Deep Inductive Logic Reasoning for Multi-Hop Reading Comprehension

Wenya Wang and **Sinno Jialin Pan**
Nanyang Technological University, Singapore
{wangwy, sinnopan}@ntu.edu.sg

Abstract

Multi-hop reading comprehension requires an ability to reason across multiple documents. On the one hand, deep learning approaches only implicitly encode query-related information into distributed embeddings which fail to uncover the discrete relational reasoning process to infer the correct answer. On the other hand, logic-based approaches provide interpretable rules to infer the target answer, but mostly work on structured data where entities and relations are well-defined. In this paper, we propose a deep-learning based inductive logic reasoning method that firstly extracts query-related (candidate-related) information, and then conducts logic reasoning among the filtered information by inducing feasible rules that entail the target relation. The reasoning process is accomplished via attentive memories with novel differentiable logic operators. To demonstrate the effectiveness of our model, we evaluate it on two reading comprehension datasets, namely WikiHop and MedHop.

1 Introduction

Reasoning has been extensively studied in the structured domain, e.g., knowledge base completion which infers missing facts given background entities and relations. However, when the background knowledge is expressed in natural languages, as shown in the multi-hop reading comprehension problem with triplet-form questions (Welbl et al., 2018), it becomes difficult to conduct complex reasoning because the entities and relations are not explicitly labeled in the documents. For example, consider the question "country(*Moonhole*, ?)", given the following documents:

> "*Moonhole* is a private community on the island of *Bequia*. *Moonhole* was founded by *Thomas* and *Gladys Johnston* in the 1960s."
>
> "*Gladys Johnston* was born in *United States*."
>
> "*Bequia* is an island and is part of the country of *Saint Vincent and the Grenadines*"

In this example, the underlined entities are used to infer the correct answer, i.e., "country(*Moonhole*, *Saint Vincent and the Grenadines*)", but are not explicitly annotated for relational reasoning.

Deep neural networks (DNNs) for multi-hop reading comprehension (RC) can be summarized into following three categories. 1) Memory-based models (Zhong et al., 2019; Wang et al., 2018; Zhuang and Wang, 2019) that produce query-aware context representations. 2) Graph-based approaches (Song et al., 2018; De Cao et al., 2019) that use graph neural networks to propagate information based on pre-constructed entity (context) graphs. 3) Neural Module networks (Andreas et al., 2016) that decompose the question into a series of action modules (Jiang and Bansal, 2019; Gupta et al., 2020). However, DNNs only implicitly encode relevant contexts and fail to explicitly uncover the underlying relational compositions for complex inference. For instance, in the above example, DNNs may encode *Bequia* and *Gladys Johnson* into 1-hop features, given the fact that both entities co-occur with the query *Moonhole*. As a result, the model may predict *United States* by linking it with *Gladys Johnson* instead of the correct answer *Saint Vincent and the Grenadines*. In contrast, human beings would easily produce the correct answer given the knowledge "if A is in B and B is part of country C, then A is in country C" and by examining the relations between each entity pair co-occurred in the context.

Inductive logic programming (ILP) (Muggleton, 1991) aligns with human reasoning by inducing interpretable rules to entail positive but not negative examples. To answer the previous query, ILP could generate a rule as located_in$(X, Z) \wedge$ country$(Z, Y) \Rightarrow$ country(X, Y). Combining deep learning with ILP is a promising direction to benefit from both worlds (Evans and Grefenstette, 2018; Dong et al., 2019). Deep logic models have been proposed for structured knowledge base comple-

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 4999 - 5009

tion (Minervini et al., 2017, 2020; Yang and Song, 2020; Rocktäschel and Riedel, 2017; Yang et al., 2017). However, it becomes much more challenging when dealing with natural language inputs, as in the case of multi-hop reading comprehension. Weber et al. (2019) proposed to combine a symbolic reasoner: prolog, with weak unifications based on distributed embeddings as a backward-chaining theorem prover to induce feasible rules for multi-hop reasoning. However, their work relies on the degree of precision for pre-extracted NERs and is limited by the number of rule templates.

To address the aforementioned limitations, we propose a novel end-to-end integration of deep learning and logic reasoning termed **D**eep **I**nductive **L**ogic **R**easoning (DILR). It consists of two components: 1) a hierarchical attentive reader that filters query-related and candidate-related information from given documents, and 2) a multi-hop reasoner that conducts inductive logic reasoning by attentively selecting proper predicates to form candidate rules and refines them upon given examples. We introduce novel differentiable logic operators combined with attention mechanisms for smooth back-propagation. Compared to existing deep logic models, we build connections between raw text inputs and the symbolic domain by mapping high-level semantic representations to logic predicates and instantiating logic variables with neural representations to conduct relational reasoning. We also parameterize the entire process for end-to-end differentiable learning.

The contributions of this work include: 1) We introduce a novel smooth connection between deep representation learning with logic reasoning by associating distributed representations with discrete logic predicates and their probabilistic evaluations. 2) We propose deep-learning-based inductive logic programming via attentive memories and differentiable logic operators for the task of multi-hop reading comprehension considering the number of reasoning steps. 3) We provide comprehensive evaluations of our model on two benchmark datasets.

2 Related Work

Multi-Hop Reading Comprehension Recent works for multi-hop RC include memory-based methods which apply attentions to iteratively update query and context representations considering their interactions (Dhingra et al., 2018; Clark and Gardner, 2018; Wang et al., 2018; Zhong et al., 2019; Zhuang and Wang, 2019; Jiang et al., 2019). To explicitly incorporate entity connections, De Cao et al. (2019), Ding et al. (2019), Qiu et al. (2019), Tang et al. (2020), Song et al. (2018) and Tu et al. (2019) proposed to build entity graphs and apply Graph Neural Networks for information propagation. Kundu et al. (2019) formalized reasoning as a path-finding problem with neural encoding to rank candidate paths. Path modeling was also adopted in (Chen et al., 2019) using pointer networks. However, these approaches only focus on local information without the ability to generalize, and some of them rely on off-the-shelf NER tools. Dhingra et al. (2020) proposed to convert texts into a virtual knowledge base for retrieval using a pre-constructed entity database. Another research direction is to decompose target questions into sub-questions (Min et al., 2019) or sub-modules parameterized with neural module networks (Jiang and Bansal, 2019; Gupta et al., 2020; Chen et al., 2020) which also fail to explicitly uncover the underlying logic for reasoning.

Deep Learning with Logic Reasoning Neuro-symbolic learning aims to integrate deep learning's ability on dealing with uncertainty and logic programming's ability on reasoning. Deep neural networks have been used to parameterize discrete logic operators and logic atoms (França et al., 2014; Hu et al., 2016; Manhaeve et al., 2018; Xu et al., 2018; Li and Srikumar, 2019; Wang and Pan, 2020; Wu et al., 2020) given the logic rules. A more challenging direction is inductive logic programming that automatically learns rules through representation learning and differentiable backpropagation (Evans and Grefenstette, 2018; Dong et al., 2019; Wang et al., 2019; Yang and Song, 2020).

Neuro-symbolic learning has been applied to knowledge-base completion through logic embeddings (Guo et al., 2016), tensor operations (Cohen, 2016; Rocktäschel and Riedel, 2017), adversarial learning (Minervini et al., 2017), variational learning (Qu and Tang, 2019) or attentions (Yang and Song, 2020). Differentiable theorem proving has also been proposed with weak unifications and backward chaining (Rocktäschel and Riedel, 2017; Campero et al., 2018; Minervini et al., 2020). However, unlike multi-hop RC, knowledge-base completion only takes structured inputs without the need to address language ambiguity. The most related work to ours is NLProlog (Weber et al., 2019), a neural theorem prover for multi-hop RC by con-

verting language utterances to distributed embeddings. However, NLProlog relies on a NER tool to extract entities and its expressiveness is limited by the number of rule templates.

3 Background

We focus on multi-hop reading comprehension tasks containing explicit query types which align with the standard ILP setting. Formally, for each RC problem, we are given a set of documents $C = \{c_1, ..., c_K\}$, a structured query in the form of a relational triplet $(s, q, ?)$, where s denotes the subject of the relation q, and a list of candidate answers $A = \{a_1, ..., a_n\}$. The task is to select an answer $a \in A$ such that $q(s, a)$ is satisfied, i.e., a is the object of relation q given the subject s. For example, country$(Moonhole, ?)$ is a query asking for the country where *Moonhole* is located. This task could be converted into an ILP problem with the formal definition as follows.

Definition 3.1 (Inductive Logic Programming)**.** Given a logic theory B representing the background knowledge (facts), a set of positive examples E^+ and a set of negative examples E^-, an ILP system aims to derive a hypothesis $\mathcal{H}$ which entails all the positive and none of the negative examples: $B \wedge \mathcal{H} \models \gamma$ for $\gamma \in E^+$. $B \wedge \mathcal{H} \not\models \gamma$ for $\gamma \in E^-$.

The hypothesis $\mathcal{H}$ is a logic program consisting of definite clauses $b_1 \wedge ... \wedge b_N \Rightarrow h$ where $b_1, ..., b_N$ and h are logic atoms. The LHS of "$\Rightarrow$" is the clause body and h is the head. An atom is composed of a predicate and its arguments, e.g., $h =$ located_in(X, Y) with predicate "located_in" and arguments X, Y. A ground atom is obtained by instantiating variables in the arguments with constants, e.g., $X =$ "US". We use $\mu(\cdot)$ to denote the value of an atom or a clause. For smooth optimization, we assign $\mu(\cdot) \in [0, 1]$ which indicates the probability of the atom or clause being true.

For multi-hop reading comprehension, we treat the query relation $q(X, Y)$ as the head atom of the clauses to be induced. The correct answer a_i^+ from each problem forms the set of positive examples $E^+ = \{q(s_i, a_i^+)\}_{i=1}^{N_+}$, and the incorrect answer a_i^- forms the set of negative examples $E^- = \{q(s_i, a_i^-)\}_{i=1}^{N_-}$. Here we use lower cases: s_i, a_i^+, a_i^- to represent constants and upper cases: X, Y to represent variables. The predicates in the logic domain correspond to pairwise relations between two entities[1]. To differentiate the number of inference steps, we define a l-hop reasoning clause as $F_0(X_0, X_1) \wedge ... \wedge F_l(X_l, X_{l+1}) \Rightarrow r(X_0, X_{l+1})$ with l denoting the number of extra arguments as bridging entities in the rule body except those in the head atom. Here r denotes a predicate, i.e., a relation between X_0 and X_{l+1}. Each subclause $F_t(X_t, X_{t+1})$ can be one or a conjunction ($\wedge$) of 2-ary atoms taking only X_t and X_{t+1} as arguments, e.g., $F_t(X_t, X_{t+1}) = r_1(X_t, X_{t+1}) \wedge r_2(X_t, X_{t+1})$.

4 Methodology

Overall, DILR simulates a multi-hop reasoning process considering different number of inference steps. It is an end-to-end framework consisting of two components: a *Hierarchical Attentive Reader* and a *Multi-hop Reasoner*. The attentive reader learns to select relevant information from the given documents to produce query-aware, candidate-aware and bridging entity representations. These representations are passed to the multi-hop reasoner to instantiate logic atoms in order to generate and evaluate clauses that are relevant to the query relation. The multi-hop reasoner conducts rule induction via attentive memories that softly select atoms to form new clauses and novel differentiable logic operators that produce probabilistic values for generated clauses. The final loss can be back-propagated smoothly to update the attentive reader for more accurate selections. Next, we illustrate each component with more details.

4.1 Hierarchical Attentive Reader

To avoid inevitable errors brought by off-the-shelf NER tools for named entity extraction, we propose to extract relevant information using an attentive reader. Since multiple documents (contexts) are involved for each question, we design a 2-level hierarchical attention network to progressively filter token-level and context-level information. Specifically, the token-level attentions aim to select l-hop $(l = 0, ..., L)$ relevant entities in each context. Then the context-level attentions produce the final representations by softly attending to each context considering different number of reasoning hops.

4.1.1 Token-Level Attention

Given a query subject s with n_s tokens, a candidate a with n_a tokens, and a context c of length

[1] We restrict each atom as a 2-ary atom that takes exactly 2 arguments, analogical to relations in the knowledge base.

n_c, we denote by $\mathbf{S} \in \mathbb{R}^{n_s \times D}$, $\mathbf{A} \in \mathbb{R}^{n_a \times D}$ and $\mathbf{C} \in \mathbb{R}^{n_c \times D}$ their word features after a biGRU layer, respectively. For multi-hop reasoning, we use different attentions for finding or relocating target tokens in each context, inspired by (Gupta et al., 2020). Firstly, a subject-to-context attention is adopted to find similar tokens as the subject in each context: $\mathbf{B}^s_{ij} = \mathbf{w}_s^\top [\mathbf{S}_i; \mathbf{C}_j; \mathbf{S}_i \circ \mathbf{C}_j]$ where $\mathbf{w}_s$ is a learnable transformation vector and $[;]$ denotes concatenations. We obtain the normalized similarity score α^s_{ij} between the i-th token in the subject and the j-th token in the context via a softmax operation on each row of $\mathbf{B}^s$. A subject-aware (0-hop) context representation is produced as

$$\mathbf{h}_s = \sum_{j=1}^{n_c} \bar{\alpha}^s_j \mathbf{C}_j, \text{ with } \bar{\alpha}^s_j = \sum_{i=1}^{n_s} \alpha^s_{ij} \beta^s_i, \quad (1)$$

where β^s_i weighs the contribution of each subject token via a self-attention: $\boldsymbol{\beta}^s = \text{softmax}(\bar{\mathbf{w}}_s^\top \mathbf{S} + b_s)$. Similarly, we produce an attention score α^a_{ij} for the j-th context token w.r.t. the i-th candidate token and a candidate-aware context representation $\mathbf{h}_a$. We denote by $\mathbf{s} = \boldsymbol{\beta}^s \mathbf{S}$, and $\mathbf{a} = \boldsymbol{\beta}^a \mathbf{A}$ the feature representation of the query subject and the candidate entity, respectively.

For $(l+1)$-hop reasoning ($l \geq 0$), it is desired to relocate to intermediate (bridging) entities that are related to the l-hop entities. Hence, we adopt context-to-context attentions $\mathbf{B}^{l+1}_{ij} = \mathbf{w}_l^\top [\mathbf{C}_i + \mathbf{h}^l; \mathbf{C}_j; (\mathbf{C}_i + \mathbf{h}^l) \circ \mathbf{C}_j]$ given the l-hop representation $\mathbf{h}^l$ where $\mathbf{h}^0 = \mathbf{h}_s$. We use α^{l+1}_{ij} to denote a normalized attention score between the i-th and the j-th context tokens after applying a softmax operator over each row of $\mathbf{B}^{l+1}$. With $\bar{\alpha}^0_j = \bar{\alpha}^s_j$, the $(l+1)$-hop bridging context representation becomes

$$\mathbf{h}^{l+1} = \sum_{j=1}^{n_c} \bar{\alpha}^{l+1}_j \mathbf{C}_j, \text{ with } \bar{\alpha}^{l+1}_j = \sum_{i=1}^{n_c} \alpha^{l+1}_{ij} \bar{\alpha}^l_i. \quad (2)$$

4.1.2 Context-Level Attention

With multiple contexts (documents) available, we use a context-level attention to produce the final l-hop feature representations. When $l = 0$, the model softly attends to each context to produce context-attended subject representation as

$$\widehat{\mathbf{h}}_s = \sum_{k=1}^{K} \bar{\gamma}^s_k \mathbf{h}_{s,k}, \quad (3)$$

where $\bar{\gamma}^s_k$ is the attention weight of context c_k obtained by normalizing over a score vector $\boldsymbol{\gamma}^s$ with entries $\gamma^s_k = \mathbf{v}_s^\top [\mathbf{s}; \mathbf{h}_{s,k}; \mathbf{s} \circ \mathbf{h}_{s,k}]$. Here $\mathbf{h}_{s,k}$ is the subject-aware context representation computed in (1) corresponding to the k-th context c_k. $\mathbf{v}_s$ is a trainable transformation vector. The final subject representation is produced as $\bar{\mathbf{h}}_s = \mathbf{W}_s [\mathbf{s}; \widehat{\mathbf{h}}_s; \mathbf{s} \circ \widehat{\mathbf{h}}_s]$ incorporating both original features and attended information. Similar procedure applies to each candidate entity to produce $\bar{\mathbf{h}}_a$. We treat $\bar{\mathbf{h}}_s$ and $\bar{\mathbf{h}}_a$ as 0-hop subject and candidate representations, respectively.

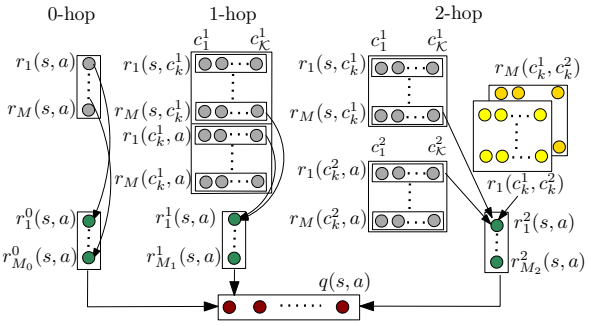

Figure 1: An example of multi-hop ILP. The existential predicates $r_1, ..., r_M$ are used to define invented predicates $r^l_1, ..., r^l_{M_l}$ through attentions. The invented predicates will produce the final clauses to define q.

When $l > 0$, the context-level attention aims to produce the probability of each context being chosen as a bridging entity using

$$p^l_k = \sigma(\mathbf{v}_l^\top [\bar{\mathbf{h}}_s; \mathbf{h}^l_k; \bar{\mathbf{h}}_s \circ \mathbf{h}^l_k]), \quad (4)$$

where $\sigma(\cdot)$ is the sigmoid function, and $\mathbf{h}^l_k$ is the l-hop intermediate entity representation for context $c_k \in C$ computed using (2).

4.2 Multi-Hop Reasoner

The multi-hop reasoner aims to conduct complex reasoning by first generating probable logic clauses and then evaluating each clause by instantiating the variables with relevant contexts obtained from the attentive reader. The clause generation process is parameterized by attentive memories which compute the probability of selecting each atom to form a relevant clause to entail the query relation. An illustration of the procedure is shown in Figure 1 and is elaborated in the following sub-section. The clause evaluation process is then to instantiate variables in each atom with constants such as query subjects, candidate entities or bridging entities. The outputs from the attentive reader, i.e., $\bar{\mathbf{h}}_s$, $\bar{\mathbf{h}}_a$ and $\{\mathbf{h}^l_k\}$'s ($l > 0$), can be used as feature representations for these constants to compute the atom scores for clause evaluation and updates.

4.2.1 Clause Generation

A definite clause is composed of atoms defined over relational predicates. Since there are no explicit relations given in this task, we pre-define a fixed set of relations for each corpus, named as existential predicates: $P_E = \{r_1, ..., r_M\}$, e.g., "located_in", "next_to". For expressiveness, we further create a set of invented predicates $P_I = \cup_{l=0}^{L} P_I^l$ defined from the existential predicates, inspired by (Evans and Grefenstette, 2018). Specifically, $P_I^l = \{r_1^l, ..., r_{M_l}^l\}$ consists of invented predicates r_m^l defined using l-hop reasoning clauses $F_0(X_0, X_1) \wedge ... \wedge F_l(X_l, X_{l+1}) \Rightarrow r_m^l(X_0, X_{l+1})$. For example, located_in$(X_0, X_1) \wedge$ next_to$(X_1, X_2) \Rightarrow$ outside(X_0, X_2) defines a 1-hop invented predicate "outside". Here L is the maximum hop number. The final clauses defining the query relation will be produced by learning to select relevant invented predicates, e.g., $r_i^{l_1}(X, Y) \wedge ... \wedge r_j^{l_n}(X, Y) \Rightarrow q(X, Y)$ with $0 \leq l_1 \leq ... \leq l_n \leq L$. The number of actual inference steps l_n to answer q is flexibly decided by the model itself, which will be discussed later.

The clause generation process is divided into two stages: 1) to generate clauses defining invented predicates using only the existential predicates, and 2) to generate final clauses defining query relation q using only the invented predicates. To allow for smooth optimization, we parameterize both stages by computing an attention weight for each predicate indicating its probability to appear in the clause body. Specifically, we assign each predicate a learnable embedding to indicate its semantics. Let $\mathbf{U} \in \mathbb{R}^{D \times M}$ denote the embedding matrix for M existential predicates and $\mathbf{U}^l \in \mathbb{R}^{D \times M_l}$ ($l \in \{0, 1, ..., L\}$) denote the embedding matrix for l-hop invented predicates. In the first stage, we use attentive memories to generate

$$\mathbf{S}_t^l = \text{sparsemax}((\mathbf{W}_t^l \mathbf{U}_t^l)^\top (\mathbf{W}_b^l \mathbf{U})), \quad (5)$$

$$\mathbf{U}_{t+1}^l = \mathbf{U}_t^l + \mathbf{S}_t^l \cdot (\mathbf{W}_v^l \mathbf{U}), \quad (6)$$

where $\mathbf{U}_0^l = \mathbf{U}^l$, and $\mathbf{W}_t^l$ and $\mathbf{W}_b^l$ are transformation matrices for invented predicates (queries) and existential predicates (keys), respectively. We use *sparsemax*, a sparse version of softmax (Martins and Astudillo, 2016), to select only a small number of predicates. Intuitively, to learn to define a l-hop invented predicate r_m^l, (5) and (6) sequentially produce $F_t(X_t, X_{t+1})$ at each step $t \in \{0, ..., l\}$ to form the clause body by attending over all the existential predicates with attention weight $\mathbf{S}_t^l$. For example, when $l = 1$, (5) first attends to the existential predicate r_i to generate $F_0(X_0, X_1) = r_i(X_0, X_1)$ at step $t = 0$, and then attends to another predicate r_j to generate $F_1(X_1, X_2) = r_j(X_1, X_2)$ at step $t = 1$. The resulting clause $r_i(X_0, X_1) \wedge r_j(X_1, X_2) \Rightarrow r_m^1(X_0, X_2)$ defines the invented predicate r_m^1.

In the second stage, we produce H final clauses taking invented predicates to define the target atom $q(X, Y)$. Given an embedding $\mathbf{u}_q \in \mathbb{R}^D$ for the target relation q, we use a multi-head attention mechanism to compute a probability distribution $\mathbf{s}_h$ over all the invented predicates for each head $h \in \{1, ..., H\}$ to produce the h-th final clause:

$$\mathbf{s}_h = \text{sparsemax}\{(\mathbf{W}_q^h \mathbf{u}_q)^\top (\mathbf{W}_e^h [\mathbf{U}^0; ...; \mathbf{U}^L])\}, \quad (7)$$

where $\mathbf{s}_h$ is a sparse selective distribution over $P_I = \{r_1^0, ..., r_{M_L}^L\}$. For example, if $\mathbf{s}_h$ selects r_1^0 and r_2^1, the final clause becomes $r_1^0(X, Y) \wedge r_2^1(X, Y) \Rightarrow q(X, Y)$, which involves at most 1 inference step because r_2^1 is a 1-hop invented predicate. This completes the recursive rule generation step with multi-hop inference. To this end, we generate H clauses that can be used to define $q(X, Y)$.

4.2.2 Clause Evaluation

Instantiation The clauses generated using the attentive memories need to be tested and refined against the given positive and negative examples, known as learning from entailment that tries to maximize the truth probabilities of positive examples and minimize those of negative examples. The positive examples correspond to $q(s, a)$ and the negative examples correspond to $\{q(s, a^-)\}$'s, where s, a and a^- refers to the query subject, correct answer and incorrect candidate, respectively. To obtain the truth probabilities of these atoms, we first instantiate the variables for each generated clause with constant contexts, e.g., $X = s$ and $Y = a$ (or $Y = a^-$) in $q(X, Y)$. The bridging variables $X_1, ..., X_l$ are instantiated using the bridging contexts selected via the attentive reader as introduced in 4.1. Specifically, to instantiate each X_l, we pick top-$\mathcal{K}$ contexts (documents) $\{c_1^l, ..., c_{\mathcal{K}}^l\} \subseteq C$, namely $X_l = c_k^l, 1 \leq k \leq \mathcal{K}$ with highest probabilities according to p_k^l computed via (4).

Neural Logic Operator Given a definite clause $b_1 \wedge ... \wedge b_N \Rightarrow h$ consisting of grounded atoms (e.g., $b_1 = r_1(s, a)$), we could obtain the value for its head atom as $\mu(h) = \mu(b_1 \wedge ... \wedge b_N)$. To compute the RHS involving logic operators ($\wedge$, $\vee$), T-norm (Klement et al., 2013) is usually adopted:

$T : [0,1] \times [0,1] \rightarrow [0,1]$. For example, minimum t-norm defines $T_\wedge(\mu_1, \mu_2) = \min(\mu_1, \mu_2)$, $T_\vee(\mu_1, \mu_2) = \max(\mu_1, \mu_2)$. Product t-norm defines $T_\wedge(\mu_1, \mu_2) = \mu_1 \cdot \mu_2$, $T_\vee(\mu_1, \mu_2) = 1 - (1 - \mu_1) \cdot (1 - \mu_2)$. Here $\mu_1, \mu_2 \in [0,1]$ refer to the value for the body atoms. However, minimum t-norm is prone to learning plateau because the gradient only flows through one of the inputs. Product t-norm is less stable and is prone to exponential decay when the number of atoms in the clause grows.

To address these limitations, we propose a novel neural logic operator $\mathcal{G}$ defined as follows:

$$\mathcal{G}_\vee(\mu_1, ..., \mu_N) = 1 - \exp\left(\sum_{n=1}^{N} \log(1 - \mu_n + \epsilon)\right)^{\frac{1}{N}},$$
$$\mathcal{G}_\wedge(\mu_1, ..., \mu_N) = \exp\left(\sum_{n=1}^{N} \log(\mu_n + \epsilon)\right)^{\frac{1}{N}}, \quad (8)$$

where $\mu_1, ..., \mu_N \in [0,1]$ refer to the probabilistic values of all the atoms in the conjunctive ($\wedge$) or disjunctive ($\vee$) clause. ϵ is a small value to guarantee the validity for logarithm. The operator $\mathcal{G}$ has the following property that is ideal for logic semantics.

Proposition 1. *When* $\forall \mu_n \rightarrow 1$ *with* $1 \leq n \leq N$, $\mathcal{G}_\wedge(\mu_1, ..., \mu_N) \rightarrow 1$, *aligning with logic "AND". When* $\exists \mu_n \rightarrow 1$, $\mathcal{G}_\vee(\mu_1, ..., \mu_N) \rightarrow 1$, *aligning with logic "OR".*

Proposition 2. $0 \leq \mathcal{G}_\wedge(\mu_1, ..., \mu_N) - \mu_{\min} \leq (N^{1/(1-N)} - N^{N/(1-N)})(\prod_{n \neq \min} \mu_N)^{1/(N-1)}$, *where* min *refers to the index of the minimum value among* $\{\mu_1, ..., \mu_N\}$.

Proof.

$$\begin{aligned}
&\mathcal{G}_\wedge(\mu_1, ..., \mu_N) - \mu_{\min} \\
&= \exp\left(\sum_{n=1}^{N} \log(\mu_n + \epsilon)\right)^{1/N} - \mu_{\min} \\
&\approx \prod_{n=1}^{N} \mu_n^{1/N} - \mu_{\min}. \quad (9)
\end{aligned}$$

Without loss of generality, assume the minimum value is $\mu_{\min} = \mu_1$. By fixing $\mu_2, ..., \mu_N$ as constants, we obtain the gradient for (9) w.r.t. μ_1 as

$$\frac{1}{N} \mu_1^{(1-N)/N} \prod_{n=2}^{N} \mu_n^{1/N} - 1. \quad (10)$$

(9) obtains its maximum value when (10) equals 0, resulting in

$$\mu_1 = N^{N/(1-N)} \prod_{n=2}^{N} \mu_n^{1/(N-1)}. \quad (11)$$

By replacing $\mu_{\min}$ in (9) with (11), we have

$$\begin{aligned}
&\mathcal{G}_\wedge(\mu_1, ..., \mu_N) - \mu_{\min} \\
&\leq N^{1/(1-N)} \prod_{n=2}^{N} \mu_n^{1/N(N-1)+1/N} \\
&- N^{N/(1-N)} \prod_{n=2}^{N} \mu_n^{1/(N-1)} \\
&= (N^{1/(1-N)} - N^{N/(1-N)}) \prod_{n=2}^{N} \mu_n^{1/(N-1)}.
\end{aligned}$$

This completes the proof. □

In other words, the difference between $\mathcal{G}_\wedge$ and $\mu_{\min}$ is bounded. When $N = 2$, the RHS of the inequality equals to $1/4 \cdot \mu_{n \neq \min}$, which makes $\mathcal{G}_\wedge$ closer to $\mu_{\min}$ when $\mu_{n \neq \min}$ is smaller. This formulation results in a more stable and smooth gradient flow compared to minimum t-norm. Moreover, It avoids exponential decay in the output when $N > 1$. It also facilitates neural learning when the exact clause is parameterized with attention scores.

Evaluation With the neural logic operator defined above, the value for the head atom can be inferred once the value for each body atom is given. For grounded atoms over existential predicates, e.g., $r_m(s, a)$, we directly generate its value using a relational network $\mathcal{F} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^M$ that takes the features of two constant arguments as input to produce a probability distribution over all the existential predicates $r_1, ..., r_M$:

$$\begin{aligned}
&\mathcal{F}(\bar{\mathbf{h}}_s, \bar{\mathbf{h}}_a) \\
&= \text{softmax}(\mathbf{W}_r \tanh[\bar{\mathbf{h}}_s; \bar{\mathbf{h}}_a; \bar{\mathbf{h}}_s - \bar{\mathbf{h}}_a; \bar{\mathbf{h}}_s \circ \bar{\mathbf{h}}_a]),
\end{aligned}$$

where $\mu(r_m(s, a))$ equals the m-th entry of $\mathcal{F}(\bar{\mathbf{h}}_s, \bar{\mathbf{h}}_a)$, and $\bar{\mathbf{h}}_s$ and $\bar{\mathbf{h}}_a$ are the outputs from the attentive reader. Similarly, $\mathbf{h}_k^l$ can be regarded as the feature of c_k^l generated from the attentive reader which is used to compute atom values with bridging entities, e.g., $\mu(r_m(s, c_k^l))$.

For l-hop grounded atoms over invented predicates $\{r_1^l(s, a), ..., r_{M_l}^l(s, a)\}$, we compute their values according to the value of the clause body that defines them, e.g., $\mu(F_0(s, c_k^1) \wedge ... \wedge F_l(c_k^l, a))$ using neural logic operators:

$$\boldsymbol{\mu}^l = \max_{z \in \mathcal{Z}_l} \exp\left(\sum_{t=0}^{l} \mathbf{S}_t^l \log(\boldsymbol{\mu}_{(z_t, z_{t+1})} + \boldsymbol{\epsilon})\right)^{\frac{1}{(l+1)}} \quad (12)$$

Here $\boldsymbol{\mu}^l = [\mu(r_1^l(s,a)), ..., \mu(r_{M_l}^l(s,a))]^\top$ denotes the vector of the atom values formed by those l-hop invented predicates. We denote by $\mathcal{Z}_l = \{(s, c_k^1, ..., c_k^l, a)\}_{1 \le k \le \mathcal{K}}$ the set for all possible instantiations for l-hop reasoning and denote by z_t the t-th constant of $z \in \mathcal{Z}_l$. $\boldsymbol{\mu}_{(z_t, z_{t+1})} = [\mu(r_1(z_t, z_{t+1})), ..., \mu(r_M(z_t, z_{t+1}))]^\top$ is a vector of values for grounded atoms over existential predicates. Thus, $\exp(\cdot)$ gives a neural approximation of logic conjunctions as shown in (8) over $\{F_t(z_t, z_{t+1})\}_{0 \le t \le l}$, each of which is a sparse selection of existential predicates using $\mathbf{S}_t^l$. We use a $\max$ operator to generate the maximum score over all possible instantiations in $\mathcal{Z}_l$ to represent the final truth probability of each invented predicate. Intuitively, a relation between two entities should be satisfied as long as there is at least one instantiation that follows the rule. Also note that (12) has the effect that when $\mathbf{S}_t^l[i,j] \approx 0$, the corresponding predicate r_j will have little effect on the value of its head r_i^l, which is in contrast to existing T-norms.

The final value for $q(s,a)$ is computed via

$$\mu(q(s,a)) = \max_{1 \le i \le H} \left\{ \exp\left(\mathbf{s}_i \log([\boldsymbol{\mu}^0; ...; \boldsymbol{\mu}^L] + \boldsymbol{\epsilon}) \right) \right\},$$

which selects the maximum score over H final clauses that define $q(s,a)$. We use the cross-entropy loss over $\mu(q(s,a))$ as the final objective to train the entire model (except the word embeddings which are kept fixed) in an end-to-end manner. Here we organize the dataset according to subject-candidate pairs: (s,a). We associate the ground-truth label $y = 1$ with (s,a) if a is the correct answer, otherwise, $y = 0$.

5 Experiment

We conduct experiments on two multi-hop reading comprehension datasets, namely WikiHop and MedHop (Welbl et al., 2018). The WikiHop dataset contains 43,738 training and 5,129 development instances ranging over 277 query relations. MedHop is a medical dataset containing 1,620 training and 342 development instances with a unique query relation, i.e., "*interact with*". For WikiHop, we experiment with both non-contextual (follow (Weber et al., 2019)) named as DILR and contextual word embeddings (BERT (Devlin et al., 2019)) named as DILR-BERT to demonstrate our model's generalization ability. For MedHop, we use the same setting following (Weber et al., 2019). We define $M = 10$ relations as existential predicates and $M_l = 5$ invented predicates for each hop with $l = 0, 1, 2$. The number of final clauses to define the query relation is $H = 5$ and the number of candidate bridging contexts for each hop is set to $\mathcal{K} = 5$. The dimension of predicate embeddings and biGRU layer is 100 and 200, respectively. For training, we adopt Adam optimization with learning rate initialized at 0.001. The batch size is set to 10. For all the experiments, we use the development dataset to evaluate the results because the test data is not publicly available to be grouped by query relations under our setting.

Model	WikiHop				MedHop
	Publisher	*Developer*	*Country*	*Record_label*	
EPAr	81.48	65.52	70.10	80.57	64.90
HDEG	85.19	**79.31**	73.20	83.39	-
DynSAN	85.19	75.86	76.80	83.39	-
NLProlog	83.33	68.97	77.84	79.51	65.78
DrMD	85.19	75.86	74.23	81.27	67.25
DILR	**88.89**	**79.31**	**79.38**	**84.10**	**71.35**
BERT	**88.89**	79.31	81.96	83.04	-
DILR-BERT	**88.89**	**82.76**	**84.02**	**84.45**	-

Table 1: Comparison results with baseline models on MedHop and four selected relations in WikiHop.

Data	D1	D2	D1+D2	D3	D1+D2+D3
# training data	<1000	1000 ∼ 4000	<4000	>4000	0 ∼ 5000
# query relations	38	7	45	2	47
EPAr	64.79	61.47	63.12	68.70	64.43
HDEG	69.49	63.75	66.61	70.71	67.57
DynSAN	68.61	60.95	64.76	68.70	65.69
NLProlog	63.20	56.51	59.84	59.64	59.79
DrMD	69.25	62.23	65.73	68.89	66.47
DILR	**71.37**	**64.39**	**67.87**	**73.66**	**69.23**
BERT	73.96	66.14	70.04	74.24	71.02
DILR-BERT	**74.42**	**67.83**	**71.27**	**77.29**	**72.68**

Table 2: Comparisons on WikiHop in terms of the number of training instances for each query relation.

5.1 Experimental Result

Weber et al. (2019) only selects four different query relations from WikiHop, namely *Publisher*, *Developer*, *Country* and *Record_label*. For fair comparisons, we first follow their setting to evaluate on these four domains, then we further evaluate on all the other 47 valid query relations[2] in WikiHop for a more complete analysis. For pure deep learning baselines, we use two memory-based models: EPAr (Jiang et al., 2019) and DynSAN (Zhuang and Wang, 2019), and a graph-based model HDEG (Tu et al., 2019)[3]. For reasoning baselines, we use NLProlog (Weber et al., 2019) and DrMD which is a differentiable reasoning model adapted from (Dhin-

[2]We consider those query relations containing at least 20 development and 100 training instances as valid.

[3]These baselines demonstrate good performances according to the leaderboard with available code for implementation.

Setting	Located	Publisher	Producer	Country	Occupation	Record
$\leq$ 0 Hop	70.24	85.19	61.76	76.29	71.78	78.80
$\leq$ 1 Hop	71.72	87.04	**67.65**	77.84	71.78	81.98
$\leq$ 2 Hop	**73.38**	**88.89**	**67.65**	**79.38**	**73.62**	**84.10**
$\leq$ 3 Hop	70.79	83.33	58.82	76.29	69.63	82.69
DILR	**73.38**	**88.89**	**67.65**	**79.38**	**73.62**	**84.10**
− PI	69.03	**88.89**	58.82	77.32	69.63	81.63
− ILP	65.80	85.19	52.94	75.77	68.71	80.92
− AR	66.73	85.19	55.88	74.23	65.95	83.39
− Rel	63.22	87.04	64.71	78.35	69.63	79.15
prod-T	70.98	**88.89**	58.82	77.84	69.33	81.63
min-T	69.32	87.04	64.71	78.87	70.25	80.92
NLO	**73.38**	**88.89**	**67.65**	**79.38**	**73.62**	**84.10**

Table 3: Abalation study over model architecture. − indicates the removal of an element from the model.

gra et al., 2020) by removing pre-defined entities and only consider mention interactions following our settings. BERT is a baseline model that first generates a context-aware query subject (or candidate) representation given the input "[CLS] query subject (or candidate) [SEP] context [SEP]". Then the representations of the query subject and each candidate are concatenated to be fed into a binary classifier. On the other hand, DILR-BERT adopts the attentive reader and the multi-hop reasoner on top of BERT representations. For all the baselines, we train the models on each query relation separately to test the reasoning capability, same as our setting.

Table 1 lists the results for MedHop and four query relations from WikiHop according to (Weber et al., 2019). Clearly, DILR gives the best performances across all the baselines, demonstrating the advantage of combining deep attentive learning with logic reasoning. Though NLProlog also conducts logic reasoning, it is limited by the model's capacity and relies on the extraction accuracy of the NER tool. Even with well-trained contextualized word embeddings (DILR-BERT), our model still brings consistent performance gains.

For a more thorough analysis, we take the entire WikiHop dataset and group the query relations in terms of the number of training instances. As shown in Table 2, there are 38 relations (D1) containing less than 1,000 training examples, 7 relations (D2) with training examples ranging from 1,000 to 4,000 and 2 relations (D3) having more than 4,000 training examples. The entire data (D1+D2+D3) contains 36,653 training instances and 4,462 development instances. We report the micro-average accuracy scores over all the domains within each data group and their combinations in Table 2. Clearly, our model achieves the best performances over all data groups. The margin is larger for D1 and D3, demonstrating the consistency of our proposed model with varying data sizes. In fact, ILP could be beneficial when training data is not sufficient via learning of generalized rules.

5.2 Analysis

To provide detailed analysis, we conduct ablation experiments on 6 datasets as shown in Table 3. For fair demonstration, we pick one relation in D3 (Located), 2 relations in D2 (Occupation and Record) and 3 relations in D1 (Publisher, Producer, Country). The first four rows reflect the accuracies by varying the maximum allowed number of reasoning hops (L). Clearly, ≤ 0 Hop and ≤ 3 Hop produce lower accuracies because ≤ 0 Hop fails to model the bridging entities and ≤ 3 Hop could overfit the model given most of the questions only involve at most 2 reasoning hops.

The middle part of Table 3 reflects the effect of each element of DILR. Specifically, −PI removes the invented predicates: remove (5), (6), (12) and replace $\mathbf{U}^l$ with $\mathbf{U}$ in (7). −ILP removes the reasoner and uses a classifier on top of the attentive reader to produce the final predictions. −AR removes the attentive reader and uses NER tools to extract entities for reasoning. −Rel only computes binary relations that decide whether two constants are related or not. This demonstrates the effect of relational reasoning considering different relations. By comparison, it is evident that removing any component will suffer from non-trivial prediction loss, especially for ILP. To verify the effect of the neural logic operator (NLO), we compare it with two T-norm operators, namely "prod-T" for product T-norm and "min-T" for minimum T-norm. Clearly, NLO produces the best performances across all the experiments.

To provide a concrete view of how the attentive reader filters relevant information and how the generated clauses look like, we list three examples as shown in Table 4. The underlined texts have the maximum attention weights learned from the attentive reader. The bold texts indicate the query subject and the correct answer for each query. Clearly, the attentive reader is able to select bridging entities relevant to the answer. The third column lists some learned clauses from the reasoner. The first row of each example shows the clauses that define an invented predicate and the second row shows the final clause that entails the query relation[4]. We use ab-

[4]We convert the attention scores to the exact clauses by

Query	Attentive Reader	Generated Clause
country(CC, ?)	1. **Christ Church** is a historic church building at ..., Massachusetts. 2. Massachusetts is the most populous state in ... **United States**.	$r_7(CC, M) \wedge r_2(M, US) \Rightarrow r_3^1(CC, US)$ $r_3^1(CC, US) \Rightarrow \text{country}(CC, US)$
record_label(MM, ?)	1. **Method Man** is the B-side ... titled Enter the Wu-Tang. 2. Enter the Wu-Tang is the debut ... on **Loud Records**	$r_3(MM, WT) \wedge r_4(WT, LR) \Rightarrow r_2^1(MM, LR)$ $r_2^1(MM, LR) \Rightarrow \text{record\_label}(MM, LR)$
genre(FO, ?)	1. **Face Off** ! is an hockey game ... 2. Hockey is a family of **sports** ...	$r_7(FO, H) \wedge r_6(H, S) \Rightarrow r_5^1(FO, S)$ $r_5^1(FO, S) \Rightarrow \text{genre}(FO, S)$

Table 4: Examples of learned attentions and clauses.

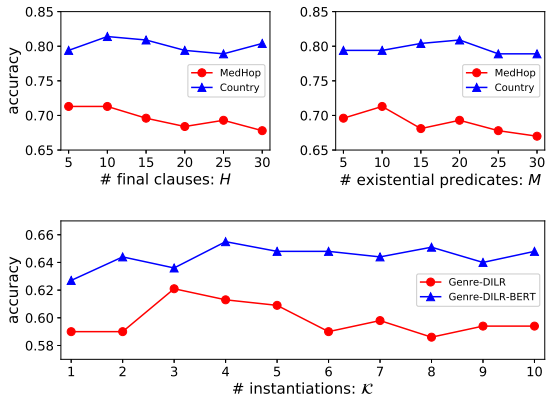

Figure 2: Sensitivity test over H, M and $\mathcal{K}$.

breviated entities as the constants in each grounded atom (e.g., "CC" is short for "Chris Church"). The two clauses for the first example could be read as: if *Chris Church* and *Massachusetts* has relation r_7, and *Massachusetts* and *United States* has relation r_2, then the country of *Chris Church* is *United States*.

We further demonstrate the robustness of DILR by varying model parameters, as shown in Figure 2. The top subplots reveal the accuracies on MedHop and Country datasets when changing the number of final clauses H (left) and the number of existential predicates M (right). The subplot in the bottom depicts the accuracies when varying the number of instantiations $\mathcal{K}$ of the bridging contexts for Genre dataset under both DILR and BERT-DILR models. We shall observe that the performances are relatively stable given that the total number of testing examples are less than 400 for each dataset.

6 Conclusion

We propose an end-to-end model DILR to solve the problem of multi-hop reading comprehension. DILR smoothly connects a hierarchical attentive reader with a multi-hop reasoner to conduct automatic information extraction and complex reasoning. We also introduce differentiable logic operators to induce valid clauses with smooth and stable gradient-based learning. Extensive experiments reveal consistent improvements brought by DILR.

keeping those predicates with scores higher than 0.4.

Acknowledgement

This work is supported by Microsoft Research Asia collaborative research grant and the College of Engineering International Postdoctoral Fellowship of Nanyang Technological University.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 39–48.

Andres Campero, Aldo Pareja, Tim Klinger, Josh Tenenbaum, and Sebastian Riedel. 2018. Logical rule induction and theory learning using neural theorem proving. *CoRR*, abs / 1809.02193.

Jifan Chen, Shih-Ting Lin, and Greg Durrett. 2019. Multi-hop question answering via reasoning chains. *CoRR*, abs/1910.02610.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V. Le. 2020. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855.

William W. Cohen. 2016. Tensorlog: A differentiable deductive database. *CoRR*, abs/1605.06523.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2306–2317.

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

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2018. Neural models for reasoning over multiple mentions using coreference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 42–48.

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W. Cohen. 2020. Differentiable reasoning over a virtual knowledge base. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2694–2703.

Honghua Dong, Jiayuan Mao, Tian Lin, Chong Wang, Lihong Li, and Denny Zhou. 2019. Neural logic machines. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Richard Evans and Edward Grefenstette. 2018. Learning explanatory rules from noisy data. *J. Artif. Intelligent Res.*, 61:1–64.

Manoel V. França, Gerson Zaverucha, and Artur S. D'avila Garcez. 2014. Fast relational learning using bottom clause propositionalization with artificial neural networks. *Mach. Learn.*, 94(1):81–104.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2016. Jointly embedding knowledge graphs and logical rules. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 192–202.

Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner. 2020. Neural module networks for reasoning over text. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2410–2420.

Yichen Jiang and Mohit Bansal. 2019. Self-assembling modular networks for interpretable multi-hop reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4474–4484.

Yichen Jiang, Nitish Joshi, Yen-Chun Chen, and Mohit Bansal. 2019. Explore, propose, and assemble: An interpretable model for multi-hop reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2714–2725.

Erich Peter Klement, Radko Mesiar, and Endre Pap. 2013. Triangular norms. *Springer Science and Business Media*.

Souvik Kundu, Tushar Khot, Ashish Sabharwal, and Peter Clark. 2019. Exploiting explicit paths for multi-hop reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2737–2747.

Tao Li and Vivek Srikumar. 2019. Augmenting neural networks with first-order logic. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 292–302.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. 2018. Deepproblog: Neural probabilistic logic programming. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 3753–3763.

André F. T. Martins and Ramón Fernandez Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1614–1623.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop reading comprehension through question decomposition and rescoring. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6097–6109.

Pasquale Minervini, Matko Bosnjak, Tim Rocktäschel, Sebastian Riedel, and Edward Grefenstette. 2020. Differentiable reasoning on large knowledge bases and natural language. In *AAAI*, pages 5182–5190.

Pasquale Minervini, Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. 2017. Adversarial sets for regularised neural link predictors. In *UAI*.

Stephen Muggleton. 1991. Inductive logic programming. *New Generation Computing*, 8:295–318.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6140–6150.

Meng Qu and Jian Tang. 2019. Probabilistic logic neural networks for reasoning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7710–7720.

Tim Rocktäschel and Sebastian Riedel. 2017. End-to-end differentiable proving. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3788–3800.

Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. 2018. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. *CoRR*, abs/1809.02040.

Zeyun Tang, Yongliang Shen, Xinyin Ma, Wei Xu, Jiale Yu, and Weiming Lu. 2020. Multi-hop reading comprehension across documents with path-based graph convolutional network. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3905–3911.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2704–2713.

Po-Wei Wang, Priya L. Donti, Bryan Wilder, and J. Zico Kolter. 2019. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6545–6554.

Wenya Wang and Sinno Jialin Pan. 2020. Integrating deep learning with logic fusion for information extraction. In *AAAI*, pages 9225–9232.

Yizhong Wang, Kai Liu, Jing Liu, Wei He, Yajuan Lyu, Hua Wu, Sujian Li, and Haifeng Wang. 2018. Multi-passage machine reading comprehension with cross-passage answer verification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1918–1927.

Leon Weber, Pasquale Minervini, Jannes Münchmeyer, Ulf Leser, and Tim Rocktäschel. 2019. NLProlog: Reasoning with weak unification for question answering in natural language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6151–6161.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Meixi Wu, Wenya Wang, and Sinno Jialin Pan. 2020. Deep Weighted MaxSAT for Aspect-based Opinion Extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5618–5628.

Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. 2018. A semantic loss function for deep learning with symbolic knowledge. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5498–5507.

Fan Yang, Zhilin Yang, and William W. Cohen. 2017. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2319–2328.

Yuan Yang and Le Song. 2020. Learn to explain efficiently via neural logic inductive learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Victor Zhong, Caiming Xiong, Nitish Shirish Keskar, and Richard Socher. 2019. Coarse-grain fine-grain coattention network for multi-evidence question answering. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Yimeng Zhuang and Huadong Wang. 2019. Token-level dynamic self-attention network for multi-passage reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2252–2262.