# Improved Logical Reasoning of Language Models via Differentiable Symbolic Programming

**Hanlin Zhang** [*1] **Jiani Huang** [*2] **Ziyang Li** [2] **Mayur Naik** [2] **Eric Xing** [1 3]

## Abstract

Pre-trained large language models (LMs) struggle to perform logical reasoning reliably despite advances in scale and compositionality. In this work, we tackle this challenge through the lens of symbolic programming. We propose DSR-LM, a **D**ifferentiable **S**ymbolic **R**easoning framework where pre-trained **LM**s govern the perception of factual knowledge, and a symbolic module equipped with provenance generates top-k proofs by deductive reasoning. In contrast to works that rely on hand-crafted logic rules, our differentiable symbolic reasoning architecture efficiently learns weighted rules to further improve LMs. DSR-LM is scalable, interpretable, and allows easy integration of prior knowledge, thereby supporting extensive symbolic programming to robustly derive a logical conclusion. Our experiments show that DSR-LM leads to improved logical reasoning of pre-trained LMs and outperforms a spectrum of competitive baselines even under systematic distribution shifts on sequence lengths.

## 1. Introduction

Pre-trained LMs have demonstrated remarkable predictive performance but they tend to generate harmful, inconsistent, untruthful, or toxic outputs (Weidinger et al., 2021), in part because they largely lack capabilities of systematic understanding and reasoning about factual knowledge (Elazar et al., 2021; Hase et al., 2021; Valmeekam et al., 2022). In particular, several works have observed that logical inconsistency is a significant weakness of LMs (Kassner et al., 2020; Helwe et al., 2021; Creswell et al., 2022) even in a confined problem space (Zhang et al., 2022). Such observations are notable evidence that pre-trained LMs are far from human-level understanding.

In this work, we seek to tackle the above challenges and focus on improving the deductive logical reasoning process of LMs. Our approach has the same key objectives as neuro-symbolic programming (Chaudhuri et al., 2021): compositionality, consistency, interpretability and easy integration of prior knowledge. Existing neuro-symbolic works (Rocktäschel & Riedel, 2017; Manhaeve et al., 2018; Zhang et al., 2019) target various confined problem spaces and are therefore not suitable for improving pre-trained LMs. To this end, we develop DSR-LM, which supports a tight integration of pre-trained LMs and differentiable symbolic reasoning to leverage the advantages of both worlds.

DSR-LM integrates LMs with differentiable symbolic programming in a principled manner: the LM governs the perception of `(relation, subject, object)` tuples, and the symbolic program efficiently and deductively reasons with desirable encoded logical rules to generate the top-k proofs via forward chaining for predicting the truth of a query tuple. Specifically, the LM predicts the pairwise relations of entities mentioned in each input sentence. With those probabilistic input facts, the symbolic module leverages the framework of provenance for deductive databases (Cheney et al., 2009) to generate top-k proofs in a compositional manner for rule-based logic operations in Datalog. To overcome the common limitation of reliance on human-crafted logical rules (Huang et al., 2021; Nye et al., 2021), we propose to efficiently learn weighted rules from noisy inputs in an end-to-end differentiable manner.

We conduct extensive experiments showing that DSR-LM can consistently improve the logical reasoning capability of pre-trained LMs. Moreover, we show that DSR-LM can induce logic rules that are amenable to human understanding to explain decisions given only a few schemas. As generalization over long-range dependency is a significant weakness of transformer-based language models (Lake & Baroni, 2018; Tay et al., 2020), we highlight that in systematic, long-context scenarios, where most pre-trained or neural approaches fail to generalize compositionally, our model can still achieve considerable performance gains.

[*]Equal contribution [1]Carnegie Mellon University [2]University of Pennsylvania [3]MBZUAI. Correspondence to: Hanlin Zhang <hanlinzh@cs.cmu.edu>, Jiani Huang <jianih@seas.upenn.edu>.

*First Workshop of Pre-training: Perspectives, Pitfalls, and Paths Forward at ICML 2022*, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

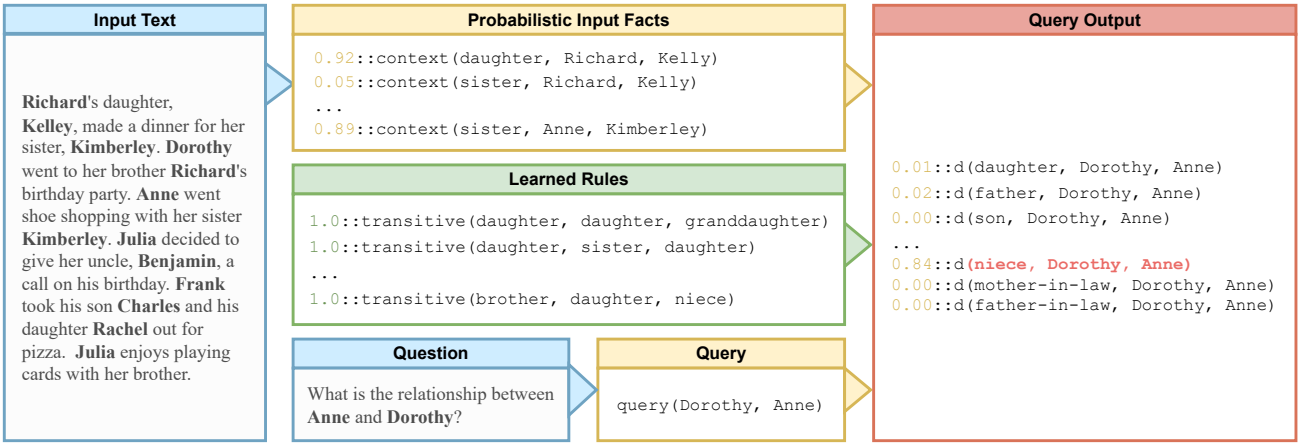


*Figure 1.* A motivating example where "Anne is the niece of Dorothy" should be logically inferred from the context.

## 2. Related Works

**Logical Reasoning with LMs.** Pre-trained LMs have been shown to struggle with logical reasoning over factual knowledge (Kassner et al., 2020; Helwe et al., 2021; Talmor et al., 2020a). For example, LMs cannot consistently predict the answer $fly$ given the queries with the same meaning: "Birds can [MASK]" and "Birds are able to [MASK]"; LMs also tend to fail catastrophically with logical relationships, e.g. rules including negation "(giraffe, is, tall) $\Longleftrightarrow$ (giraffe, is not, short) (Kassner et al., 2020). There is encouraging recent progress in using transformers for reasoning tasks (Zhou et al., 2020; Clark et al., 2021; Wei et al., 2022; Chowdhery et al., 2022; Zelikman et al., 2022) but these approaches usually require a significant amount of computation for re-training and human annotations on reasoning provenance (Zhou et al., 2020; Wei et al., 2022). Moreover, their entangled nature with natural language makes it fundamentally hard to achieve robust inference over symbolic factual knowledge (Greff et al., 2020).

There are other obvious remedies for LMs' poor reasoning capability. Ensuring that the training corpus contains a sufficient amount of exemplary episodes of sound reasoning reduces the dependency on normative biases and annotation artifacts (Talmor et al., 2020b; Betz et al., 2020; Hase et al., 2021). Heuristics like data augmentation are also shown to be effective (Talmor et al., 2020b). But the above works require significant efforts for crowdsourcing and auditing training data. Our method handily encodes a few prototypes/templates of logic rules and is thus more efficient in terms of human effort. Moreover, our goal is fundamentally different from theirs in investigating the tight integration of neural and symbolic models in an end-to-end manner.

**Neuro-Symbolic Reasoning.** Neuro-symbolic approaches are proposed to integrate the perception of deep neural components and the reasoning of symbolic components. Representative works can be briefly categorized into regularization (Xu et al., 2018), program synthesis (Mao et al., 2018), and proof-guided probabilistic programming (Rocktäschel & Riedel, 2017; Manhaeve et al., 2018; Zhang et al., 2019; Huang et al., 2021). To improve compositionality of LMs over natural language, previous works propose to parameterize grammatical rules (Kim, 2021; Shaw et al., 2021) but show that those hybrid models are inefficient and usually underperform neural counterparts. In contrast to the above works, we focus on improving LMs' reasoning over logical propositions with tight integration of their pre-trained knowledge.

## 3. Method

**Problem Formulation.** Each data-point in the dataset is a 3-tuple containing context $c$, query $q$, and the answer $r$. Figure 1 shows an instance which we will use as our running example. The context $c$ is a sequence of natural language sentences within which there will be a set of entities, possibly referenced by 3rd person pronouns. These sentences hint at the relationships between entities. For example, "Diane is taking her brother Brad out for a late dinner" implies that Brad is Diane's brother and Diane is Brad's sister. The query $q$, on the other hand, is a tuple of two entities, denoting the relationship between whom we want to infer. The expected relation is stored in the answer $r$, which will be one of a confined set of possible relations $R$, allowing us to treat the whole problem as an $|R|$-way classification problem. We focus only on the problems where the desired relation is not explicitly stated in the context, but need to be deduced through a sequence of reasoning. The derivation process for our running example is demonstrated in the proof tree shown below:

$$\frac{\dfrac{\text{brother(Dorothy, Richard)} \quad \text{daughter(Richard, Kelly)}}{\text{niece(Dorothy, Kelly)}} \quad \text{sister(Kelly, Anne)}}{\text{niece(Dorothy, Anne)}}$$

This problem poses the following major challenges: (i) The way in which relationships are expressed can vary a lot in natural language, indicating a need for large pre-trained LM to understand the meaning of these phrases. For example, one might use "John, the father's father of Alice, is taking her to..." to imply that John is the grandfather of Alice. (ii) To predict the relationship between the two entities in the desired query, multi-hop reasoning is needed. In fact, the number of facts needed to deduce the final outcome is denoted by a parameter $k$. Usually, the larger the $k$ is, the longer the context will be, and the reasoning will be even harder. (iii) Further, to test the ability for systematic generalization, we train on contexts with $k = 2$ and $k = 3$ while we test our model on longer contexts upto $k = 10$. (iv) We might not possess a knowledge base where the rules such as "father's father is grandfather" are explicitly written, requiring us to learn the rules along with the relation extraction. (v) During training, unlike the common practice of many previous neuro-symbolic approaches (Mao et al., 2018), we do not obtain supervision for any intermediate relation: only the final answer relations for the queries are used for training. This imposes additional challenges to LMs since we do not have intermediate weak supervisions for modeling long-context dependencies in our settings.

**Methodology Overview.** To overcome the difficulties listed above, we make the following design decisions. First, we adopt and fine-tune a large pre-trained LM such as RoBERTa to perform the basic relation prediction over shorter windowed contexts. This way, we can ensure that LMs are reasoning about similar-length-context when $k$ becomes larger during test time. From here, we generate a probabilistic database containing relations extracted from natural language context and rules that are either to-be-learnt or obtained from a knowledge base. This database is then fed to a differentiable logical reasoning engine that will perform multi-hop reasoning that yields a predicted relation distribution over $R$. It allows us to pass predicted results into a loss function such as binary-cross-entropy (BCE) to be compared with the ground-truth at the end. Since the logical reasoning module is fully differentiable, we obtain an end-to-end training loop that can back-propagate gradients to fine-tune the LM and learn the rules with suitable learning objectives. Figure 1 provides a conceptual illustration of the workflow.

**Neural Perception.** Since LMs have strong pattern recognition capabilities for tasks like Named-Entity-Recognition (NER) and Relation Extraction (RE) (Tenney et al., 2019; Soares et al., 2019), we adopt them as our neural components in DSR-LM. The goal of perception of LMs is to extract relations of mentioned entities, which obtains a distribution of relations between every pair of entities in each windowed context. We assume a finite set of relations $R$ to be classified. For example in CLUTRR (Sinha et al., 2019), we have 20 kinship relations including mother, son, uncle, father-in-law, etc. The entity pairs are directional. That is, the relation of *Diane* to *Brad* is not the same as the relation of *Brad* to *Diane*. Consequently, if a windowed context contains $n$ distinct names, we will predict $n(n-1)$ individual distributions of relations. Note that, for a given pair of entities there might not be any implied relationship between them. Therefore we would need to employ a $(|R|+1)$-way classification where the additional class represents "no relation" between the two entities.

In practice, the windowed contexts are split based on simple heuristics of "contiguous one to three sentences that contain at least two entities", to account for coreference resolution. The windowed contexts can be overlapping and we allow the reasoning module to deal with noisy and redundant data. Our relation extraction module is implemented by adding an MLP classifier after the LM, accepting a concatenation of the embedding of the two entities and the aggregated embedding of the whole windowed context.

**Symbolic Reasoning.** The symbolic reasoning module combines the information from multiple windowed texts and deduces the answer for the query by performing multi-hop reasoning. Conceptually, we represent a relation between two entities extracted from the context as an arity-3 *fact*, `context(r, s, o)`, to denote that the *object* $o$ is related to *subject* $s$ with the *relation* $r$. For example, "Dorothy went to her brother Richard's birthday party" should be represented by two ground-truth facts, `context(brother, Dorothy, Richard)`, and `context(sister, Richard, Dorothy)`. During inference time, since we are predicting a probabilistic distribution over all possible relations between Dorothy and Richard, we will obtain two sets of disjunctive facts, one of which is shown below:

```
0.01::ctx(son, Dorothy, Richard);
0.01::ctx(daughter, Dorothy, Richard);
...
0.90::ctx(brother, Dorothy, Richard);
...
0.01::ctx(motherinlaw, Dorothy, Richard);
0.01::ctx(none, Dorothy, Richard)
```

We model these facts as disjunctive facts because exactly one of them could be true. This helps the reasoning engine to rule out dubious reasoning chains. Assuming there are $m$ windowed contexts extracted from $c$ and maximum $n$ entities extracted from each windowed context, we generate in total $O(mn(n-1)(|R|+1))$ probabilistic facts that are then passed to our reasoning module.

Now the role of our reasoning module is to combine these base facts to deduce the answer for the expected

query. For the CLUTRR dataset, it suffices to define the higher-order predicate called `transitive`. Transitivity defines how two known relations can be connected to derive the third relation. As an example, the rule "father's mother is grandmother" is encoded as `transitive(father, mother, grandmother)` – this is equivalent to saying, if $B$ is $A$'s father, and $C$ is $B$'s mother, then $C$ is $A$'s grandmother. We manually crafted 92 transitivity triplets which are treated as an external knowledge base for this setup, and the full Scallop program for CLUTRR is shown in Figure 4.

In a more generalized setting, a single "transitive" definition would not suffice. Therefore, we add in a few other higher-order predicates as shown in Table 1. To account for negation, we create helper relations (e.g. "negchild") to denote the nonexistence of its positive counterpart (e.g. "child"), and use the predicate `negation` to connect them. With this set of rules, our program is expressive enough to encode all the logic and rules used in the dataset presented in RuleBERT (Saeed et al., 2021).

| Predicate | Example |
|---|---|
| symmetric | `symmetric(spouse)` |
| transpose | `transpose(husband, wife)` |
| implies | `implies(mother, parent)` |
| negation | `negation(child, negchild)` |

*Table 1.* Example higher-order predicates.

**Rule Learning.** For CLUTRR, specifying 92 transitive rules manually could be time-consuming. With a differentiable reasoning engine like Scallop, it is possible to do this in a smarter way. In CLUTRR, since there are 20 basic kinship relations, we have a finite space containing $20^3$ = 8K possible transitive facts. We treat all 8K transitivity facts probabilistic and to be learnt, and therefore we store them inside a tensor of size $20 \times 20 \times 20$. During training, we will allow gradients to be back-propagated to this transitivity tensor too so that these weights can be learnt simultaneously. In this case, the user does not specify any transitivity fact manually and everything is learnt on the fly. We can randomly initialize this tensor but with a range such as $[0, 0.1]$, since otherwise an insensible transitive fact may be getting a random high weight while it effectively does nothing for reasoning. The insensible transitive fact will obtain no gradient during training and the weight will be kept very high, making us unable to differentiate it from the other sensible transitive facts.

There are a few catches here. First, 8K transitive facts for CLUTRR is a lot that can slow down the training time. In real life, we use multinomial sampling to retrieve 150 transitive facts for training. During testing, we simply pick the top 150 facts from this. The sampling step shrinks the training time from 30 minutes per epoch to 10 minutes. These numbers are picked based on our knowledge of roughly 92 transitive facts that are essential to reasoning. When such knowledge is absent, one can set this number freely by treating it as another hyper-parameter. Secondly, during end-to-end training, it is possible to lose the semantic meaning of individual relations. That is, the relation we perceive as "sister" might not be used to represent "sister" internally, which will reduce the interpretability of the learnt model. There are ways to distill the true mapping or regularize the rules being learnt, but for the sake of this experiment, we keep it to the bare minimum and do not interpret the rules learnt. Lastly, the learning of rules and the fine-tuning of the underlying LM should happen separately with different learning rates – fine-tuning LM is an intricate process that requires a very small learning rate, while rules should be learnt with larger learning rates since gradients are directly back-propagated onto the weights. This can be realized by employing two separate optimizers, one for fine-tuning and the other for rule learning.

## 4. Experiments

We evaluate DSR-LM on the CLUTRR and DBpedia-INF datasets. We show that DSR-LM has accurate and generalizable long-range reasoning ability, and it overcomes the limitation of Scallop that hand-crafted rules must be provided through its rule learning feature. We refer readers to Appendix A for a full set of setups.

### 4.1. Experimental Setup

**Reasoner.** We use Scallop (Huang et al., 2021) to perform the differentiable probabilistic reasoning. To enable logic reasoning with negation, as needed in the DBpedia-INF dataset, we use the `topbottomkclauses` provenance with $k$ set to 3 throughout all the experiments.

**Pre-trained Language Models (PLMs) for Fine-tuning.** We used the HuggingFace pre-trained *w2v-google-news-300*, *RoBERTa*$_{\text{BASE}}$, and *DeBERTa*$_{\text{BASE}}$ as the pretrained language models. We finetune *RoBERTa*$_{\text{BASE}}$ and *DeBERTa*$_{\text{BASE}}$ during the training process with binary cross entropy loss. Note that although GPT-3 supports fine-tuning, it does not suit the end-to-end learning pipeline with Scallop, as the differentiable logical reasoning engine needs access to the underlying LM for back-propagating gradients.

### 4.2. Multi-hop Question Answering

**Baselines.** We compare DSR-LM with a spectrum of baselines from purely neural to logically structured. The baselines include pretrained large language models (BERT (Kenton & Toutanova, 2019) and RoBERTa (Liu et al., 2019)), non-LM counterparts (BiLSTM (Hochreiter & Schmidhu-

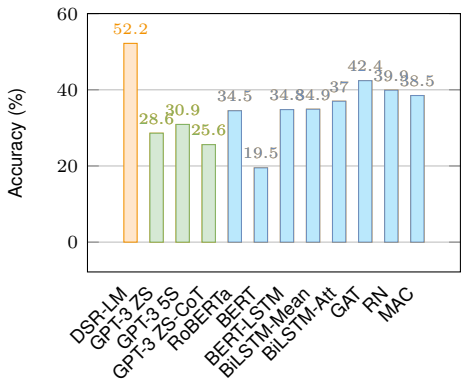


*Figure 2.* CLUTRR overall performance, including GPT-3 with ZS and ZS-CoT setup.

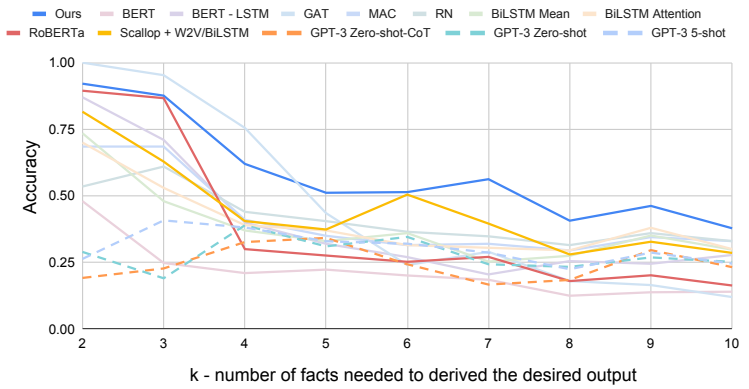


*Figure 3.* CLUTRR systematic generalization. Models except GPT-3* are trained on $k \in \{2, 3\}$. All models are tested on $k \in \{2, \ldots, 10\}$.

ber, 1997; Cho et al., 2014) and BERT-LSTM), structured models (GAT (Veličković et al., 2018), RN (Santoro et al., 2017), and MAC (Hudson & Manning, 2018)), and another neuro-symbolic model (RuleBert (Saeed et al., 2021)). The structured models include those models with relational inductive biases, while the neuro-symbolic model uses a logical reasoning engine for regularization.

**Accuracy and Generalizability.** We evaluate DSR-LM on the CLUTRR dataset and DBpedia-INF dataset, as reported in Figure 2 and Table 7.

In the CLUTRR experiment, DSR-LM achieves the best performance among all the models. A fine-grained generalizability study reveals that although all models' performance decline as the length of the test sequence increases, the pure neural-based models decrease the fastest, as shown in Figure 3. The experimental result aligns with previous observations that sequence models are not robust under compositional or systematic distribution shifts (Lake & Baroni, 2018). In the DBpedia-INF experiment, DSR-LM outperforms RuleBert on generalizability by 37% in terms of overall performance.

**Ablation Study.** Since DSR-LM has a model agnostic architecture, we study how the choice of different language models impacts the reasoning performance. As shown in Table 2, the two transformer-based models have on-par performance and outperform the word2vec one. However, note that the word2vec-based model still has better performance than all baselines other than GAT. Besides the higher final accuracy, the pre-trained transformer-based language model also accelerates the training process. Both DSR-LM-RoBERTa and DSR-LM-DeBERTa reach their best performance within 10 epochs, while the DSR-LM-w2v-BiLSTM model requires 40 epochs to be trained.

**GPT-3 Baselines.** We conduct experiments on the CLUTRR dataset with GPT-3 under the Zero-Shot (GPT-3 ZS) and Few(5)-Shot (GPT-3 5S) (Brown et al., 2020), as well as Zero-Shot-CoT (GPT-3 ZS-CoT) (Kojima et al., 2022a) settings.

| Model | Accuracy (%) |
|---|---|
| DSR-LM-w2v-BiLSTM | 40.39 ± 0.06 |
| DSR-LM-RoBERTa | **52.20** ± **4.07** |
| DSR-LM-DeBERTa | 51.57 ± 1.10 |

*Table 2.* CLUTRR Ablation study: We compare the performance of DSR-LM using different LMs.

| Model | Accuracy (%) |
|---|---|
| DSR-LM-RoBERTa | **52.20** ± **4.07** |
| DSR-LM-No-Rule | 51.48 ± 0.57 |

*Table 3.* CLUTRR Ablation study: We compare the performance of DSR-LM, one with hand-crafted rules and one with no hand-crafted rule but learnt all of them.

## 5. Concluding Remarks

In this work, we investigate how to improve LMs' logical reasoning capability using differentiable symbolic reasoning. Through extensive experiments, we demonstrate the effectiveness and utility of DSR-LM over challenging scenarios where widely deployed large LMs fail to reason reliably. We hope our work can lay the groundwork for exploring neuro-symbolic programming techniques to improve the robustness of LMs. One potential future work is using temporal logic to model dynamic semantics and *state tracking* in dialog systems – identifying the user's intent and parameters in each dialog act, and using them to drive the learning agent's actions consistently (Andreas et al., 2020).

## References


Andreas, J., Bufe, J., Burkett, D., Chen, C., Clausman, J., Crawford, J., Crim, K., DeLoach, J., Dorner, L., Eisner, J., et al. Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571, 2020.

Betz, G., Voigt, C., and Richardson, K. Critical thinking for language models. *arXiv preprint arXiv:2009.07185*, 2020.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *NeurIPS*, 2020.

Chaudhuri, S., Ellis, K., Polozov, O., Singh, R., Solar-Lezama, A., Yue, Y., et al. *Neurosymbolic Programming*. Now Publishers, 2021.

Cheney, J., Chiticariu, L., Tan, W.-C., et al. Provenance in databases: Why, how, and where. *Foundations and Trends® in Databases*, 1(4):379–474, 2009.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*, 2014.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Clark, P., Tafjord, O., and Richardson, K. Transformers as soft reasoners over language. In *IJCAI*, 2021.

Creswell, A., Shanahan, M., and Higgins, I. Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint arXiv:2205.09712*, 2022.

Elazar, Y., Kassner, N., Ravfogel, S., Ravichander, A., Hovy, E., Schütze, H., and Goldberg, Y. Measuring and improving consistency in pretrained language models. *TACL*, 2021.

Greff, K., Van Steenkiste, S., and Schmidhuber, J. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

Hase, P., Diab, M., Celikyilmaz, A., Li, X., Kozareva, Z., Stoyanov, V., Bansal, M., and Iyer, S. Do language models have beliefs? methods for detecting, updating, and visualizing model beliefs. *arXiv preprint arXiv:2111.13654*, 2021.

Helwe, C., Clavel, C., and Suchanek, F. M. Reasoning with transformer-based models: Deep learning, but shallow reasoning. In *AKBC*, 2021.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 1997.

Huang, J., Li, Z., Chen, B., Samel, K., Naik, M., Song, L., and Si, X. Scallop: From probabilistic deductive databases to scalable differentiable reasoning. *NeurIPS*, 2021.

Hudson, D. A. and Manning, C. D. Compositional attention networks for machine reasoning. In *ICLR*, 2018.

Kassner, N., Krojer, B., and Schütze, H. Are pretrained language models symbolic reasoners over knowledge? *arXiv preprint arXiv:2006.10413*, 2020.

Kenton, J. D. M.-W. C. and Toutanova, L. K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

Kim, Y. Sequence-to-sequence learning with latent neural grammars. *Advances in Neural Information Processing Systems*, 34:26302–26317, 2021.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022a.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners, 2022b.

Lake, B. and Baroni, M. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*, 2018.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Manhaeve, R., Dumancic, S., Kimmig, A., Demeester, T., and De Raedt, L. Deepproblog: Neural probabilistic logic programming. *NeurIPS*, 2018.

Mao, J., Gan, C., Kohli, P., Tenenbaum, J. B., and Wu, J. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *ICLR*, 2018.

Nye, M., Tessler, M., Tenenbaum, J., and Lake, B. M. Improving coherence and consistency in neural sequence models with dual-system, neuro-symbolic reasoning. *Advances in Neural Information Processing Systems*, 34, 2021.

Rocktäschel, T. and Riedel, S. End-to-end differentiable proving. *NeurIPS*, 2017.

Saeed, M., Ahmadi, N., Nakov, P., and Papotti, P. Rulebert: Teaching soft rules to pre-trained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1460–1476, 2021.

Santoro, A., Raposo, D., Barrett, D. G., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T. A simple neural network module for relational reasoning. *NeurIPS*, 2017.

Shaw, P., Chang, M.-W., Pasupat, P., and Toutanova, K. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 922–938, 2021.

Sinha, K., Sodhani, S., Dong, J., Pineau, J., and Hamilton, W. L. Clutrr: A diagnostic benchmark for inductive reasoning from text. *EMNLP*, 2019.

Soares, L. B., Fitzgerald, N., Ling, J., and Kwiatkowski, T. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2895–2905, 2019.

Talmor, A., Elazar, Y., Goldberg, Y., and Berant, J. olmpics-on what language model pre-training captures. *TACL*, 2020a.

Talmor, A., Tafjord, O., Clark, P., Goldberg, Y., and Berant, J. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. *NeurIPS*, 2020b.

Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., and Metzler, D. Long range arena: A benchmark for efficient transformers. In *International Conference on Learning Representations*, 2020.

Tenney, I., Das, D., and Pavlick, E. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, 2019.

Valmeekam, K., Olmo, A., Sreedharan, S., and Kambhampati, S. Large language models still can't plan (a benchmark for llms on planning and reasoning about change), 2022.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *ICLR*, 2018.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

Weidinger, L., Mellor, J., Rauh, M., Griffin, C., Uesato, J., Huang, P.-S., Cheng, M., Glaese, M., Balle, B., Kasirzadeh, A., et al. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*, 2021.

Xu, J., Zhang, Z., Friedman, T., Liang, Y., and Broeck, G. A semantic loss function for deep learning with symbolic knowledge. In *ICML*, 2018.

Zelikman, E., Wu, Y., and Goodman, N. D. Star: Bootstrapping reasoning with reasoning. *arXiv preprint arXiv:2203.14465*, 2022.

Zhang, H., Li, L. H., Meng, T., Chang, K.-W., and Broeck, G. V. d. On the paradox of learning to reason from data, 2022.

Zhang, Y., Chen, X., Yang, Y., Ramamurthy, A., Li, B., Qi, Y., and Song, L. Efficient probabilistic logic reasoning with graph neural networks. In *ICLR*, 2019.

Zhou, W., Hu, J., Zhang, H., Liang, X., Sun, M., Xiong, C., and Tang, J. Towards interpretable natural language understanding with explanations as latent variables. *NeurIPS*, 2020.

## A. Additional Experimental Setups

**CLUTRR.** The CLUTRR (Sinha et al., 2019) dataset consists of kinship reasoning questions. Given a context that describes a family's routine activity, the goal is to deduce two-family members' relationship that is not explicitly mentioned in the story. Although the dataset is synthetic, the sentences are crowd-sourced and hence there is a considerable amount of naturalness inside the dataset. The family kinship graph is synthetic and the names of the family members are randomized. Therefore, we have the following symbolic information at hand:

1. the full kinship graph corresponding to the story,
2. 92 hand-crafted rules for deriving relationships from known facts, and
3. a query representing the question.

We seek to use this symbolic information for a controlled study, but in a general setting where we compare with other baselines, none of this information is available. Additionally, we have the CLUTRR dataset being divided into different difficulties measured by $k$, the number of facts used in the reasoning chain. For training, we only have 10K data points with 5K $k = 2$ and another 5K $k = 3$, meaning that we can only receive supervision on data with short reasoning chains. The testing dataset, on the other hand, contains 1.1K data-points with $k \in \{2, \ldots, 10\}$.

**DBpedia-INF.** The DBpedia-INF dataset is a curated subset of the evaluation dataset used in RuleBert (Saeed et al., 2021). Similar to CLUTRR, it is generated synthetically with the purpose to test the reasoning capability of LMs. Given a synthetic passage describing the relation between entities, and soft rules between the entities, we target to deduce the relationship between any two entities. The symbolic program of DBpedia-INF consists of 26 predicates, 161 soft rules mined from DBpedia, and 16 rules defining the negation and symmetricity between the predicates. The difficulty of the questions is represented in terms of chaining depth from $D \in \{0, \ldots, 5\}$. The longer the chaining depth is, the harder the question. Compared to the exact dataset used in Rulebert, we clean it in order to ensure the question-answer pairs are logically consistent and probabilistically correct. Here is an example data-point from the original dataset that is logically inconsistent:

$c$ : Alice is not Bob's successor.
$q$ : Is Bob not Alice's successor?
$a$ : False

The logical fallacy is that the original dataset is generated assuming at least one of "successor(Alice, Bob)" and "successor(Bob, Alice)" is true. In reality, it might be the case that Alice and Bob are unrelated to each other and neither of these two facts need be true. Additionally, the sentences presented in this dataset appear to be simple and thus will be less noisy and less natural.

```
// Relation declaration
type context(rela: usize, subject: String, object: String)
type query(subject: String, object: String)
type transitive(r1: usize, r2: usize, r3: usize)

// Transitive facts that are specified manually; when learnt, we remove these facts
rel transitive = { /* ... 92 transitivity facts */ }

// Rules to derive the final answer
rel derive(r, s, o) = context(r, s, o)
rel derive(r3, x, z) = derive(r1, x, y), derive(r2, y, z), transitive(r1, r2, r3),
                       x != z
rel answer(r) = query(s, o), derive(r, s, o)
```

*Figure 4.* An illustrative example of a Scallop program used in the CLUTRR reasoning task.

**Hardware.** We perform all the experiments on a server with two 20-core Intel Xeon CPUs, four GeForce RTX 2080 Ti GPUs, and 768 GB RAM.

**GPT-3 prompts design.** For Zero-Shot, we use the prompt "So $B$ is $A$'s:" for the query pair $(A, B)$ to ask GPT-3 to complete the relationship between $A$ and $B$. We pick the phrase in the first line or before the first period from the completion text, and compare it directly with the ground truth relation. For the Few(5)-Shot setting, we randomly select 5

data-points from the training dataset used for other models ($k \in [2, 3]$) to serve as the examples. We use the same prompt for Few-Shot as the Zero-Shot. For the Zero-Shot-CoT setting, we use the prompt "Who is $B$ to $A$? Let's think step by step" to suggest GPT-3 to auto-complete while working out a reasoning chain. Under this setup, it is impossible to compare the answer to the ground truth automatically. Therefore, we manually check through the whole testing dataset of CLUTRR. Interestingly, ZS scores 28.6% accuracy on CLUTRR while ZS-CoT scores 25.6%, suggesting that the chain-of-thought prompting might not be helping the actual reasoning. In fact, there are many cases where GPT-3 favors complication over simplicity: GPT-3 frequently answers "stepdaughter", "stepmother", and "adopted son", while the real answers are simply "daughter", "mother", and "son". Additionally, GPT-3 could be deriving the correct result for the wrong reason, e.g. "Jeffrey is Gabrielle's son, which would make William her grandson, and Jeffrey's brother." While we count the final answer to be correct (William is Jeffrey's brother), there is a clear inconsistency in the reasoning chain: William cannot be Gabrielle's grandson and Jeffrey's brother simultaneously, given that Jeffrey is Gabrielle's son. Lastly, we observe that, unlike other methods that see an accuracy drop as $k$ becomes larger, ZS and ZS-CoT stay relatively consistent, suggesting that the size of context and the reasoning chain have a seemingly low impact on GPT-3's performance, as shown in Figure 3.

**Failure case analysis.** In Table 4, we showcase the failure cases of large LMs for logical inference, where Zero-shot-CoT denotes zero-shot chain-of-thoughts (Kojima et al., 2022b).

## B. Additional Experimental Results

### B.1. Failure case analysis.

We showcase in Appendix Table 4 that even state-of-the-art large LMs are prone to logical fallacies. On the other hand, the failure case of our method usually occurs in the stage of relation extraction. For example, for the following sentence "Christopher and Guillermina are having a father-daughter dance", our relation extractor fails to recognize the father-daughter relationship but rather thinks Christopher and Guillermina have a husband-wife relationship. We require most of the relation extraction to be correct in order to avoid a cascading error and derive the correct final result. As the error rate on individual relation extraction accumulates, it leads to the observed drop in accuracy as $k$ becomes larger.

### B.2. Rule Learning

DSR-LM can automatically learn weighted rules from data. In the following series of experiments, we seek to understand a) the quality of the learnt symbolic rules from a dataset, and b) the end-to-end question-answering performance without any given rule. We evaluate the rule learning performance of DSR-LM quantitatively and qualitatively on the CLUTRR dataset. Specifically, we target to learn the transitivity rule in kinship graphs, for example, the rule "father's mother is grandmother" encoded as `transitive(father, mother, grandmother)`.

**Setup.** There are two common usages of rule learning: a) mine high-level rules from a given set of knowledge base, and b) perform question and answering directly without hand-crafted rules. We thus set up the two experiments correspondingly:

*Rule Learning*: Given the ground truth kinship graphs of the context, the query and query results, learn a set of kinship transitivity rules.

*QA-No-Rule*: Given the natural language context, the question, and the answer, simultaneously learn to extract relations between entities and learn the transitive rules to combine relations and derive the desired answer.

**Training Setup.** Since the CLUTRR dataset consists of 20 different relations, and a transitivity relationship is defined over 3 relations, there are 8K possible transitivity facts over these relations. We attach a randomized confidence score from $[0, 0.1]$ to each possible transitivity rule, and the confidence scores are updated through back-propagation. Specifically, the learning process encourages the rules that yield the correct query result and suppresses the rules that lead to wrong answers. To avoid the exponential blow-up caused by injecting all the 8K rules in the reasoning engine, we sample 150 rules according to their weights during the training time and deterministically use of the top 150 learnt rules during the test time. For the *QA-No-Rule* setup, the confidence score of rules, the MLP classifier for relation extraction, and the underlying LM are learnt and updated simultaneously during training. To account for their difference, we employ two Adam optimizers $A_{\text{RL}}$ and $A_{\text{RE}}$. $A_{\text{RE}}$ is used for optimizing models for relation extraction, and thus will take as parameters the MLP classifier and the underlying LM. It has a low learning rate $0.00001$ since it needs to fine-tune LMs. $A_{\text{RL}}$, on the other hand, will take as a parameter the confidence score tensor for the transitive rules, and is set to have a higher learning rate of $0.01$.

**Results.** In the Rule Learning experiment, we find that the learnt logic rules align well with human understanding. We

| Confidence | Learnt Rules |
|---|---|
| 1.154 | mother(A,B) ← sister(A,C) ∧ mother(C,B) |
| 1.152 | daughter(A,B) ← daughter(A,C) ∧ sister(C,B) |
| 1.125 | sister(A,B) ← daughter(A,C) ∧ aunt(C,B) |
| 1.125 | father(A,B) ← brother(A,C) ∧ father(C,B) |
| 1.123 | granddaughter(A,B) ← grandson(A,C) ∧ sister(C,B) |
| 1.120 | brother(A,B) ← sister(A,C) ∧ brother(C,B) |
| 1.117 | brother(A,B) ← son(A,C) ∧ uncle(C,B) |
| 1.105 | brother(A,B) ← daughter(A,C) ∧ uncle(C,B) |
| 1.104 | daughter(A,B) ← wife(A,C) ∧ daughter(C,B) |
| 1.102 | mother(A,B) ← brother(A,C) ∧ mother(C,B) |
| ... | ... |

*Table 6.* The learned logic rules from CLUTRR dataset, showing only top 10 ranked by their respective confidence score.

| Test Depth | DSR-LM | RuleBert |
|---|---|---|
| all | **95.87** | 72.59 |
| 0 | **100.0** | 98.40 |
| 1 | **100.0** | 54.80 |
| 2 | **98.4** | 75.20 |
| 3 | **89.2** | 64.00 |
| 4 | **88.1** | 69.89 |
| 5 | **100.0** | 72.29 |

*Table 7.* DBpedia-INF generalization evaluation. Models are trained on 10K depth $D = 0$ training datapoints, and tested on datapoints of depth $D = [0, 5]$.

compare the top 92 most likely rules ranked by their confidence scores learnt by DSR-LM against the 92 hand-crafted rules, and 70 of them match. We demonstrate the top-10 rules in Table 6. As can be seen from the rules, they are quite sensible and interpretable.

In the QA-No-Rule experiment where the ground truth kinship graph is not provided, although the learnt rules do not reflect the common sense knowledge (Table 8), we still achieve on-par performance as if the knowledge base is given (Table 3). One possible explanation is that the neural networks learn a permutation of the entity mappings. For example, the causes for the learned rule "grandson(A,B) ← son(A,C) ∧ grandson(C,B)" could be that neural networks mistakenly interpret the ground truth relation "sister" as "son".

Through this qualitative analysis, it is clear that DSR-LM provides an interface to probe the intermediate steps for an end-to-end neural network, and in turn, enhance the model interpretability.

| $k$ | Prompt | GPT-3 CoT Output | Ground Truth |
|---|---|---|---|
| 2 | Jeffrey asked his brother William if he would come help him fix his car next weekend. Gabrielle was delighted to see her grandson, William, at her birthday party. Who is Jeffrey to Gabrielle? Let's do it step by step: | Jeffrey is Gabrielle's grandson. William is Jeffrey's brother. | grandson |
| 3 | Elliott felt lonely when his wife Marian was gone to see her mother Molly. Elliott and his daughter Karen took a day off school to go to the zoo. Who is Karen to Molly? Let's do it step by step: | Karen is Elliott's daughter. Molly is Elliott's wife's mother. Therefore, Karen is Molly's granddaughter. | granddaughter |
| 4 | Frank took his daughter Julia to the park. Beverly, Emilie´s sister went with them. Kimberley was excited because today she was going to the zoo with her uncle Frank. Jenny spent a great day shopping with her daughter, Kimberley. Who is Beverly to Jenny? Let's do it step by step: | Beverly is Jenny's sister-in-law. | niece |
| 5 | Melissa and her father, Stephen, went to the marina. Stephen's daughter, Margaret, had purchased a boat, and they were eager to see it. Annie and her aunt, Evelyn, went to the deli. They got a half a pound of corned beef, and two pounds of salami. Melissa and her sister Annie were arguing over a toy. While they were fighting, Melissa's sister Margaret grabbed the toy, and ran off to play with it by herself. Who is Evelyn to Melissa? Let's do it step by step: | Melissa's father is Stephen. Stephen has a daughter, Margaret. Margaret is Annie's aunt. Therefore, Evelyn is Annie's aunt, and Melissa's great-aunt. | aunt |
| 6 | James took his grandson Davis to the baseball game. Sharon and her son Brian went out to lunch together yesterday. Brian and his brother Oliver went to the park to play basketball. Davis played scrabble with his mother Sharon and his brother Oliver. Davis asked his brother Oliver if he would come help him fix his car next weekend. Who is James to Oliver? Let's do it step by step: | James is Davis's grandfather, so he is Oliver's great-grandfather. | grandfather |
| 7 | George rushed to the hospital to find out that his wife and already given birth to a boy and had named him David. Larry was playing checkers with his brother Craig. Larry's sister Jean was waiting to play. Judy made mixed drinks for her son Craig. Judy's daughter Jean was jealous because she was n't offered one. Jean planned a trip to the zoo for her brother, Craig. They had a great time. George was having trouble with his divorce because he only got to see his daughter Jean a few times a week. Who is David to Craig? Let's do it step by step: | George's wife gave birth to a boy and named him David. David is Craig's nephew. | brother |
| 8 | Anthony went to the bakery with his uncle Roger to pick up some bread for lunch. Anthony loved going to the store with his mom Mary. She always bought him snacks Charles would n't let his son Floyd go to the park by himself. Floyd's brother Anthony offered to go with him. Faye was hoping to travel out of state to visit her son Cecil. Ellie's brother, Charles, was doing his best to find the best flight. Mary and her son Anthony went out to lunch together yesterday. Cecil took her sister, Christine, out to dinner for her birthday. Who is Roger to Christine? Let's do it step by step: | Cecil is Faye's son, so Faye is Christine's mother-in-law. Christine is married to Faye's son, so Christine is Faye's daughter-in-law. Charles is Ellie's brother, so Charles is Christine's brother-in-law. Anthony is Charles's son, so Anthony is Christine's nephew. Roger is Anthony's uncle, so Roger is Christine's brother-in-law. | uncle |
| 9 | Gino bought gifts for his lovely sister, Jami. Michele and William left that evening to go bowling. Ethel and her son, Gordon, went to the park, and had a wonderful time. Manuel's father, Kenneth, thought his son was a monster. Meanwhile Michele thought her son Manuel was a little angel. Gino, Kenneth's other son, agreed with his dad. William asked his brother Larry if he would come help him fix his car next weekend. Joseph was so happy for his brother, Larry, getting a promotion. Gordon and his uncle Joseph went to the movies Sunday after church and got popcorn and candy while they were there. Who is Jami to Ethel? Let's do it step by step: | 1. Jami is Gino's sister. 2. Ethel is Gino's mother. 3. Therefore, Jami is Ethel's daughter-in-law. | niece |

*Table 4.* Qualitative analysis of GPT-3 Zero-Shot-CoT on the CLUTRR dataset. The novelty comes from the sentence marked in orange. Queries that are of interest are marked in blue. Correct answer in the output is marked green and incorrect ones are marked red.

| Confidence | Rule |
|---|---|
| 1.153661847 | mother(A,B) ← sister(A,C) ∧ mother(C,B) |
| 1.151766777 | daughter(A,B) ← daughter(A,C) ∧ sister(C,B) |
| 1.124880195 | sister(A,B) ← daughter(A,C) ∧ aunt(C,B) |
| 1.12450707 | father(A,B) ← brother(A,C) ∧ father(C,B) |
| 1.123031497 | granddaughter(A,B) ← grandson(A,C) ∧ sister(C,B) |
| 1.119753957 | brother(A,B) ← sister(A,C) ∧ brother(C,B) |
| 1.117262602 | brother(A,B) ← son(A,C) ∧ uncle(C,B) |
| 1.104628563 | brother(A,B) ← daughter(A,C) ∧ uncle(C,B) |
| 1.10367167 | daughter(A,B) ← wife(A,C) ∧ daughter(C,B) |
| 1.101553082 | mother(A,B) ← brother(A,C) ∧ mother(C,B) |
| 1.101539135 | brother(A,B) ← father(A,C) ∧ son(C,B) |
| 1.095460534 | sister(A,B) ← mother(A,C) ∧ daughter(C,B) |
| 1.071032643 | sister(A,B) ← father(A,C) ∧ daughter(C,B) |
| 1.070597649 | son(A,B) ← son(A,C) ∧ brother(C,B) |
| 1.069653988 | uncle(A,B) ← father(A,C) ∧ brother(C,B) |
| 1.066023946 | daughter(A,B) ← son(A,C) ∧ sister(C,B) |
| 1.060760975 | brother(A,B) ← brother(A,C) ∧ brother(C,B) |
| 1.056373119 | grandson(A,B) ← husband(A,C) ∧ grandson(C,B) |
| 1.054644465 | sister(A,B) ← son(A,C) ∧ aunt(C,B) |
| 1.05249536 | grandmother(A,B) ← sister(A,C) ∧ grandmother(C,B) |
| 1.049931884 | granddaughter(A,B) ← granddaughter(A,C) ∧ sister(C,B) |
| 1.049800634 | grandmother(A,B) ← brother(A,C) ∧ grandmother(C,B) |
| 1.047104836 | grandson(A,B) ← granddaughter(A,C) ∧ brother(C,B) |
| 1.045552254 | grandfather(A,B) ← mother(A,C) ∧ father(C,B) |
| 1.036108494 | son(A,B) ← daughter(A,C) ∧ brother(C,B) |
| 1.034998536 | sister(A,B) ← brother(A,C) ∧ sister(C,B) |
| 1.028737187 | grandmother(A,B) ← mother(A,C) ∧ mother(C,B) |
| 1.027377605 | grandfather(A,B) ← sister(A,C) ∧ grandfather(C,B) |
| 1.019370079 | brother(A,B) ← mother(A,C) ∧ son(C,B) |
| 1.01684773 | granddaughter(A,B) ← wife(A,C) ∧ granddaughter(C,B) |

*Table 5.* Showcase of the learned logic rules with top@30 confidence of DSR-LM rule learning.

| Confidence | Learnt Rules |
|---|---|
| 1.164 | grandson(A,B) ← son(A,C) ∧ grandson(C,B) |
| 1.116 | father(A,B) ← sister(A,C) ∧ daughter(C,B) |
| 1.099 | brother(A,B) ← niece(A,C) ∧ niece(C,B) |
| 1.092 | nephew(A,B) ← sister(A,C) ∧ sister(C,B) |
| 1.048 | brother(A,B) ← wife(A,C) ∧ daughter-in-law(C,B) |
| ... | ... |

*Table 8.* The top 5 logic rules learned under the *QA-No-Rule* setting, ranked by their confidence score.