Probing LLMs for logical reasoning

Probing LLMs for logical reasoning

Francesco Manigrasso[†1][0000−0002−4151−8880], Stefan Schouten[†2][0000−0001−9839−9985], Lia Morra[1][0000−0003−2122−7178], and Peter Bloem[2][0000−0002−0189−5817]

[1] Politecnico di Torino, Turin, Italy
name.surname@polito.it
https://www.polito.it/
[2] Vrije Universiteit, Amsterdam, Netherlands
n.surname@vu.nl
https://vu.nl/en

Abstract. Recently, the question of what types of computation and cognition large language models (LLMs) are capable of has received increasing attention. With models clearly capable of convincingly faking true reasoning behavior, the question of whether they are also capable of real reasoning—and how the difference should be defined—becomes increasingly vexed. Here we introduce a new tool, Logic Tensor Probes (LTP), that may help to shed light on the problem. Logic Tensor Networks (LTN) serve as a neural symbolic framework designed for differentiable fuzzy logics. Using a pretrained LLM with frozen weights, an LTP uses the LTN framework as a diagnostic tool. This allows for the detection and localization of logical deductions within LLMs, enabling the use of first-order logic as a versatile modeling language for investigating the internal mechanisms of LLMs. The LTP can make deductions from basic assertions, and track if the model makes the same deductions from the natural language equivalent, and if so, where in the model this happens. We validate our approach through proof-of-concept experiments on hand-crafted knowledge bases derived from WordNet and on smaller samples from FrameNet.

Keywords: NeuroSymbolic AI · Logic Tensor Networks · Probing Large Language Models.

1 Introduction

With the recent release of ChatGPT and GPT-4 [27], there has been a notable interest regarding the types of reasoning that large language models (LLMs) can exhibit [7,20,30]. It is evident that LLMs have the ability to generate outputs resembling reasoning that may initially appear well-reasoned but lack genuine insight upon closer examination. Nevertheless, within specific domains, LLMs demonstrate the capability to provide accurate and well-reasoned responses [37,1,4,17]. A methodology capable of distinguishing authentic reasoning

[†] Equal contribution.

from "hallucinatory" outputs generated by LLMs can offer the advantage of precisely isolating the reasoning behaviors and capabilities of LLMs and developing systems capable of choosing between valid reasoning and superficial or incorrect results. Furthermore, it could be useful during the training phase to promote accurate reasoning behaviors and discourage the production of unreliable results.

In this research, we introduce the *Logic Tensor Probe (LTP)*, tailored specifically for assessing the reasoning capabilities of Large Language Models (LLMs). The LTP analyzes the knowledge acquired by the LLM model and evaluates the validity of statements within a given knowledge base. The fundamental idea is to use the the *Logic Tensor Network (LTN)* framework [2,29], which normally functions as an auxiliary loss to promote symbolic reasoning, and to have it function as a *probe* [11,18] instead. A *probe*[3] is a shallow machine learning model that is optimized to predict a target value t, based only on the hidden activations produced by a larger, frozen model m for each instance x. If the probe succeeds in predicting t, it shows that the model's representation of x contains the information necessary to map x to t. This can, for instance, be used to quantify for each layer to what extent m has learned how t relates to x [5].

The LTN framework is a *Neural Symbolic (NeSy)* method that formulates learning as maximizing the satisfiability of a knowledge base $\mathcal{K}$ expressed in Real Logic, a differentiable First Order Logic (FOL) language. Here, the LTP is trained to compute the degree of truthiness of predicates, such as `isOfClass`, that map the properties of specific instances. The LTP predicates are grounded by shallow classifiers thus acting as the probes. At training time, the LTP is trained to encode statements, such as "Gordon is a German Shepherd" and "A German Shepherd is a Dog" by learning to satisfy the corresponding FOL statements. The LTP is then used to probe the LLM for the degree of truthiness of statements that are logically entailed by those used to train it, such as "Gordon is a German Shepherd and Gordon is a Dog". If the FOL formula is satisfied, even though the `isOfClass` predicate was not explicitly trained to map the concept "Gordon" to the concept "Dog", we take it as evidence that the LTP must be able to compute the correct truth value based on information implicitly contained in the LLM embeddings.[4]By observing in which layers the degree of truthiness is maximized, we can investigate whether and which types of logical deductions are in principle possible based on the LLM embeddings. To be consistent with some form of deductive reasoning, we would expect the degree of truthiness of different FOL statements to be maximized by layers at different depths, e.g., $\texttt{isOfClass}(Gordon, GermanShepherd)$ to be optimally satisfied at deeper levels than $\texttt{isOfClass}(Gordon, Dog)$. We evaluate a proof-of-concept of this approach in two settings: first, using a newly constructed knowledge base

[3] The phrase *probe* is also used to refer to general inspection methods for neural networks. Here, we use it specifically to refer to shallow classifiers that take hidden activations as features.

retrieved from WordNet [12], modeled on named animals and their habitats, as shown in the running examples. Second, using a subset of FrameNet [3].

Our experiments show that:

1. we can successfully train an LTN probe to use the data contained within the LLM representations
2. we can determine when an LLM deduces information from a given sentence that is not explicitly stated but is implied and captured by logical knowledge.[5]

The remainder of the paper is structured as follows. Section 2 introduces background information on the LTN framework and related work. The section 3 describes the proposed architecture. In section 3.2, we analyze the behavior of the LLM with sentences retrieved from WordNet and FrameNet including in section 4 a description of the LLM models included in our settings. Finally, in section 5 and section 6, we discuss the results and future work. Code and data available at: `https://github.com/sfschouten/ltn_probes`.

2 Related Work

Probing Previous work has investigated probes for the semantic roles of phrases, which involves predicting what *role* different noun phrases play in the meaning of a sentence [9]. The investigation task would then involve predicting the syntactical or semantic roles of each sentence from the representation of (parts of) a sentence [33,23]. This is a specific case of our investigative task, in which we try to specifically recover that argumentative structure of the predicates that come closest to what is explicitly stated or predicates not explicitly expressed within, but still *implicit* from the sentence.

The probe training is separate from the LLM training, ensuring they measure the LLM's pre-existing knowledge. A key difference among different approaches is how the LLM internal representations are mapped to the probe inputs. Some previous work has used sentence representations [11,21], whereas others take representations specific to the spans of the relevant noun phrases [32,31,9,19,6]. Our probe does not operate only on those representations related to the span of the argument noun phrases, but *learns* to attend to the relevant parts of the sentence as part of its training procedure.

[4] Throughout the text we use the word *embedding* to refer to the LLM's representations of its input tokens at *any* stage of the LLM's execution, not just at the initial embedding stage.

[5] Possible alternative explanations include a model recalling a stored association rather than reasoning from scratch. LTN probes are *tools* that can be used to investigate such possibilities and, through careful use, eliminate them. We do not claim that a successfully trained LTN probe is always proof that an LLM shows the modelled reasoning.

NeSy Architectures In recent years, a considerable number of researchers have focused their attention on NeSy architectures for post-hoc interpretability [28]. Logic Tensor Networks (LTNs) [2] and Real Logic (RL) are NeSy architectures used in various image tasks such as zero-shot learning and Sudoku puzzle classification, as well as reasoning tasks in combination with NeSy models and curriculum learning [26,25,22]. RL refers to the mapping of a FOL to a differentiable fuzzy logic, and LTNs refer to the neural nets built to implement an RL knowledge base. LTNs are typically incorporated as an additional module, to enhance the reasoning capabilities of the overall architecture. Manigrasso [24] et al. demonstrated that an end-to-end training approach can facilitate knowledge sharing between a feature extractor and the LTN block. Two recent works have applied LTNs in the natural language domain, focusing on Aspect-Term Sentiment Analysis [36] and argument mining [14]. To the best of our knowledge we are the first to use LTNs as a *probe*, with the weights of the main network explicitly frozen.

3 Methodology: Logic Tensor Probe

In this section, we introduce the basic notation to define *LTNs* used as *probes*. We build the concepts to translate a sentence into FOL language showing how the *LTP* training is constructed by substituting the original training set with a grounded knowledge base $\mathcal{K}$. A *diagnostic* classifier is introduced taking as input the representations (hidden states) produced by an LLM while the original loss is converted into the best satisfiability framework of the LTN.

3.1 Conversion into Real Logic

In LTNs, grounding refers to the assignment of real-valued semantics to logical symbols, where individuals are depicted as vectors in a high-dimensional space and variables are associated with specific entities. Neural networks are used to model predicates, establishing relationships between entities and their truth values. Complex expressions are formed using a combination of logical connectives ($\wedge$, $\vee$, $\rightarrow$, $\neg$) and quantifiers ($\forall$, $\exists$), grounded through mathematical functions, generally neural networks, into true domain[0,1]. Training LTNs involves constructing a knowledge base with FOL axioms, with the aim of maximizing the satisfaction of these axioms. The loss is quantified by the cumulative degree of truthiness of the axioms. The objective is to determine the optimal parameters that maximize this satisfaction, as described by the formula:

$$\theta^* = \arg\max_{\theta} \left(\text{SatAgg}_{\phi \in \mathcal{K}} \left(\mathcal{G}_\theta(\phi) \right) \right).$$

3.2 From a sentence to a Probe

Defining the LTP requires a knowledge base comprising natural language sentences paired with corresponding FOL formulas expressing (part of) their semantics. In our experiments, we used two datasets: WordNet and FrameNet.

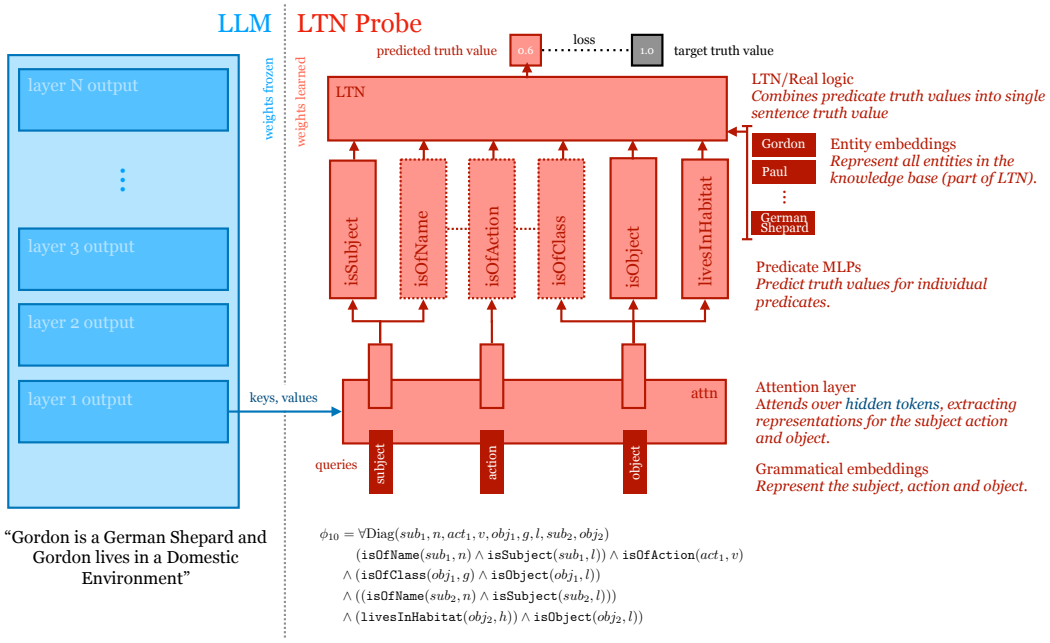

Fig. 1: A diagram of our approach. The LLM weights are frozen and a forward pass is calculated for a simple English sentence. Next, the activations of a chosen layer (in this case the first) are fed to an LTN that logically represents the sentence, along with some known implications that we are interested in. ϕ_{10} (Equation 29)represents the sentence FOL representation of the sentence "Gordon is a German shepherd and Gordon lives in a domestic environment". Using the LLM model with an appropriate attention mechanism, subject, action and object embeddings are extracted and set as inputs to the predicates of the LTN, which computes the degree of truthiness of the FOL representation of the entire sentence.

While LLMs process sentences token-by-token, to compute the LTP we instead need embeddings organized into subject, predicate and object components. Previous work has assumed that we know exactly which tokens represent which component, and typically use a fixed mapping between them [15,23]. We instead use a scaled dot-product attention mechanism, where each component gets a query vector (the grammatical embedding), and each token a key and value vector. The query vectors are a learned part of the probe and are initialized randomly. The key and value vectors are extracted from the LLM embeddings with a linear transformation. The final representation for each component is constructed with a standard attention operation: as a weighted sum over token representations (the value vectors), where the weights are calculated based on the (scaled) dot product between the query and key vectors. Thus, our LTN probes do not make any a priori assumptions about which tokens represent the subject, predicate, and object. Our goal is to analyze the information learned at each level of the LLM in the context of the selected sentence, identifying syntactic roles that emerge within specific layers. An illustrative example of this process is presented in Figure 1.

Knowledge Base Construction from WordNet Synsets Concepts were extracted from the synsets of WordNet [12] (see also Appendix B). This knowledge base $\mathcal{K}$ consists of statements related to various entities and categories within the animal kingdom, as inferred from the synsets. Categories were extracted through a hierarchical process involving two levels (macro class and class), starting from a root synset and traversing the tree formed by the hypernym relation. This methodology allows for the formulation of a knowledge base $\mathcal{K}$ containing a series of statements used to express relationships between the training data and the labels employed to train the probe. Subsequently, a different set of axioms is introduced at test time to assess the consistency of the information present in the LLM model by testing it with the probe.

We extracted 271 animal names, 753 distinct animal classes, 28 distinct habitat types, and 8 macro classes. The training set comprises a total of 6014 statements, evenly distributed between positive and negative statements.

The training set comprises statements that serve as examples for the LTP to learn the predicate grounding. The training set contains sentences such as:

- "Gordon is a German Shepherd" (positive sentence)
- "German Shepherd is a Carnivore" (positive sentence)
- "German Shepherd lives in a Domestic Environment" (positive sentence)
- "German Shepherd is a Gordon" (negative sentence)

A negative sentence is one in which the subject and object are reversed, creating semantically incorrect or logically inconsistent statements.

Conversely, the test set consists of FOL statements, and corresponding natural language sentences, that the LTP has not been exposed to during training. These statements are used to investigate whether LLMs are capable of making logical deductions. Below is a sample of the test set:

– "Kobe is a cheetah" (positive sentence)
– "Kobe is a carnivore" (positive sentence)
– "Lancelot is a saiga" (positive sentence)
– "Lancelot lives in Grassland" (positive sentence)

Through logical deductions enabled by the FOL representation, we obtain more complex sentences to probe the LLM:

– "Kobe is a cheetah and Kobe is a carnivore" (positive sentence)
– "Lancelot is a saiga and Lancelot lives in Grassland" (positive sentence)

Since the LTP must learn to map the extracted embeddings to entities, we include the same entities in both training and test sets, but in different sentences.

LTN Probing with WordNet. Each entity and animal class, along with its corresponding macro class and reference habitat, is formulated in the LTN framework. Below are the axioms used to train the LTP:

$$\phi_1 = \forall \text{Diag}(sub, n)(\texttt{isOfName}(sub, n)) \tag{1}$$
$$\phi_2 = \forall \text{Diag}(act, v)(\texttt{isOfAction}(act, v)) \tag{2}$$
$$\phi_3 = \forall \text{Diag}(obj, g)(\texttt{isOfClass}(obj, g)) \tag{3}$$
$$\phi_4 = \forall \text{Diag}(obj, q)(\texttt{isOfMacroclass}(obj, q)) \tag{4}$$
$$\phi_5 = \forall \text{Diag}(obj, h)(\texttt{livesInHabitat}(obj, h)) \tag{5}$$
$$\phi_6 = \forall \text{Diag}(sub, l)(\texttt{isSubject}(sub, l)) \tag{6}$$
$$\phi_7 = \forall \text{Diag}(obj, l)(\texttt{isObject}(obj, l)) \tag{7}$$

where the variables g, q, n, a, h, and l represent different class labels, each belonging to a specific set:G (Classes, e.g., animal species), Q (Macroclasses, e.g., macro categories), N (Names, e.g., specific entity names), A (Actions, e.g., verbs describing actions), H (Habitats, e.g., environments where entities live), and L (Labels indicating whether the element is a subject or object in the sentence). One-hot encoding is used for labels (refer to subsection B.1 for more details on the LTN grounding). In the LTP training, we use labeled examples for each component of a sentence. Specifically, we extract features for the subject (*sub*), action (*act*), and object (*obj*) to train the predicates. For instance, in the sentence "Gordon is a German Shepherd", *sub* denotes "Gordon", *act* indicates "is a" and *obj* refers to "German Shepherd". The extracted *sub* feature is used for predicates such as `isOfName` and `isSubject`, the *act* feature for the `isOfAction` predicate, and the *obj* feature for predicates such as `isOfClass`, `isOfMacroclass`, `livesInHabitat`, and `isObject`. In sentences like "German Shepherd is a Carnivore" and "German Shepherd lives in a Domestic Environment", the *obj* feature is used for the `isOfMacroclass` and `livesInHabitat` predicates, respectively.

The $\forall$ aggregator is grounded by the generalized mean with respect to the error. More details on the grounding of logical and non-logical symbols can be

found in Appendix B.3, while the hyperparameters used during training are provided in Appendix A.1.

During the testing phase, we introduce a series of complex sentences to evaluate the LTP's ability to identify relationships between entities based on its training.

For example, while giving to the LLM as input a sentence like "Gordon is a German Shepherd and Gordon is Carnivorous", we use the LTP to verify the satisfiability of the axioms Equation 28 and Equation 29.

The probe handles multiple embeddings ($sub_1, sub_2, act_1, obj_1, obj_2$, etc.) by simultaneously processing the entire sentence and extracting the necessary components in a single pass. This allows us to identify whether the extracted representations are related to the first or second part of the sentence.

$$\begin{aligned}
\phi_8 = \forall \text{Diag}(&sub, n, act, v, obj, g, l) \\
&(\texttt{isOfName}(sub, n) \wedge \texttt{isSubject}(sub, l)) \wedge \texttt{isOfAction}(act, v) \\
&\wedge (\texttt{isOfClass}(obj, g) \wedge \texttt{isObject}(obj, l)) \qquad (8) \\
\phi_9 = \forall \text{Diag}(&sub_1, n, act_1, v, obj_1, g, l, sub_2, act_2, obj_2) \\
&(\texttt{isOfName}(sub_1, n) \wedge \texttt{isSubject}(sub_1, l)) \wedge \texttt{isOfAction}(act_1, v) \\
&\wedge (\texttt{isOfClass}(obj_1, g) \wedge \texttt{isObject}(obj_1, l)) \\
&\wedge (\texttt{isOfName}(sub_2, n) \wedge \texttt{isSubject}(sub_2, l)) \wedge \\
&(\texttt{isOfMacroclass}(obj_2, q) \wedge \texttt{isObject}(obj_2, l)) \qquad (9) \\
\phi_{10} = \forall \text{Diag}(&sub_1, n, act_1, v, obj_1, g, l, sub_2, obj_2) \\
&(\texttt{isOfName}(sub_1, n) \wedge \texttt{isSubject}(sub_1, l)) \wedge \texttt{isOfAction}(act_1, v) \\
&\wedge (\texttt{isOfClass}(obj_1, g) \wedge \texttt{isObject}(obj_1, l)) \\
&\wedge ((\texttt{isOfName}(sub_2, n) \wedge \texttt{isSubject}(sub_2, l))) \\
&\wedge (\texttt{livesInHabitat}(obj_2, h)) \wedge \texttt{isObject}(obj_2, l)) \qquad (10)
\end{aligned}$$

Axiom ϕ_8 represent the degree of truthiness of the sentence like "Gordon is a German Shepherd", axiom ϕ_9 of "Gordon is a German Shepard and Gordon is Carnivorous", axiom ϕ_{10} of "Gordon is a German Shepard and Gordon lives in a domestic environment" as showed in section 3.2.

Knowledge Base Construction from FrameNet Synsets In a second experiment we apply our method to data from FrameNet [3]. FrameNet is a database of manually annotated sentences that provides a mapping from words to semantic frames [6]. Each kind of situation or event is associated with its own frame, such that a sentence like "John lives in Amsterdam" could be annotated as eliciting the frame `Residence`. Frames have elements which correspond to the entities involved with the situation/event, such as "John" being the `Resident` and "Amsterdam" the `Location`.

FrameNet contains information on how frames relate to one another. We use the following set of relations to find a suitable subset of FrameNet:

- `inheritance` - when frames are more specific/generic versions of each other (e.g. `Temporary_stay` inherits from `Residence`);
- `using` - when frames presuppose the existence of each other (e.g. `Colonization` presupposes that the `Colonists` have taken up `Residence` somewhere);
- `subframe` - when frames are part of each other (e.g. `Arrest` and `Arraignment` are part of `Criminal_Process`);
- `perspective_on` - when a frame captures a specific perspective on another (neutral) frame (e.g. `Hiring` and `Get_a_job` are perspectives on `Employment_start`).

To process frames that can be inferred from the presence of other frames even when nothing in the text explicitly alludes to it:

1. count the number of available annotated sentences;
2. create a list of other frames from which this one may be inferred, by finding paths through the directed graph formed by the frame relations listed above; then,
3. discard each frame that cannot be inferred from at least five and at most ten other frames that themselves have at least one hundred annotated sentences.

By requiring there to be at least five frames to infer from, we increase the chances that we are learning to infer this frame rather than some other property that they have in common. We require at most ten to filter out the most generic frames that are inferable from almost any frame. All frames meeting these conditions (*implicit* frames) and the corresponding five to ten frames from which they may be inferred (*explicit* frames) are used to create the labels with which our probes are trained. This process allows us to extract implications like "`Hostile_encounter` $\implies$ `Emotions`" (see also Appendix C).

LTN Probing with FrameNet Each Frame and Frame Element becomes a predicate in the LTN. The attention mechanism, rather than building representations for subject, action, object, builds one representation for each frame element that could be mentioned.

As in the WordNet-based experiments, axioms are introduced to express the relation between the data and the labels, which are used to train the probe. And similarly, additional statements are used at test time to query the consistency of the probe's predictions with our knowledge. In this case, this consists of the implications which have as their antecedent the presence of an explicit frame, and as their consequent an implicit frame whose presence is inferable. The hyperparameters adopted during training are reported in Appendix A.2.

[6] A frame in FrameNet is a structured representation of a situation, including participants, props, and conceptual roles. Each frame contains a textual description (frame definition), associated elements, lexical units, example sentences, and relations with other frames.

4 Design of Model and Probe Structures

For our experiments, we used pretrained embeddings from two large language models: BERT-large (uncased) [10] and OpenLLAMA-7b [16]. We use a 4-bit OPTQ [13] quantized version of OpenLLaMA[7]. The embedding dimension is 1024 and 4096 for BERT-large and OpenLLAMA-7b, respectively. We trained LTP on the extracted embeddings separately for each layer of the LLM to determine the optimal layer to satisfy each predicate. The number of layers are 24 and 32 for BERT-large and OpenLLAMA-7b, respectively. We use standard scaled dot-product attention as is also used in the Transformer architecture [35] on which the LLMs are based. The weights of this mechanism are optimized along with the LTN itself as part of the probe (see Appendix B for more implementation details).

The primary metric used to evaluate the LTP is the satisfiability score, which measures how well the LTN taking as input the LLM's embeddings satisfy the logical statements.

5 Results

Table 1 shows the main experimental results, reporting the average satisfiability of each statement for each dataset. For both LLMs it presents the best-performing layer and the corresponding score. The LTP was compared against two baselines: probes trained with the labels shuffled w.r.t. the features; and a probe trained on completely random features. When the probes outperform these baselines (as they do), it indicates that the representations contain relevant information that the probes are exploiting.

5.1 WordNet

In Table 1 the first five rows reflect the ability of the probe to correctly predict the presence of named entities (ϕ_1), the action performed (ϕ_2), the class(ϕ_3) and macro class of the animal (ϕ_4), and its habitat (ϕ_5). The representations learnt by the LLM contain information to distinguish the subject, the object and the action mentioned by the sentence, and to determine which concepts can appear as the subject or object of a given action. In this case, the higher performance of the random baseline reflects the lower number of classes; likewise, the lower satisfiability of the `isOfClass` predicate could be explained by the higher number of classes.

The second part of the table (rows 5 to 10) show to what extent logical consequences of the atomic formulas mentioned above are correctly computed at inference time. For instance, for the sentence of "Gordon is a German Shepherd", not only should the name ("Gordon"), the action ("is") and the class ("German Shepherd") be correctly predicted, but the probe should be able to infer the

[7] hf.co/TheBloke/open-llama-7b-open-instruct-GPTQ

			BERT			OpenLLaMA			
	Axioms		ℓ	score	shuffled	ℓ	score	shuffled	random
WordNet	isOfName		8	.97 ± .03	.00 ± .00	24	.99 ± .00	.00 ± .00	.00 ± .00
	isOfAction		8	.99 ± .00	.82 ± .00	24	.99 ± .00	.78 ± .02	.82 ± .03
	isOfClass		8	.67 ± .06	.00 ± .00	24	.79 ± .11	.00 ± .00	.00 ± .00
	isOfMacroclass		8	.92 ± .04	.19 ± .00	24	.99 ± .00	.17 ± .03	.19 ± .03
	livesinHabitat		8	.74 ± .03	.12 ± .00	24	.97 ± .03	.08 ± .01	.08 ± .02
	isOfName ∧ isOfAction ∧ isOfClass (ϕ_8)†		8	.66 ± .05	.00 ± .00	24	.74 ± .01	.00 ± .00	.00 ± .00
	isOfName ∧ isOfAction ∧ isOfClass ∧ isOfName ∧ isOfMacroclass (ϕ_9)†		8	.65 ± .05	.00 ± .00	24	.74 ± .01	.00 ± .00	.00 ± .00
	isOfName ∧ isOfAction ∧ isofClass ∧ isOfName ∧ livesinHabitat (ϕ_{10})†		8	.58 ± .05	.00 ± .00	24	.73 ± .02	.00 ± .00	.00 ± .00
FrameNet	Explicit	Frame Elements	8	.77 ± .00	.63 ± .00	4	.75 ± .00	.64 ± .00	.64 ± .00
		Frames	8	.82 ± .00	.46 ± .00	8	.80 ± .01	.49 ± .00	.47 ± .00
		Frames (F1)	8	.76 ± .12	.06 ± .05	8	.73 ± .15	.07 ± .05	.03 ± .05
		Element ⟹ Frame †	12	.98 ± .00	.95 ± .00	12	.98 ± .00	.97 ± .00	.96 ± .00
	Implicit	Frames	8	.72 ± .00	.46 ± .00	8	.71 ± .00	.50 ± .01	.45 ± .00
		Frames (F1)	8	.82 ± .04	.41 ± .11	8	.80 ± .05	.42 ± .10	.34 ± .10
	Explicit Frame ⟹ Implicit Frame †		12	.96 ± .00	.90 ± .00	8	.96 ± .00	.89 ± .01	.94 ± .00

Table 1: Probe performance for WordNet and FrameNet test sets on best-performing layers ℓ, compared to baselines 'shuffled' and 'random'. Mean ± standard deviation (w.r.t. different random initialization of probe weights) of satisfiability (and F1 on rows with '(F1)') for all axioms in the knowledge base are shown. Rows marked with † highlight axioms present only at test time. Scores are also displayed as gradient on cell background color.

macro class ("dog") from the LLM internal representations. Hence, the conjunction of all the predicates (indicated as ϕ_9 in Table 1) should evaluate to true, as indeed shown by our results. The satisfiability of these axioms is not enforced during the training of the LTP, and thus is derived by the logical consistency of the predictions of the probe and thus, by extension, of the LLM internal representations.

In Figure 2f and Figure 2e the results are analyzed on a layer by layer basis. We focus in particular on axioms evaluated only at test time ($\phi_8 - \phi_{10}$), which support the possibility of deductive reasoning. The two investigated models appear to behave quite differently in this respect. For OpenLLAMA, the satisfiability is higher in deeper layers, with a clear gap between layer 4 and 8, and a peak at layer 24. For BERT, on the other hand, consistency appears to be higher in the earliest layers, with a peak around layer 8, whereas the satisfiability is much lower in the deepest layer. Results in Figure 2e show that the LLMs internal representations of syntactic roles are distributed differently across layers in each model. Specifically, OpenLLAMA seems to consolidate this information in its deeper layers, while BERT captures it more effectively in its earlier layers. Comparable to [19,6], our findings demonstrate that different layers of models process and interpret syntactic information and complex concepts at varying depths.

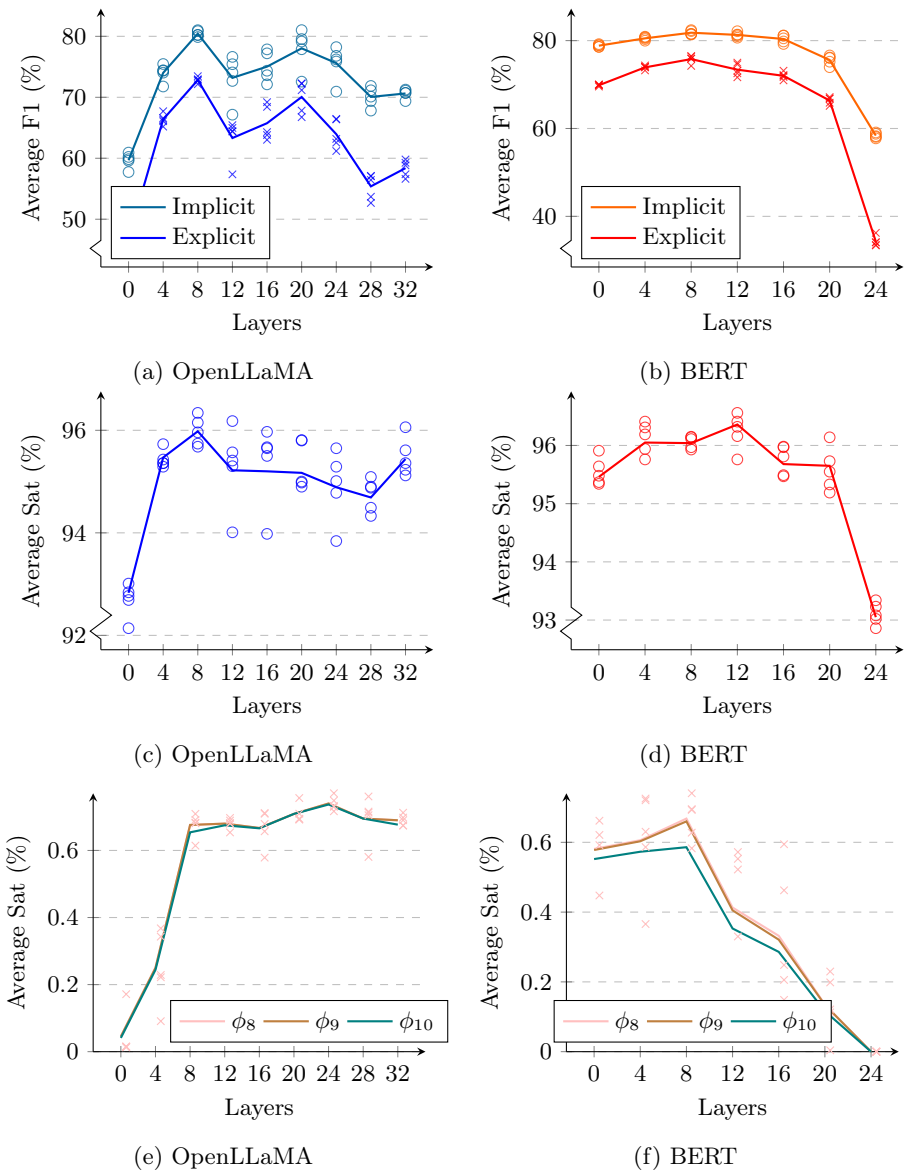

Fig. 2: Probe F1 scores obtained for FrameNet after five runs for each layer on OpenLLaMA (Figure 2a) and BERT (Figure 2b). Probe satisfiability of 'Explicit Frame $\implies$ Implicit Frame' rules for FrameNet obtained after five runs for each layer on OpenLLaMA (Figure 2c) and BERT (Figure 2d). Probe satisfiability of axioms ϕ_8 through ϕ_{10} obtained for WordNet after five runs for each layer on OpenLLaMA (Figure 2e) and BERT (Figure 2f).

5.2 FrameNet

In Table 1 under FrameNet, the first three rows of results show how well the probe is able to correctly predict the explicitly mentioned Frame Elements and Frames. The fourth row shows to what extent the probe is logically consistent with element-frame implications, because for each element mentioned in the input sentence, the frame to which it belongs must also be evaluated true by the LTP. The fifth and sixth lines display the analysis results on implicit frames. The seventh and final row of results shows logical consistency with respect to frame-to-frame implications. We know for each explicit frame if it implies an implicit frame, and thus when the probe predicts the former it must also predict the presence of the latter. Both the element-frame and frame-frame implications show a modest, but significant difference with the baselines (note that for the implications the random guessing baseline is already .75). In Figure 2b and Figure 2a we can see the average F1 scores obtained by the explicit and implicit frames. Both obtain their maximum performance in the same layers for both models. This indicates that on average, the models are not inferring the presence of implicit frames from the presence of explicit frames, as would be the case when reasoning deductively. Rather, it seems that the model reaches the conclusions about both types of frames simultaneously. Figure 2d and Figure 2c displays the satisfaction of the 'Explicit Frame $\implies$ Implicit Frame' rules by layer. For OpenLLaMA we see consistency with these rules reach its maximum in concert with the frame prediction F1, in layer 8. For BERT the consistency is highest in layer 12, rather than layer 8, which shows the best frame prediction performance.

5.3 Discussion

Table 1 shows that identification of semantic roles(classes, actions) and names entities achieves high scores, while the more complex axioms that introduce logical expressions obtain lower results, in line with [31] findings. In Figure 2c and Figure 2d, in line with the findings of [31], we observe that BERT tends to use the lower layers to maximize the satisfiability of the knowledge base. Specifically, in BERT the axioms introduced for WordNet are maximized in layer 8 decreasing significantly in later layers, which is not the case for FrameNet. For OpenLLaMA, satisfiability also sees the largest increase up to layer 8, but does not decrease (as much) after, with WordNet satisfiability peaking in layer 24. The results obtained for WordNet and FrameNet, as supported by [21], may be influenced by the linguistic formalism used in their representation. FrameNet, for example, organizes verbs and their possible interpretations around broader conceptual scenarios called frames, which represent specific situations or events and include the roles that different parts of a sentence can take on. For example, in frame *Arriving*, a sentence like "She arrived at the station", is analyzed by identifying *Theme* (the entity that arrives, "She") and *Goal* (the destination, "the station"). In contrast, WordNet focuses on more specific contexts for individual sentences than these broader frames. The type of information examined can significantly impact the probe's ability to extract relevant information at different layers.

6 Conclusion

We have shown that LTNs can be effectively repurposed to probe large language models. We envision practitioners using this tool to investigate LLMs, and in particular whether they perform certain logical inferences, and how much information needs to be present in its input, before it does so. To accomplish this, they will write a small knowledge base of domain-specific facts and rules for the type of reasoning they are interested in, together with a method for translating a given logical sentence into natural language. Then, a probe may be trained and queried to provide insight into the LLM.

Our WordNet experiments revealed that LLM layers differ in capturing logical predicates, with deeper layers generally performing better on complex predicates. For instance, satisfiability scores of axioms ϕ_8, ϕ_9, and ϕ_{10} showed that deeper layers (e.g., layer 24 in OpenLLAMA) were more effective. This suggests that as the model progresses, it refines abstract representations crucial for complex predicates. In BERT, early to middle layers (e.g., layer 8) perform better, while in OpenLLAMA, deeper layers (e.g., layer 24) are more effective. In FrameNet experiments, it appears that the model simultaneously arrives at conclusions about both types of frames. This may indicate that information relevant to identify semantic roles emerges within these specific layers. This versatility makes LTPs valuable for applications like natural language understanding, automated reasoning, and decision-making systems. Insights from LTPs can guide the development of future LLMs, helping to design models with better logical capabilities, leading to more robust and interpretable AI systems.

We believe that our approach can be further extended in several ways. For example, it is possible to introduce other datasets to investigate other kinds of logical deductions. The knowledge base could also be extended to consider other types of relationships, beyond class hierarchies or geographical location. Finally, several kinds of large language models can be used.

References

1. Azaria, A., Mitchell, T.: The internal state of an LLM knows when it's lying. In: The 2023 Conference on Empirical Methods in Natural Language Processing (2023)
2. Badreddine, S., Garcez, A.d., Serafini, L., Spranger, M.: Logic tensor networks. Artificial Intelligence **303**, 103649 (2022)
3. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The Berkeley FrameNet Project. In: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1. pp. 86–90. Association for Computational Linguistics, Montreal, Quebec, Canada (Aug 1998). https://doi.org/10.3115/980845.980860
4. Bang, Y., Cahyawijaya, S., Lee, N., Dai, W., Su, D., Wilie, B., Lovenia, H., Ji, Z., Yu, T., Chung, W., et al.: A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity. In: Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd

Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 675–718 (2023)
5. Belinkov, Y.: Probing Classifiers: Promises, Shortcomings, and Advances. Computational Linguistics **48**(1), 207–219 (Apr 2022). https://doi.org/10.1162/coli_a_00422, https://doi.org/10.1162/coli_a_00422
6. Bronzini, M., Nicolini, C., Lepri, B., Staiano, J., Passerini, A.: Unveiling llms: The evolution of latent representations in a temporal knowledge graph. ArXiv **abs/2404.03623** (2024), https://api.semanticscholar.org/CorpusID:268889716
7. Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y.T., Li, Y., Lundberg, S., et al.: Sparks of artificial general intelligence: Early experiments with GPT-4. arXiv preprint arXiv:2303.12712 (2023)
8. Carraro, T.: LTNtorch: PyTorch implementation of Logic Tensor Networks (mar 2022), https://doi.org/10.5281/zenodo.6394282
9. Conia, S., Navigli, R.: Probing for Predicate Argument Structures in Pretrained Language Models. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 4622–4632. Association for Computational Linguistics, Dublin, Ireland (May 2022). https://doi.org/10.18653/v1/2022.acl-long.316, https://aclanthology.org/2022.acl-long.316
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: North American Chapter of the Association for Computational Linguistics (2019), https://api.semanticscholar.org/CorpusID:52967399
11. Ettinger, A., Elgohary, A., Resnik, P.: Probing for semantic evidence of composition by means of simple classification tasks. In: Proceedings of the 1st workshop on evaluating vector-space representations for nlp. pp. 134–139 (2016)
12. Fellbaum, C.: WordNet: An Electronic Lexical Database. Bradford Books (1998), https://mitpress.mit.edu/9780262561167/
13. Frantar, E., Ashkboos, S., Hoefler, T., Alistarh, D.: OPTQ: Accurate quantization for generative pre-trained transformers. In: The Eleventh International Conference on Learning Representations (2022)
14. Galassi, A., Lippi, M., Torroni, P.: Investigating Logic Tensor Networks for Neural-Symbolic Argument Mining (2021)
15. Ganesh, P., Chen, Y., Lou, X., Khan, M.A., Yang, Y., Chen, D., Winslett, M., Sajjad, H., Nakov, P.: Compressing large-scale transformer-based models: A case study on BERT. Transactions of the Association for Computational Linguistics **9**, 1061–1080 (2020), https://api.semanticscholar.org/CorpusID:211532645
16. Geng, X., Liu, H.: OpenLLaMA: An open reproduction of LLaMA (May 2023), https://github.com/openlm-research/open_llama
17. Huang, J., Chang, K.C.C.: Towards reasoning in large language models: A survey. arXiv preprint arXiv:2212.10403 (2022)
18. Hupkes, D., Veldhoen, S., Zuidema, W.: Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. Journal of Artificial Intelligence Research **61**, 907–926 (2018)
19. Jin, M., Yu, Q., Huang, J., Zeng, Q., Wang, Z., Hua, W., Zhao, H., Mei, K., Meng, Y., Ding, K., Yang, F., Du, M., Zhang, Y.: Exploring concept depth: How large language models acquire knowledge at different layers? ArXiv **abs/2404.07066** (2024), https://api.semanticscholar.org/CorpusID:269033222
20. Kosinski, M.: Theory of mind may have spontaneously emerged in large language models. Tech. rep., Stanford University, Graduate School of Business (2023)

21. Kuznetsov, I., Gurevych, I.: A matter of framing: The impact of linguistic formalism on probing results. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 171–182. Association for Computational Linguistics, Online (Nov 2020). https://doi.org/10.18653/v1/2020.emnlp-main.13, https://aclanthology.org/2020.emnlp-main.13
22. Kyriakopoulos, S., d'Avila Garcez, A.S.: Continual reasoning: Non-monotonic reasoning in neurosymbolic AI using continual learning. In: International Workshop on Neural-Symbolic Learning and Reasoning (2023), https://api.semanticscholar.org/CorpusID:258461140
23. Liu, Y.H., He, H., Han, T., Zhang, X., Liu, M., Tian, J., Zhang, Y., Wang, J., Gao, X., Zhong, T., Pan, Y., Xu, S., Wu, Z., Liu, Z., Zhang, X., Zhang, S., Hu, X., Zhang, T., Qiang, N., Liu, T., Ge, B.: Understanding llms: A comprehensive overview from training to inference. ArXiv **abs/2401.02038** (2024), https://api.semanticscholar.org/CorpusID:266755678
24. Manigrasso, F., Miro, F.D., Morra, L., Lamberti, F.: Faster-LTN: a neuro-symbolic, end-to-end object detection architecture. In: International Conference on Artificial Neural Networks. Springer (2021)
25. Manigrasso, F., Morra, L., Lamberti, F.: Fuzzy logic visual network (FLVN): A neuro-symbolic approach for visual features matching. In: International Conference on Image Analysis and Processing. pp. 456–467. Springer (2023)
26. Morra, L., Azzari, A., Bergamasco, L., Braga, M., Capogrosso, L., Delrio, F., Giacomo, G.D., Eiraudo, S., Ghione, G., Giudice, R., Koudounas, A., Piano, L., Rege, D., Cambrin, Risso, M., Rondina, M., Russo, A.S., Russo, M., Taioli, F., Vaiani, L., Vercellino, C.: Designing logic tensor networks for visual sudoku puzzle classification. In: International Workshop on Neural-Symbolic Learning and Reasoning (2023)
27. OpenAI, R.: GPT-4 technical report. arXiv pp. 2303–08774 (2023)
28. Padalkar, P., Wang, H., Gupta, G.: NeSyFOLD: Extracting logic programs from convolutional neural networks. In: ICLP Workshops (2023), https://api.semanticscholar.org/CorpusID:263875519
29. Serafini, L., Garcez, A.d.: Logic tensor networks: Deep learning and logical reasoning from data and knowledge. arXiv preprint arXiv:1606.04422 (2016)
30. Shapira, N., Levy, M., Alavi, S.H., Zhou, X., Choi, Y., Goldberg, Y., Sap, M., Shwartz, V.: Clever hans or neural theory of mind? stress testing social reasoning in large language models. arXiv preprint arXiv:2305.14763 (2023)
31. Tenney, I., Das, D., Pavlick, E.: BERT rediscovers the classical NLP pipeline. In: Annual Meeting of the Association for Computational Linguistics (2019), https://api.semanticscholar.org/CorpusID:155092004
32. Tenney, I., Xia, P., Chen, B., Wang, A., Poliak, A., McCoy, R.T., Kim, N., Durme, B.V., Bowman, S.R., Das, D., Pavlick, E.: What do you learn from context? Probing for sentence structure in contextualized word representations (Sep 2018), https://openreview.net/forum?id=SJzSgnRcKX
33. Thawani, A., Ghanekar, S., Zhu, X., Pujara, J.: Learn your tokens: Word-pooled tokenization for language modeling. In: Findings of the Association for Computational Linguistics: EMNLP 2023. pp. 9883–9893 (2023)
34. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. Journal of Machine Learning Research (11) (2008)
35. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is All you Need. In: Advances in Neural Information Processing Systems. vol. 30. Curran Asso-

ciates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
36. Zhang, B., Huang, X., Huang, Z., Huang, H., Zhang, B., Fu, X., Jing, L.: Sentiment Interpretable Logic Tensor Network for Aspect-Term Sentiment Analysis. In: Proceedings of the 29th International Conference on Computational Linguistics. pp. 6705–6714. International Committee on Computational Linguistics, Gyeongju, Republic of Korea (Oct 2022), https://aclanthology.org/2022.coling-1.582
37. Zhang, Y., Li, Y., Cui, L., Cai, D., Liu, L., Fu, T., Huang, X., Zhao, E., Zhang, Y., Chen, Y., et al.: Siren's song in the AI ocean: A survey on hallucination in large language models. arXiv preprint arXiv:2309.01219 (2023)

A HYPERPARAMETERS

For all experiments, the aggregation function parameter in the knowledge base defined in Equation 27 is set to $p_\forall = 2$. We implemented the probes in PyTorch, using the LTNtorch library [8] and training was done using the Adam optimizer.

A.1 WordNet

To train the WordNet probes, we adopted a learning rate of 1e-5 and trained our architecture for 100 epochs. The knowledge base during training included batches of 128 randomly chosen sentences, both positive and negative. Experiments were done on a single Nvidia 2080 Ti GPU.

A.2 FrameNet

To train the FrameNet probes, we adopted a learning rate of 1e-3 and trained our architecture for 15 epochs and a batch size of 128. The experiments were performed on a single Nvidia 3090 GPU.

B WORDNET LTN

B.1 Variables, Predicates and axioms

In this subsection, we present the foundations of variables and predicates with the definition of the knowledge base $\mathcal{K}$.

Grounding Variables and their corresponding domains are grounded as follows:

$$\begin{aligned}
&\mathcal{G}(n) = \mathbb{N}^N, \;\; \mathcal{G}(v) = \mathbb{N}^V, \;\; \mathcal{G}(g) = \mathbb{N}^G, \;\; \mathcal{G}(q) = \mathbb{N}^Q, \\
&\mathcal{G}(h) = \mathbb{N}^H, \;\; \mathcal{G}(l) = \mathbb{N}^L, \;\; \mathcal{G}(s) = \mathbb{R}^{B\times D}, \\
&\mathcal{G}(sub),\, \mathcal{G}(act),\, \mathcal{G}(obj) = \mathbb{R}^{T\times m}
\end{aligned} \tag{11}$$

where g, q, n, a, h represent the class labels belonging to a set of classes G, macroclasses Q, names N, actions A, and habitats H sets. Additionally, l represents the label used to distinguish different sections of the sentence, such as subject and object. The variables B and D are used in the definition of s to represent the dimensions of the feature space, where B is the batch size and D is the dimensionality of the features.

On the other hand *sub*, *act* and *obj* retrieved with the attention model of an entire sentence s, are grounded into a feature space. *obj* can be translated into macroclass when the sentence is in the form `isOfName isOfAction isOfMacroclass` instead of `isOfName isOfAction isofObject`. Within the FOL language, different predicates are defined: `isOfName(sub, n)`, `isOfAction(act, a)`, `isOfClass(obj, c)` with the aim of categorizing the sentence components, `livesInHabitat(obj, h)` and `isOfMacroclass(obj, q)` allow classifying objects respect to habitat and macroclass and finally `isSubject(sub, l)` with `isObject(obj, l)` to predict if a sentence is logically true by investigating whether the subject and its complement are inserted in a correct order.

The predicate groundings $\mathcal{G}$(`isOfName`), $\mathcal{G}$(`isOfAction`), $\mathcal{G}$(`isOfClass`) are formed by the similarity between the input features and the corresponding trainable class vectors with an MLP p_1 with softmax activation function, as the classes are mutually exclusive.

$$\mathcal{G}(\texttt{isOfName}) :\text{sub}, n \rightarrow n^T p_1(\mathcal{G}(sub), n) \tag{12}$$

$$\mathcal{G}(\texttt{isOfAction}) :\text{act}, v \rightarrow v^T p_1(\mathcal{G}(act), v) \tag{13}$$

$$\mathcal{G}(\texttt{isOfClass}) :\text{obj}, g \rightarrow g^T p_1(\mathcal{G}(obj), g) \tag{14}$$

Likewise, the $\mathcal{G}$(`isOfMacroclass`), $\mathcal{G}$(`livesInHabitat`) predicates are grounded with two simple MLP layers with a softmax activation function p_2, p_3:

$$\mathcal{G}(\texttt{isOfMacroclass}) :\text{obj}, q \rightarrow v^T p_2(\mathcal{G}(obj), q) \tag{15}$$

$$\mathcal{G}(\texttt{isOfMacroclass}) :\text{obj}, q \rightarrow v^T p_2(\mathcal{G}(obj), q) \tag{16}$$

$$\mathcal{G}(\texttt{livesInHabitat}) :\text{obj}, h \rightarrow g^T p_3(\mathcal{G}(obj), h) \tag{17}$$

Finally, for the grounding of `isSubject` and `isObject` two parametric similarity functions based on two simple MLP layers with softmax activation function p_4, p_5:

$$\mathcal{G}(\texttt{isSubject}) :\text{sub}, l \rightarrow l^T p_4(\mathcal{G}(sub), l) \tag{18}$$

$$\mathcal{G}(\texttt{isObject}) :\text{obj}, l \rightarrow l^T p_5(\mathcal{G}(obj), l) \tag{19}$$

where l allows us to distinguish the different sections (subject, action, object) of the sentence.

B.2 Learning from labeled examples

The following is each axiom representing the labeled examples used during training:

$$\phi_1 = \forall \text{Diag}(sub, n)(\texttt{isOfName}(sub, n)) \tag{20}$$
$$\phi_2 = \forall \text{Diag}(act, v)(\texttt{isOfAction}(act, v)) \tag{21}$$
$$\phi_3 = \forall \text{Diag}(obj, g)(\texttt{isOfClass}(obj, g)) \tag{22}$$
$$\phi_4 = \forall \text{Diag}(obj, q)(\texttt{isOfMacroclass}(obj, q)) \tag{23}$$
$$\phi_5 = \forall \text{Diag}(obj, h)(\texttt{livesInHabitat}(obj, h)) \tag{24}$$
$$\phi_6 = \forall \text{Diag}(sub, l)(\texttt{isSubject}(sub, l)) \tag{25}$$
$$\phi_7 = \forall \text{Diag}(obj, l)(\texttt{isObject}(obj, l)) \tag{26}$$

B.3 Grounding logical connectives and aggregators.

The knowledge base $\mathcal{K}$ includes a collection of axioms, with a view to reconstructing logical connectives and aggregators into Real Logic. Gradient descent is adopted to train the probe to maximize the satisfiability of the problem. In this configuration, we make use of the standard negation $\neg$ defined as $N_S(a) = 1 - a$, and the Reichenbach implication $\rightarrow$ defined as $I_R(a, b) = 1 - a + ab$, where a and b are both truth values within the range of $[0, 1]$. To approximate the universal quantifier $\forall$, we use the generalized mean with respect to the error, denoted as A_{pME}, as described in [2,34]. Given a set of n truth values $a_1, \ldots, a_n \in [0, 1]$:

$$\forall : A_{pME}\left(a_1, \ldots, a_n\right) = \\ 1 - \left(\frac{1}{n}\sum_{i=1}^{n}\left(1 - a_i\right)^{p_\forall}\right)^{\frac{1}{p_\forall}} \quad p_\forall \geqslant 1 \tag{27}$$

A_{pME} is a measure of how much, on average, truth values a_i deviate from the true value of 1. Further details on the role of $p_\forall$ can be found in [2].

B.4 Knowledge base for inference time

The knowledge base used at testing time is shown below:

$$\begin{aligned}
\phi_8 = \forall \mathrm{Diag}(&sub, n, act, v, obj, g, l) \\
&(\texttt{isOfName}(sub, n) \wedge \texttt{isSubject}(sub, l)) \wedge \texttt{isOfAction}(act, v) \\
\wedge\, &(\texttt{isOfClass}(obj, g) \wedge \texttt{isObject}(obj, l)) \qquad (28) \\
\phi_9 = \forall \mathrm{Diag}(&sub_1, n, act_1, v, obj_1, g, l, sub_2, act_2, obj_2) \\
&(\texttt{isOfName}(sub_1, n) \wedge \texttt{isSubject}(sub_1, l)) \wedge \texttt{isOfAction}(act_1, v) \\
\wedge\, &(\texttt{isOfClass}(obj_1, g) \wedge \texttt{isObject}(obj_1, l)) \\
\wedge\, &(\texttt{isOfName}(sub_2, n) \wedge \texttt{isSubject}(sub_2, l)) \wedge \\
&(\texttt{isOfMacroclass}(obj_2, q) \wedge \texttt{isObject}(obj_2, l)) \qquad (29) \\
\phi_{10} = \forall \mathrm{Diag}(&sub_1, n, act_1, v, obj_1, g, l, sub_2, obj_2) \\
&(\texttt{isOfName}(sub_1, n) \wedge \texttt{isSubject}(sub_1, l)) \wedge \texttt{isOfAction}(act_1, v) \\
\wedge\, &(\texttt{isOfClass}(obj_1, g) \wedge \texttt{isObject}(obj_1, l)) \\
\wedge\, &((\texttt{isOfName}(sub_2, n) \wedge \texttt{isSubject}(sub_2, l))) \\
\wedge\, &(\texttt{livesInHabitat}(obj_2, h)) \wedge \texttt{isObject}(obj_2, l)) \qquad (30)
\end{aligned}$$

The subscript introduced in the annotation allows us to identify whether the extracted representations come from the previous sentence or the following one. Axiom ϕ_8 represent the degree of truthiness of a sentence like "Gordon is a German Shepherd", axiom ϕ_9 express "Gordon is a German Shepard and Gordon is Carnivorous", axiom ϕ_{10} highlights "Gordon is a German Shepard and Gordon lives in a domestic environment".

C FRAMENET DATA DETAILS

C.1 Frame-frame implications

The following is the complete list of all implications included with the FrameNet experiments. For more information on the meaning of each of the frames, please see `https://framenet.icsi.berkeley.edu/frameIndex`.

```
                  f_192_Locale_(asserted)->
           f_196_Locale_by_use_(asserted)->
         f_191_Natural_features_(asserted)->
                f_173_Buildings_(asserted)->
                 f_203_Roadways_(asserted)->
                                        -> f_960_Being_located_(implied)

                  f_192_Locale_(asserted)->
            f_196_Locale_by_use_(asserted)->
                  f_54_Arriving_(asserted)->
f_1178_Interior_profile_relation_(asserted)->
         f_199_Locative_relation_(asserted)->
          f_191_Natural_features_(asserted)->
                f_173_Buildings_(asserted)->
                 f_203_Roadways_(asserted)->
                                           -> f_660_Existence_(implied)
```

```
            f_196_Locale_by_use_(asserted)->
          f_191_Natural_features_(asserted)->
                 f_173_Buildings_(asserted)->
                  f_203_Roadways_(asserted)->
                                                  -> f_192_Locale_(implied)

                   f_429_Vehicle_(asserted)->
                    f_426_Weapon_(asserted)->
                     f_106_Gizmo_(asserted)->
                 f_173_Buildings_(asserted)->
                  f_421_Artifact_(asserted)->
                      f_298_Text_(asserted)->
                                                  -> f_1141_Using_(implied)

                   f_513_Purpose_(asserted)->
                   f_54_Arriving_(asserted)->
                   f_5_Causation_(asserted)->
                  f_1158_Project_(asserted)->
                   f_1346_Supply_(asserted)->
                                       -> f_88_Eventive_affecting_(implied)

                    f_192_Locale_(asserted)->
            f_196_Locale_by_use_(asserted)->
                   f_54_Arriving_(asserted)->
f_1178_Interior_profile_relation_(asserted)->
          f_191_Natural_features_(asserted)->
                 f_173_Buildings_(asserted)->
                  f_203_Roadways_(asserted)->
                                       -> f_199_Locative_relation_(implied)

                   f_513_Purpose_(asserted)->
                    f_424_Attack_(asserted)->
                   f_5_Causation_(asserted)->
          f_93_Hostile_encounter_(asserted)->
                  f_1158_Project_(asserted)->
                                    -> f_720_Cognitive_connection_(implied)

                 f_660_Existence_(asserted)->
                    f_192_Locale_(asserted)->
            f_196_Locale_by_use_(asserted)->
                   f_54_Arriving_(asserted)->
f_1178_Interior_profile_relation_(asserted)->
         f_199_Locative_relation_(asserted)->
          f_191_Natural_features_(asserted)->
                 f_173_Buildings_(asserted)->
                  f_203_Roadways_(asserted)->
                             -> f_2100_Cycle_of_existence_scenario_(implied)

                    f_192_Locale_(asserted)->
            f_196_Locale_by_use_(asserted)->
                   f_54_Arriving_(asserted)->
f_1178_Interior_profile_relation_(asserted)->
         f_199_Locative_relation_(asserted)->
          f_191_Natural_features_(asserted)->
                 f_173_Buildings_(asserted)->
                  f_203_Roadways_(asserted)->
                                       -> f_2790_Locative_scenario_(implied)

                   f_513_Purpose_(asserted)->
                   f_54_Arriving_(asserted)->
                   f_5_Causation_(asserted)->
                  f_1158_Project_(asserted)->
                   f_1346_Supply_(asserted)->
                                    -> f_2961_Undergoing_scenario_(implied)

                   f_513_Purpose_(asserted)->
                    f_424_Attack_(asserted)->
```

```
          f_5_Causation_(asserted)->
  f_93_Hostile_encounter_(asserted)->
         f_1158_Project_(asserted)->
                          -> f_2902_Conditional_scenario_(implied)

       f_114_Likelihood_(asserted)->
              f_972_Age_(asserted)->
            f_2282_Size_(asserted)->
         f_1152_Research_(asserted)->
            f_424_Attack_(asserted)->
  f_93_Hostile_encounter_(asserted)->
         f_80_Frequency_(asserted)->
          f_31_Scrutiny_(asserted)->
       f_990_Capability_(asserted)->
     f_354_Desirability_(asserted)->
                                     -> f_964_Attributes_(implied)

       f_114_Likelihood_(asserted)->
              f_972_Age_(asserted)->
            f_2282_Size_(asserted)->
         f_1152_Research_(asserted)->
            f_424_Attack_(asserted)->
  f_93_Hostile_encounter_(asserted)->
         f_80_Frequency_(asserted)->
          f_31_Scrutiny_(asserted)->
       f_990_Capability_(asserted)->
     f_354_Desirability_(asserted)->
                           -> f_1015_Gradable_attributes_(implied)

          f_513_Purpose_(asserted)->
         f_366_Desiring_(asserted)->
           f_424_Attack_(asserted)->
  f_93_Hostile_encounter_(asserted)->
         f_1158_Project_(asserted)->
     f_354_Desirability_(asserted)->
                                       -> f_1712_Emotions_(implied)

          f_513_Purpose_(asserted)->
         f_366_Desiring_(asserted)->
           f_424_Attack_(asserted)->
  f_93_Hostile_encounter_(asserted)->
         f_1158_Project_(asserted)->
     f_354_Desirability_(asserted)->
                                 -> f_48_Experiencer_focus_(implied)

          f_429_Vehicle_(asserted)->
           f_426_Weapon_(asserted)->
            f_106_Gizmo_(asserted)->
        f_173_Buildings_(asserted)->
             f_298_Text_(asserted)->
                                       -> f_421_Artifact_(implied)

           f_192_Locale_(asserted)->
    f_196_Locale_by_use_(asserted)->
  f_191_Natural_features_(asserted)->
         f_173_Buildings_(asserted)->
          f_203_Roadways_(asserted)->
                                       -> f_329_Boundary_(implied)

           f_192_Locale_(asserted)->
    f_196_Locale_by_use_(asserted)->
  f_191_Natural_features_(asserted)->
         f_173_Buildings_(asserted)->
          f_203_Roadways_(asserted)->
                                    -> f_237_Bounded_entity_(implied)

       f_1321_Substance_(asserted)->
  f_1322_Active_substance_(asserted)->
```

```
          f_192_Locale_(asserted)->
   f_196_Locale_by_use_(asserted)->
f_191_Natural_features_(asserted)->
       f_173_Buildings_(asserted)->
        f_203_Roadways_(asserted)->
                                -> f_302_Physical_entity_(implied)

         f_513_Purpose_(asserted)->
          f_424_Attack_(asserted)->
         f_5_Causation_(asserted)->
f_93_Hostile_encounter_(asserted)->
        f_1158_Project_(asserted)->
                                 -> f_2950_Alternativity_(implied)
```