs for simple yes/no questions. For encoder models, we use the CoQA finetuned *base* and *large* variants of BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) from Chaturvedi et al. (2024). The suite of synthetic datasets along with the best performing LLM finetuned for FOL translation are available here [1].

[1] https://huggingface.co/akshay107/nl-to-fol

Model	CoQA	FT
BERT-base	56.8	99.1
BERT-large	72.2	99.8
RoBERTa-base	57.0	99.7
RoBERTa-large	84.6	99.0

Table 2: Question-Answer Accuracy for different models on the $D_{1,1}$ dataset. CoQA refers to models finetuned only on CoQA and FT refers to further finetuning on the window/door templates for org question type of $D_{1,1}$. For FT, the score is on the test set of $D_{1,1}$.

Model	D_{and}	$D_{2,2}$	$D_{3,3}$
BERT-base	50.1/ 50.0	55.2 / 52.3	54.2/ 51.3
BERT-large	58.4 / 52.1	74.6 / 56.8	69.4 /50.0
RoBERTa-base	50.0 / 50.6	62.7 / 62.2	65.5 / 54.4
RoBERTa-large	63.5 / 50.7	62.7 / 62.2	88.2 / 50.0

Table 3: Question-Answer Accuracy for different models on the D_{and}, $D_{2,2}$ and $D_{3,3}$ datasets. The scores in each cell are in the format CoQA/FT.

5 Results

5.1 Results from the Q/A approach

We use our different datasets to see whether models can generalize from certain kinds of training to new datasets that resembled training data but introduced new elements—more modifications of objects, long distance links, etc. Generalizability is not guaranteed, as finetuning LLMs can lead to overfitting (Dehghani et al., 2019).

Table 2 shows the results of encoder models when further finetuned on window/door templates of $D_{1,1}$ for original question type. From the table, we can see that the encoder models, after finetuning, perform better on the augmented synthetic $D_{1,1}$ dataset in comparison to the original models. However, they clearly seem to overfit to the patterns of the $D_{1,1}$, as their performance drops drastically on the $D_{2.2}$ and $D_{3,3}$ datasets as shown in Table 3. The question answering task for $D_{2,2}$ and $D_{3,3}$ datasets is more difficult, as the models had to answer 8 questions for each example for $D_{2,2}$ and 12 for each example in $D_{3,3}$. A similar pattern is observed for D_{add} as well in Table 3.

Overall, our experiments show that encoder models finetuned on the simple synthetic dataset $D_{1,1}$ fail to generalize to more complex scenarios involving more properties or predicates ascribed to the two objects. Table 3 shows that further finetuning leads to near-random accuracy for all the models and they underperform their original counterparts on more complex datasets.

In addition, we note that the scores in Table 2 give accuracy for all the questions. This is not sufficient to determine predicate argument structure. In order to do so, we need the model to answer all questions for a particular input correctly. We refer to this accuracy as *pred-arg accuracy*. For pred-arg accuracy, the model needs to answer all 4 questions for a $D_{1,1}$ example correctly; whereas for a more complex dataset like $D_{3,3}$, it needs to answer all 12 yes/no questions correctly. Given the prefect scores in Table 2, the models will have a very high pred-arg accuracy on $D_{1,1}$, but not on more complex datasets, as evident from Table 3.

Table 4 shows the result of Q/A prompting for different LLMs. We report both Q/A as well as pred-arg accuracy. For both cases, overall as well as question type specific accuracy is provided. From the table, we can see that prompting leads to relatively good but not perfect scores on $D_{1,1}$. Prompting also provides a more graceful decline in performance over the more complex datasets. However, there are some peculiarities. Llama-2-7B and Llama-3-8b have low pred-arg accuracy, even though both the models have good overall Q/A accuracy. An exception is D_{and} dataset for LLama-3-8B where the model achieves good overall pred-arg accuracy. Across the LLMs, Llama2-13b performed the best indicating that a larger number of parameters helps on the more complex datasets, especially for pred-arg accuracy. Llama2-13b's overall pred-arg accuracy on the complex datasets like $D_{3,1}, D_{3,2}, D_{3,3}$ is often more than double that of Llama-3-8B and triple that of Llama 2-7B. Mistral-7B also achieves a decent overall pred-arg accuracy across different datasets.

As mentioned earlier, Table 4 also shows the Q/A and pred-arg accuracy separately for each question type across datasets and LLMs. Here, we see that, all the models treat the two semantically equivalent questions quite differently, as there is a significant difference in the two scores. This raises serious concerns about the robustness and reliability of the Q/A method, as well as theoretical issues about an LLM's grasp of the meaning of questions and question equivalence.

5.2 Results on the translation task

For FOL translation, all the LLMs are finetuned on FOLIO train set and window/door template of $D_{1,1}$. Table 5 shows two accuracy scores for the translation task: an accuracy score calculated

Model	Dataset	Q/A Accuracy	Pred-Arg Accuracy
Mistral-7B	$D_{1,1}$	87.2 / 94.1 / 80.3	59.0 / 78.4 / 39.6
	D_{and}	75.0 / 69.5 / 80.5	50.3 / 39.1 / 61.5
	$D_{2,1}$	86.4 / 89.4 / 83.4	41.2 / 45.2 / 37.2
	$D_{2,2}$	89.5 / 94.0 / 84.9	51.7 / 63.5 / 40.0
	$D_{3,1}$	80.8 / 84.9 / 76.6	20.2 / 23.0 / 17.4
	$D_{3,2}$	86.4 / 92.1 / 80.7	31.7 / 41.1 / 22.3
	$D_{3,3}$	89.3 / 96.1 / 82.4	48.5 / 61.8 / 35.3
Llama-2-7B	$D_{1,1}$	69.9 / 74.3 / 65.6	25.2 / 32.9 / 17.5
	D_{and}	64.5 / 72.2 / 56.9	30.4 / 46.9 / 13.8
	$D_{2,1}$	69.6 / 74.9 / 64.3	11.6 / 14.2 / 8.9
	$D_{2,2}$	76.4 / 81.2 / 71.7	22.7 / 23.1 / 22.3
	$D_{3,1}$	63.9 / 68.3 / 59.5	2.6 / 3.2 / 2.0
	$D_{3,2}$	69.9 / 75.4 / 64.5	6.1 / 8.8 / 3.4
	$D_{3,3}$	74.1 / 77.6 / 70.6	13.4 / 16.9 / 9.8
Llama-2-13B	$D_{1,1}$	89.1 / 91.8 / 86.4	62.6 / 72.2 / 53.1
	D_{and}	83.7 / 75.1 / 92.3	68.2 / 51.8 / 84.5
	$D_{2,1}$	90.2 / 93.5 / 86.9	51.7 / 61.9 / 41.6
	$D_{2,2}$	93.2 / 97.5 / 88.9	54.6 / 81.5 / 27.7
	$D_{3,1}$	88.1 / 90.5 / 85.7	36.1 / 45.2 / 27.0
	$D_{3,2}$	92.5 / 95.0 / 90.0	47.6 / 59.7 / 35.4
	$D_{3,3}$	93.3 / 94.1 / 92.6	49.6 / 56.0 / 43.2
Llama-3-8B	$D_{1,1}$	86.9 / 90.0 / 83.7	56.5 / 64.4 / 48.7
	D_{and}	86.1 / 90.5 / 81.7	73.9 / 84.0 / 63.8
	$D_{2,1}$	86.5 / 84.7 / 88.3	43.2 / 39.5 / 46.9
	$D_{2,2}$	84.8 / 81.6 / 88.0	35.0 / 27.7 / 42.3
	$D_{3,1}$	84.1 / 84.5 / 83.6	23.4 / 26.4 / 20.4
	$D_{3,2}$	85.1 / 83.8 / 86.4	21.8 / 23.8 / 19.7
	$D_{3,3}$	83.4 / 80.7 / 86.2	20.8 / 24.6 / 17.1

Table 4: Results of Q/A prompting for different LLMs and datasets. The three values in Q/A accuracy and pred-arg accuracy denote the **overall accuracy/ accuracy for the original question type/ accuracy for the modified question type** respectively. For overall pred-arg accuracy, the model has to predict correctly for all the questions of a particular question type.

relative to an exact match of the FOL translation with a gold standard (i.e., FOL accuracy) and the accuracy of the predicate argument structure that we can algorithmically infer from the LLM's predicted FOL (i.e. Pred-Arg accuracy). The discrepancy between the two scores shows that LLMs stray rather frequently from the translation paradigm for our synthetic dataset. One main problem problem has to do with an LLM strategy of *glueing predicates* together to avoid conjunctions. For instance *big, red, shiny car* might be translated as $BigRedShiny(x) \wedge Car(x)$ or simply as $BigRedShiny(Car)$ instead of the desired $Big(x) \wedge Red(x) \wedge Shiny(x) \wedge Car(x)$. The FOLIO dataset sometimes exemplifies this glueing strategy for complex predicates with modifiers like *very large*; so we hypothesize that the LLMs are learning this behavior during finetuning. Even though these glued predicate translations are not correct first order logic formulas, they can, however, easily be converted to the appropriate logical form.

The lack of accuracy for predicate argument structure based on translation surfaces in three other ways. First, the LLM's translation may *drop* certain predicates from logical form. For instance *big, red, shiny car* might be translated as $Big(x) \wedge Red(x) \wedge Car(x)$ or some other subsequence of the desired translation. Second, relational predicates like *in front of* as in *the big car in front of the old house* are not rendered correctly. A third source of difficulty is that sometimes models mess up the quantificational structure (though this happens rarely with Llama-2-13b or Llama-3-8b).

A final and major problem is that hallucinated content is sometimes added to the translation. In particular, Llama2-13b and Llama3-8b have significant rates of hallucination ($> 10\%$ of the cases on the $D_{3,2}$ dataset). For example,

1. "A clean red glass was placed on a modern dirty white table." $\exists x \exists y \exists z(Glass(x) \wedge Red(x) \wedge Clean(x) \wedge Table(y) \wedge Modern(y) \wedge White(y) \wedge Dirty(y) \wedge DontMindIfImWhite(z) \wedge Table(z))$

2. "A vintage blue glass was placed on a modern dirty red table." $\exists x \exists y \exists z(Vintage(x) \wedge BlueGlass(x) \wedge$

Model	Dataset	FOL Accuracy	Pred-Arg Accuracy	Hallucination Cases
Mistral-7B	$D_{1,1}$	48.0	85.7	12/440
	$D_{2,1}$	6.9	27.2	10/640
	$D_{2,2}$	0.4	22.3	0/260
	$D_{3,1}$	1.8	23.0	2/1368
	$D_{3,2}$	3.6	21.6	4/2088
	$D_{3,3}$	0.6	20.5	1/468
Llama-2-7B	$D_{1,1}$	52.5	88.6	16/440
	$D_{2,1}$	8.9	9.7	8/640
	$D_{2,2}$	11.2	11.2	8/260
	$D_{3,1}$	9.7	12.2	41/1368
	$D_{3,2}$	5.7	7.0	57/2088
	$D_{3,3}$	5.1	6.2	18/468
Llama-2-13B	$D_{1,1}$	73.4	92.3	27/440
	$D_{2,1}$	31.6	41.4	87/640
	$D_{2,2}$	33.1	44.6	42/260
	$D_{3,1}$	28.1	38.3	197/1368
	$D_{3,2}$	25.8	36.6	358/2088
	$D_{3,3}$	30.6	45.1	103/468
Llama-3-8B	$D_{1,1}$	81.4	90.5	33/440
	$D_{2,1}$	51.7	58.4	126/640
	$D_{2,2}$	60.8	70.0	53/260
	$D_{3,1}$	37.4	42.0	100/1368
	$D_{3,2}$	38.9	41.7	234/2088
	$D_{3,3}$	28.6	29.1	52/468
Llama-3-8B*	$D_{1,1}$	84.1	91.8	62/440
	$D_{2,1}$	64.1	67.4	135/512
	$D_{2,2}$	73.1	77.7	66/260
	$D_{3,1}$	47.6	50.1	165/1368
	$D_{3,2}$	52.1	56.1	275/2088
	$D_{3,3}$	50.9	51.7	69/468

Table 5: Results of FOL approach for different fine tuned models and datasets. The first four models were trained on FOLIO and window/door templates of $D_{1,1}$. However, llama3-8B* was trained on FOLIO, window/door templates of $D_{1,1}$ and $D_{2,1}$. The scores are provided on the test set. Hallucination cases are shown in the format $\#cases/\#total$.

$$Table(x) \wedge Modern(x) \wedge Dirty(x) \wedge Red(x) \wedge Vintage(y) \wedge BlueGlass(y) \wedge Cabinet(y) \wedge Queen(y) \wedge Vintage(z) \wedge BlueGlass(z) \wedge Toilet(z) \wedge King(z))$$

Table 5 gives figures for hallucination cases as well.

We also note some interesting peculiarities about these failings. A model may adopt the glueing predicate strategy for instances of one template but not another. Further, if a model adopts a glueing predicate strategy, normally it hallucinates very little. The difference between the two accuracies (FOL and Pred-Arg) is an indicator of the extent of glueing in model's predicted FOL. As we see from the table, the small models also produce far fewer hallucinations than the larger models; although their accuracy score wae significantly low. Mistral-7B has very few hallucination cases as it mostly relied on glueing predicates.

In Section 5.1, we saw that finetuning smaller encoder models for the Q/A task on our synthetic led to overfitting to the training pattern. Unfortunately, we see a similar pattern especially with the small LLMs on their predictions for the FOL task when we move from the simple dataset to more complex ones. While results are very good on the $D_{1,1}$ dataset, the smaller models fail pretty much completely once we have two modifiers of one noun in the $D_{2,1}$ dataset; they fail to produce anything meaningful on $D_{2,1}$ or more complex datasets $D_{2,2}, D_{3,1}, D_{3,2}, D_{3,3}$. Even Llama-2-13b and Llama-3-8B, perform considerably worse when we move to the more complex datasets .

Nevertheless, Llama-2-13b and Llama-3-8b show considerable generalization ability on the complex datasets, as they achieve an accuracy score in the double digits. For complex datasets, these models also fail to mention some of the predicates that should be in the logical form. In these cases, we can't reconstruct the proper logical form, thereby resulting in lower accuracy. More particularly, Llama-2-13b has a drastic performance drop from $D_{1,1}$ to $D_{2,1}$. $D_{2,1}$ is the simplest dataset after $D_{1,1}$. On the other hand, it performs similar on the balanced sets $D_{2,2}$ and $D_{3,3}$. The imbalanced datasets $D_{3,2}$ and $D_{3,1}$ is

more challenging for the model than the balanced dataset $D_{3,3}$.

Llama-3-8B's performance still attains good accuracy for $D_{2,2}$ but drops significantly for the imbalanced dataset $D_{2,1}$. Its performance drops further once we move to $D_{3,1}$ and $D_{3,2}$. Unlike Llama-2-13B, Llama-3-8B has the lowest performance on the $D_{3,3}$ dataset. In order to address the issue of imbalanced datasets, we augmented the training set for Llama-3-8B by also including window/door templates of $D_{2,1}$. The resultant model is referred to as Llama3-8b* in Table 5. We can see that Llama3-8b* has much higher accuracy on $D_{2,1}$ but, not to the level of $D_{1,1}$. One possible reason for this is Llama-3's very high ($\sim$ 25%) hallucination rate on that dataset. However, its accuracy for the rest of the complex datasets increases considerably, surpassing its counterpart, Llama-3-8B, as well as Llama-2-13B, by a large margin.

5.3 Comparisons between Q/A and FOL translation

Both Q/A and FOL translation show that models were able to solve difficulties with predicate argument structures that come from long distance dependencies. However, with respect to difficulties that stem from depth of embedding, these methods have quite different characteristics. Finetuning encoder models for the Q/A task provided essentially perfect accuracy on the $D_{1,1}$ dataset. However, the scores plummeted when the finetuned models had to answer simple yes/no questions on more complex datasets like $D_{2,2}$ or $D_{3,3}$. The scores even for LLMs dropped considerably in terms of pred-arg accuracy, as seen in Table 4.

Furthermore, encoder models and LLM performance on Q/A tasks is quite unstable in that semantically irrelevant differences in the surface form of the question affects how the models respond. Finally, even an exhaustive Q/A as in Table 4 only offers a partial view of the predicate argument structure; it tells us what predicates that are mentioned go with which arguments are mentioned, but it does not prevent the model from adding hallucinatory content of an arbitrary nature. This hallucinatory content can clearly affect downstream reasoning tasks. Given the random nature of the hallucinatory content, we see no finite way of using Q/A to eliminate the possibility of hallucinated content. Therefore, we conclude that Q/A is not an optimal way to probe for mastery of predicate argument structure or for models to learn it. The FOL translation task, on the other hand, makes hallucinatory content immediately obvious and also completely determines predicate argument structure.

5.4 Comparisons between finetuning and prompting for pred-arg structure

Prompting for the Q/A strategy with LLMs yields a substantially lower overall pred-arg accuracy in comparison to their FOL finetuned counterparts and even smaller finetuned models like BERT, and RoBERTa. If we compare predicate-argument accuracy for a given model (even though the two tasks do not give equivalent results because of the presence of unanticipated hallucinatory content), we see that finetuning for translation on Mistral gives much worse results than prompting on Q/A. For Llama-2-7B, the results for the two approaches are equally bad. For Llama2-13B, prompting achieves higher scores especially for the $D_{2,1}$ and $D_{2,2}$ datasets. However, for these datasets, the scores vary significantly for the two question types. Llama-3-8b does better with FOL finetuning. Given the increased difficulty of the translation task, we conclude that the finetuning provides better results, in spite of the overfitting problem. From this, we conclude that finetuning at least for predicate argument structure shows evidence of higher generalizability than prompting. Further evidence for our conclusion is the fact that, across the two tasks, Llama-3-8b* has the best pred-arg accuracy score for majority of the datasets.

6 Conclusions

In this paper, we investigate smaller encoder models and moderate sized LLM's grasp of predicate argument structure for simple sentences. We use two types of methods for finding predicate argument structure: Q/A and first order logic (FOL) translation and examined their behavior on our synthetic datasets. The results show that neither of the two approaches succeed in mastering this fundamental aspect of meaning. After finetuning, encoder models still could not generalize from their training to find predicate argument structure of more complex sentences. However, LLMs does manage to show considerable generalization ability with finetuning on the FOL translation task. Overall, we find that finetuning on the translation

task gave the best results for learning predicate argument structure.

Our results also show that LLMs (especially Llama-2-13B, and Llama-3-8B) tend to hallucinate when finetuned for FOL translation. Recent work has shown that a RAG-based approach can help mitigate hallucination (Ayala and Bechard, 2024). This could be a possible avenue for future work, where a RAG-based approach limits the predicates a model can use in its translation into FOL.

We have discussed prompting vs. finetuning on models of the same size. However, for much larger models, or models that are not open, finetuning is not an option. Chaturvedi et al. (2024) show that the GPT-instruct models (Ouyang et al., 2022), namely, text-davinci-002, and text-davinci-003; achieve near-perfect accuracy on their basic predicate-argument dataset. We plan to test the latest GPT models on our more complex datasets in the future.

Limitations

In this work, we investigate the models' ability to capture predicate argument structure with embedded modifiers and long distance dependencies. However, we do not put the two difficulties together, say in trying the models out on sentences with two predicates for each argument, at least one of which was connected via a long distance dependency. As suggested by our results, we suspect that finding the predicate argument structure of such sentences would be very difficult for all the models we tested. We also do not test how logical operators might affect predicate argument structure as our models already have difficulty just with the simple affirmative contexts.

We find that, for encoder models, the latent representations of relevant predicates have higher weight in the internal representations of their arguments. We try to reinforce this property with a loss function using a method inspired by Raissi et al. (2017). But, this approach doesn't enhance the Q/A accuracy of the model. Even finetuning on an exogenous source of information from FOL translation does not achieve full mastery of the predicate argument structure of the simple sentences in our synthetic datasets.

Ethics Statement

This work shows that LLMs finetuned on FOL translation tend to hallucinate for some cases. The hallucinated content is often completely unrelated to the input text. This poses a serious challenge which needs to be addressed before deploying such models in practice. The lack of robustness of LLMs across semantically equivalent questions is also a detriment to their applicability in the real world.

Acknowledgement

For financial support, we thank the National Interdisciplinary Artificial Intelligence Institute ANITI (Artificial and Natural Intelligence Toulouse Institute), funded by the French 'Investing for the Future– PIA3' program under the Grant agreement ANR-19-PI3A-000. This project has been funded by the French government as part of France 2030 and is funded by the European Union - Next Generation EU as part of the France Relance. This research is also supported by the Indo-French Centre for the Promotion of Advanced Research (IFCPAR/CEFIPRA) through Project No. 6702-2 and Science and Engineering Research Board (SERB), Dept. of Science and Technology (DST), Govt. of India through Grant File No. SPR/2020/000495. This work was granted access to the HPC resources of CALMIP supercomputing center under the allocation 2016-P23060.

References

Ayala, O. and Bechard, P. (2024). Reducing hallucination in structured outputs via Retrieval-Augmented Generation. In Yang, Y., Davani, A., Sil, A., and Kumar, A., editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pages 228–238, Mexico City, Mexico. Association for Computational Linguistics.

Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y. T., Li, Y., Lundberg, S., et al. (2023). Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv preprint arXiv:2303.12712*.

Chaturvedi, A., Bhar, S., Saha, S., Garain, U., and Asher, N. (2024). Analyzing Semantic Faithfulness of Language Models via Input Intervention on Question Answering. *Computational Linguistics*, 50(1):119–155.

Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., and Kaiser, L. (2019). Universal Transformers. In *International Conference on Learning Representations*.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. (2023). QLoRA: Efficient

Finetuning of Quantized LLMs. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al. (2024). The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*.

Feng, J. and Steinhardt, J. (2023). How do language models bind entities in context? In *NeurIPS 2023 Workshop on Symmetry and Geometry in Neural Representations*.

Goldberg, Y. (2019). Assessing BERT's Syntactic Abilities. *arXiv preprint arXiv:1901.05287*.

Han, S., Schoelkopf, H., Zhao, Y., Qi, Z., Riddell, M., Zhou, W., Coady, J., Peng, D., Qiao, Y., Benson, L., Sun, L., Wardle-Solano, A., Szabo, H., Zubova, E., Burtell, M., Fan, J., Liu, Y., Wong, B., Sailor, M., Ni, A., Nan, L., Kasai, J., Yu, T., Zhang, R., Fabbri, A. R., Kryscinski, W., Yavuz, S., Liu, Y., Lin, X. V., Joty, S., Zhou, Y., Xiong, C., Ying, R., Cohan, A., and Radev, D. (2024). FOLIO: Natural Language Reasoning with First-Order Logic. *arXiv preprint arXiv:2209.00840*.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2023). Mistral 7B. *arXiv preprint arXiv:2310.06825*.

Lakretz, Y., Desbordes, T., Hupkes, D., and Dehaene, S. (2022). Can Transformers Process Recursive Nested Constructions, Like Humans? In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3226–3232, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Li, B., Wisniewski, G., and Crabbé, B. (2023). Assessing the Capacity of Transformer to Abstract Syntactic Representations: A Contrastive Analysis Based on Long-distance Agreement. *Transactions of the Association for Computational Linguistics*, 11:18–33.

Linzen, T., Dupoux, E., and Goldberg, Y. (2016). Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Olausson, T., Gu, A., Lipkin, B., Zhang, C., Solar-Lezama, A., Tenenbaum, J., and Levy, R. (2023). LINC: A Neurosymbolic Approach for Logical Reasoning by Combining Language Models with First-Order Logic Provers. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5153–5176, Singapore. Association for Computational Linguistics.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2017). Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*.

Reddy, S., Chen, D., and Manning, C. D. (2019). CoQA: A Conversational Question Answering Challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Steedman, M. J. (1996). *Surface Structure and Interpretation*. MIT Press.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is All you Need. *Advances in neural information processing systems*, 30.

7 Appendix

Model	Org-Acc	Mod-Acc
BERT-base	50.0 (100.0)	69.4 (59.8)
BERT-large	95.2 (51.0)	77.3 (27.3)
RoBERTa-base	51.0 (99.0)	70.0 (78.5)
RoBERTa-large	99.4 (49.4)	95.0 (45.0)
XLNet-base	50.6 (6.0)	50.8 (0.7)
XLNet-large	75.2 (74.8)	79.8 (36.3)

Table 6: Effect of question paraphrasing on different models of Chaturvedi et al. (2024). Questions of type "Was the X *col1*?" are referred to as original questions (org) and question of type "Was there a *col1* X?" is referred to as modified questions (Mod). The number in brackets denote percentage of cases where the model predicted "no" as the answer.

Template	FOL
The col1 car was standing in front of a col2 house.	$\exists x \exists y$(Car(x) $\wedge$ col1(x) $\wedge$ House(y) $\wedge$ col2(y) $\wedge$ standing-in-front-of(x,y))
The car that was col1 was standing in front of a house that was col2.	$\exists x \exists y$(Car(x) $\wedge$ col1(x) $\wedge$ House(y) $\wedge$ col2(y) $\wedge$ standing-in-front-of(x,y))
col2 was not the color of the car but of the house.	$\exists x \exists y$(Car(x) $\wedge \neg$ col2(x) $\wedge$ House(y) $\wedge$ col2(y))
col1 was the color of the car in front of col2 house.	$\exists x \exists y$(Car(x) $\wedge$ col1(x) $\wedge$ House(y) $\wedge$ col2(y) $\wedge$ infrontof(x,y))
The car that was in front of the col2 house was col1.	$\exists x \exists y$(Car(x) $\wedge$ col1(x) $\wedge$ House(y) $\wedge$ col2(y) $\wedge$ infrontof(x,y))
They played with a col1 ball and col2 bat.	$\exists x \exists y$(Ball(x) $\wedge$ col1(x) $\wedge$ Bat(y) $\wedge$ col2(y) $\wedge$ play-with(they,x) $\wedge$ play-with(they,y))
The ball that they played with was col1 and the bat was col2.	$\exists x \exists y$(Ball(x) $\wedge$ col1(x) $\wedge$ Bat(y) $\wedge$ col2(y) $\wedge$ play-with(they,x) $\wedge$ play-with(they,y))
col2 was not the color of the ball but of the bat.	$\exists x \exists y$(Ball(x) $\wedge \neg$ col2(x) $\wedge$ Bat(y) $\wedge$ col2(y))
col1 was the color of the ball that was hit by the col2 bat.	$\exists x \exists y$(Ball(x) $\wedge$ col1(x) $\wedge$ Bat(y) $\wedge$ col2(y) $\wedge$ was-hit-by(x,y))
The ball that was hit by the col2 bat was col1.	$\exists x \exists y$(Ball(x) $\wedge$ col1(x) $\wedge$ Bat(y) $\wedge$ col2(y) $\wedge$ was-hit-by(x,y))
The man was wearing a col1 shirt and a col2 jacket.	$\exists x \exists y \exists z$ (Shirt(x) $\wedge$ col1(x) $\wedge$ Jacket(y) $\wedge$ col2(y) $\wedge$ Man(z) $\wedge$ wear(z,x) $\wedge$ wear(z,y))
The shirt that the man wore was col1 and the jacket was col2.	$\exists x \exists y \exists z$ (Shirt(x) $\wedge$ col1(x) $\wedge$ Jacket(y) $\wedge$ col2(y) $\wedge$ Man(z) $\wedge$ wear(z,x) $\wedge$ wear(z,y))
col2 was not the color of the shirt but of the jacket.	$\exists x \exists y$(Shirt(x) $\wedge \neg$ col2(x) $\wedge$ Jacket(y) $\wedge$ col2(y))
col1 was the color of the shirt with the col2 jacket.	$\exists x \exists y$ (Shirt(x) $\wedge$ col1(x) $\wedge$ Jacket(y) $\wedge$ col2(y))
The shirt that went with col2 jacket was col1.	$\exists x \exists y$ (Shirt(x) $\wedge$ col1(x) $\wedge$ Jacket(y) $\wedge$ col2(y) $\wedge$ went-with(x,y))
The house had a col1 window and a col2 door.	$\exists x \exists y \exists z$ (Window(x) $\wedge$ col1(x) $\wedge$ Door(y) $\wedge$ col2(y) $\wedge$ House(z) $\wedge$ had(z,x) $\wedge$ had(z, y))
The window that was col1 was next to the door that was col2.	$\exists x \exists y$ (Window(x) $\wedge$ col1(x) $\wedge$ Door(y) $\wedge$ col2(y) $\wedge$ next-to(x,y))
col2 was not the color of the window but of the door.	$\exists x \exists y$ (Window(x) $\wedge \neg$ col2(x) $\wedge$ Door(y) $\wedge$ col2(y))
col1 was the color of the window next to the col2 door.	$\exists x \exists y$ (Window(x) $\wedge$ col1(x) $\wedge$ Door(y) $\wedge$ col2(y) $\wedge$ next-to(x,y))
The window that was next to the col2 door was col1.	$\exists x \exists y$ (Window(x) $\wedge$ col1(x) $\wedge$ Door(y) $\wedge$ col2(y) $\wedge$ next-to(x,y))
A col1 glass was placed on a col2 table.	$\exists x \exists y$ (Glass(x) $\wedge$ col1(x) $\wedge$ Table(y) $\wedge$ col2(y) $\wedge$ placed-on(x,y))
The glass that was col1 was placed on a table that was col2.	$\exists x \exists y$ (Glass(x) $\wedge$ col1(x) $\wedge$ Table(y) $\wedge$ col2(y) $\wedge$ placed-on(x,y))
col2 was not the color of the glass but of the table.	$\exists x \exists y$ (Glass(x) $\wedge \neg$ col2(x) $\wedge$ Table(y) $\wedge$ col1(y))
col1 was the color of the glass placed on the col2 table.	$\exists x \exists y$ (Glass(x) $\wedge$ col1(x) $\wedge$ Table(y) $\wedge$ col2(y) $\wedge$ placed-on(x,y))
The glass that was placed on a col2 table was col1.	$\exists x \exists y$ (Glass(x) $\wedge$ col1(x) $\wedge$ Table(y) $\wedge$ col2(y) $\wedge$ placed-on(x,y))

Table 7: Templates and FOL for $D_{1,1}$ dataset. *col1* and *col2* refer to two distinct colors.

GPUs	
4 NVIDIA Volta V100	
Hyperparameters	
Training epochs	10
batch size	4
optimizer	Adam
learning rate	2e-4
learning rate scheduler	linear warm-up and cosine annealing
warm-up ratio	0.03
gradient clipping	0.3
lora r	64
lora (alpha)	16
lora dropout ratio	0.1
lora target modules	Only Attention Blocks (q_proj, v_proj)
quantization	4-bit NormalFloat

Table 8: Details on computing resources and hyperparameters for finetuning LLMs for FOL translation.

Table 8 gives the hyperparameters used for finetuning LLMs for FOL translation along with the computing resources. We adapt the finetuning code from the following repository[2]. Finetuning LLMs as per the hyperparameters given in Table 8 took around 5 hours.

[2] https://github.com/mlabonne/llm-course/blob/main/Fine_tune_Llama_2_in_Google_Colab.ipynb