The Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI-22)

Weakly Supervised Neural Symbolic Learning for Cognitive Tasks

Jidong Tian[1, 2], **Yitian Li**[1, 2], **Wenqing Chen**[1, 2], **Liqiang Xiao**[1, 2],
Hao He [*1, 2], **Yaohui Jin**[1, 2]

[1] MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
[2] State Key Lab of Advanced Optical Communication System and Network,
School of Electronic Information and Electrical Engineering,
Shanghai Jiao Tong University
{frank92, yitian_li, wenqingchen,xiaoliqiang, hehao, jinyh}@sjtu.edu.cn

Abstract

Despite the recent success of end-to-end deep neural networks, there are growing concerns about their lack of logical reasoning abilities, especially on cognitive tasks with perception and reasoning processes. A solution is the neural symbolic learning (NeSyL) method that can effectively utilize pre-defined logic rules to constrain the neural architecture making it perform better on cognitive tasks. However, it is challenging to apply NeSyL to these cognitive tasks because of the lack of supervision, the non-differentiable manner of the symbolic system, and the difficulty to probabilistically constrain the neural network. In this paper, we propose WS-NeSyL, a **W**eakly **S**upervised **Ne**ural **Sy**mbolic **L**earning model for cognitive tasks with logical reasoning. First, WS-NeSyL employs a novel back search algorithm to sample the possible reasoning process through logic rules. This sampled process can supervise the neural network as the pseudo label. Based on this algorithm, we can backpropagate gradients to the neural network of WS-NeSyL in a weakly supervised manner. Second, we introduce a probabilistic logic regularization into WS-NeSyL to help the neural network learn probabilistic logic. To evaluate WS-NeSyL, we have conducted experiments on three cognitive datasets, including temporal reasoning, handwritten formula recognition, and relational reasoning datasets. Experimental results show that WS-NeSyL not only outperforms the end-to-end neural model but also beats the state-of-the-art neural symbolic learning models.

Context: **Kristin** and her **son** **Justin** went to visit her **mother** **Carol** on a nice Sunday afternoon. They went out for a movie together and had a good time.
Question: How is **Carol** related to **Justin**?
Answer: Grandmother.

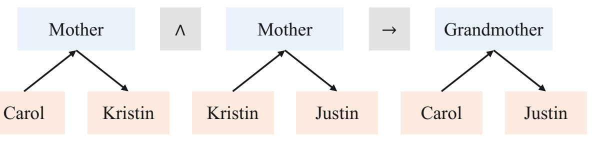


Figure 1: An example of a cognitive problem. To solve the problem, we are required to perceive information (colorful words) from the texts on the top of the figure and then reason out the result based on the bottom logic.

Introduction

Like human cognition, cognitive tasks include two processes: information perception and logical reasoning. As the example of a cognitive instance in Figure 1, to answer the question, we should first perceive subjects and kinships from the text and then make logical reasoning according to the logical rule of $\mathrm{Mother}(x, y) \wedge \mathrm{Mother}(y, z) \rightarrow \mathrm{Grandmother}(x, z)$. Although end-to-end deep neural networks (DNNs) have achieved great success recently (Devlin et al. 2019), they cannot complete these cognitive tasks in accordance with effective cognitive processes (Ribeiro, Guestrin, and Singh 2019; Sen and Saffari 2020; Sugawara et al. 2020). This is because DNNs can hardly master systematic knowledge without the supervision of the reasoning process and apply symbolic rules to make logical reasoning. In response to the problem, an expected solution is neural symbolic learning (NeSyL), which introduces a symbolic system to guide the neural network by cascading them. As a result, NeSyL can combine the extraordinary perception ability of the neural network and the stable logical reasoning ability of the symbolic system (Dai et al. 2019).

Previous studies of NeSyL (Si et al. 2019; Weber et al. 2019; Chen et al. 2020) focused more on solving the interaction between the neural network and the symbolic system. These studies solved the problem of gradient barriers to a certain extent. However, there are still three challenges that hinder the application of NeSyL to different cognitive tasks. Firstly, most cognitive tasks involve complex reasoning processes, which lack the ground truth to supervise (Weber et al. 2019; Chen et al. 2020). Secondly, due to the non-differentiable nature of the symbolic system, available NeSyL methods usually use non-generalizable strategies to integrate neural networks and symbolic systems (Weber et al. 2019; Chen et al. 2020), which hinders the reuse of these methods on cognitive tasks with different logic rules. Thirdly, even if the two modules can interact with each other, the deterministic constraints of the logic rules cannot constrain the probabilistic reasoning process of the neural net-

[*]Corresponding author.

work effectively (Manhaeve et al. 2018).

In this paper, we propose a **W**eakly **S**upervised **Ne**ural **Sy**mbolic **L**earning model (WS-NeSyL) to solve above problems on cognitive tasks. Aiming at the first two challenges, we propose a weakly supervised framework to train the neural network without the supervision of the reasoning process or the non-generalizable explicit symbolic system. Specifically, inspired by Sinha et al. (2019), WS-NeSyL includes a back search algorithm to sample "the most likely" reasoning process as the pseudo label to train the neural network in a weakly supervised manner. Based on such a back search algorithm, WS-NeSyL does not need to label the reasoning process for each instance and can back-propagate gradients to the neural network without an explicit symbolic system. To meet the last challenge, we introduce a probabilistic logic regularization into WS-NeSyL to further constrain the probabilistic reasoning process of the neural network based on the probabilistic logic theory of DeepProbLog (Manhaeve et al. 2018). Different from DeepProbLog, WS-NeSyL imposes local logic constraints instead of the global loss. Therefore, it can benefit the model in learning probabilistic logic and reducing the negative impact of noisy pseudo labels.

We verify our methods on three cognitive tasks: the temporal reasoning task of multi-hop TimeBank-Dense (Cassidy et al. 2014) (MHTBD), the handwritten formula recognition task of multi-hop HWF (Sinha et al. 2019) (MH-HWF), and the relational reasoning task of CLUTRR (Sinha et al. 2019). Results show that WS-NeSyL outperforms the baseline end-to-end neural network and neural symbolic learning models on all three datasets. In particular, WS-NeSyL can prevent overfitting through the weakly supervised framework and the regularization, and has good abilities of perception and reasoning on different cognitive tasks.

Overall, our contributions are as follows:

- We propose a novel weakly supervised framework, WS-NeSyL, for cognitive tasks with logical reasoning. WS-NeSyL employs a back search algorithm that can sample the reasoning process without the ground truth, which benefits the neural network in learning logic rules.
- We introduce a probabilistic logic regularization to the proposed method, WS-NeSyL, to match the probabilistic characteristic of the neural network, which benefits the further integration of the neural network and the rule system based on the pseudo label.
- We have experimented on three cognitive datasets: temporal reasoning, relational reasoning, and handwritten formula recognition datasets. Results show that the proposed WS-NeSyL achieves the best performance on all three datasets compared with previous end-to-end neural model and other NeSyL methods.

Related Work

Cognitive Task

Generally, cognitive tasks contain two processes of perception and reasoning to benchmark the cognitive ability of models. Theorem proving (Alvandi and Watt 2019; Zhelezniakov, Zaytsev, and Radyvonenko 2021) and handwritten formula recognition (Dai et al. 2019; Sinha et al. 2019) are typical cognitive tasks that require perceiving mathematical symbols from pictures and calculate results based on mathematical theorems. CLEVR (Johnson et al. 2017; Sampat et al. 2021) also provides a benchmark that introduces logical reasoning to visual question answering. Sinha et al. (2019) define CLUTRR benchmarks with robustness and generalization evaluations to measure models' perception and reasoning abilities, respectively. Besides, LogiQA (Liu et al. 2020) and ReCLor (Yu et al. 2020) provide more complex reasoning scenarios by considering commonsense and domain knowledge. In order to complete these cognitive tasks, information perception and logic/knowledge-based reasoning should be solved simultaneously.

Neural Symbolic Learning

Many early studies focus on implicit neural symbolic integration methods that map inputs and rules to the same vector space for cognitive tasks. Most of these studies adopt logic theories, such as fuzzy logic (Zadeh 1965; Hájek 1998), to revise neural architectures (Donadello, Serafini, and d'Avila Garcez 2017; Marra et al. 2020; Wang et al. 2020), enhance representations (Sourek et al. 2018; Li and Srikumar 2019; Dong et al. 2019; Qu and Tang 2019; Dumancic et al. 2019), and optimize loss functions (Xu et al. 2018). However, these studies cannot make full-featured logical reasoning and usually lack generalization and interpretability (Dai et al. 2019). Recently, explicit NeSyL methods cascade the neural network and the symbolic system to guide the neural network to understand the inference logic (Evans and Grefenstette 2018; Si et al. 2019; Yang, Yang, and Cohen 2017; Weber et al. 2019; Chen et al. 2020; Dai et al. 2019; Tsamoura, Hospedales, and Michael 2021). On this basis, Manhaeve et al. (2018) further propose a probabilistic theory to describe the neural reasoning process. However, these solutions require well-constructed symbolic reasoners and additional annotations, which can only be applied to limited cognitive tasks. Inspired by Li et al. (2020), we propose a weakly supervised neural symbolic learning method, WS-NeSyL, without explicit reasoners and additional annotations. WS-NeSyL can also achieve competitive performance on different cognitive tasks.

Problem Setting

We can simplify a cognitive task to a problem of outputting a hypothesis reasoning function f that maps the raw input x and the pre-defined limited set of logic rules $R = \{r_1, r_2, \cdots, r_l\}$ to the output y ($f(x, R) = y$). In fact, each logic rule r can be written as a form of conjunctive implication: $r : b_1 \wedge b_2 \wedge \cdots \wedge b_m \rightarrow h_1 \wedge h_2 \wedge \cdots \wedge h_n$, where b and h mean the literal and the atom respectively. Actually, R is a set that contains all possible rules not just deterministic rules. As a result, other logic forms, such as disjunction $\vee$ and equivalence $\equiv$ should also be included in R. For example, if we know $b_1 \vee b_2 \rightarrow h_1 \vee h_2$, we should add four rules to R: $b_1 \rightarrow h_1$, $b_1 \rightarrow h_2$, $b_2 \rightarrow h_1$, and $b_2 \rightarrow h_2$.

Based on the above description, the objective of the cognitive task is to find a target function f_θ that satisfies Eq. 1,

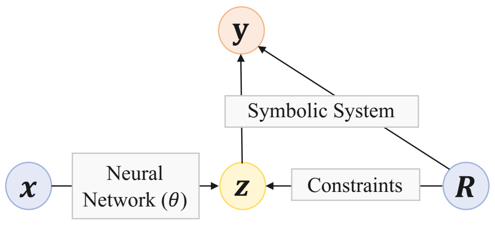


Figure 2: Variable dependency of NeSyL. The intermediate variable z is used to describe the reasoning process and is determined by the neural network (θ represents trainable parameters) based on the raw input x. Meanwhile, rules R also constrain the form of z. The output y is inferred by the symbolic system based on R and z.

where D is the set of observations and f_θ is usually formulated by a neural network.

$$\forall(x,y) \in D(f_\theta(x,R)=y) \tag{1}$$

From Eq. 1, it is hard to describe how x and R interact to infer y in detail. Therefore, NeSyL includes the reasoning process through an intermediate variable z represented by symbols. Variable dependency of NeSyL is shown in Figure 2. Based on the relations among x, R, z, and y, the objective of NeSyL is to acquire the neural network f_θ that satisfies Eq. 2, where $\vdash$ and $\vDash$ imply syntactic consequence and semantic consequence. Most previous studies (Weber et al. 2019; Manhaeve et al. 2018) adopt the ground truth or a specific symbolic reasoner to supervise the intermediate z, but our proposal method weakly supervises z based on R.

$$\forall(x,y) \in D(z=f_\theta(x), R \vdash z, R \cup z \vDash y) \tag{2}$$

Methodology

In this section, we first describe the neural architecture of the WS-NeSyL. Next, we introduce two essential mechanisms in WS-NeSyL: weakly supervised learning with a back search algorithm and probabilistic logic regularization.

Neural Perception and Reasoning

As Figure 2, the neural network is used to map the input x to a normalized probability distribution of the intermediate variable z. Most previous studies adopt the reasoning proofs as z so that the neural network is only regarded as a perception module (Li et al. 2020; Gontier et al. 2020). We regard z as the whole reasoning process described by the sequence of rules. In a word, we adopt the neural network as both the perception module and the reasoning module. Therefore, although the specific network architecture depends on the task, neural perception and reasoning can be described by a unified encoder-decoders framework with one encoder and two decoders, as shown in Figure 3. This framework can process multi-hop logical reasoning such as $b_1 \wedge b_2 \rightarrow h_1 \wedge b_3 \rightarrow h_2$. Note that we assume that the perception process of literal is independent, but we will add the available information as inputs to prevent repeated perception in practice.

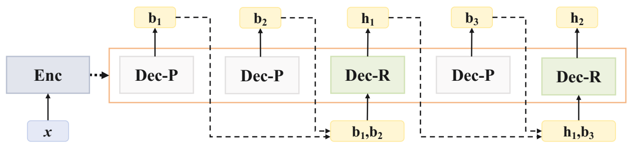


Figure 3: Neural perception and reasoning framework of encoder-decoders. For the example of multi-hop reasoning $b_1 \wedge b_2 \rightarrow h_1 \wedge b_3 \rightarrow h_2$, Enc is the encoder that encodes the raw input (texts or graphs). Dec-P is the perception decoder to perceive information (b_1, b_2, and b_3) from x, while Dec-R is the reasoning decoder to reason out the conclusion (h_1 and h_2) based on the perceived information.

Weakly Supervised Learning

In practice, it is challenging to obtain the ground truth of the intermediate reasoning process z, so we cannot train the model in a supervised manner. Alternatively, we adopt a weakly supervised method to train the neural model.

Based on the definition of the cognitive task, the final objective is to maximize the probability $p_\theta(y|x,R)$, where θ represents parameters to be optimized. When introducing the intermediate variable z, we can rewrite the probability in Eq. 3 by marginalizing over z according to the dependency in Figure 2, where $p_\theta(z|x,R)$ is calculated by the neural network, and $p(y|z,R)$ is generated by the symbolic system without parameters.

$$\begin{aligned} p_\theta(y|x,R) &= \sum_z p_\theta(y,z|x,R) \\ &= \sum_z p(y|z,R)p_\theta(z|x,R) \quad (x \perp y|z) \end{aligned} \tag{3}$$

Based on the maximum likelihood estimation, the derivation of the log-likelihood (L) can be calculated by Eq. 4.

$$\begin{aligned} \nabla_\theta L = \nabla_\theta \log p_\theta(y|x,R) &= \frac{\nabla_\theta p_\theta(y|x,R)}{p_\theta(y|x,R)} \\ &= \sum_z \frac{p(y|z,R)p_\theta(z|x,R)}{\sum_{z'} p(y|z',R)p_\theta(z'|x,R)} \nabla_\theta \log p_\theta(z|x,R) \\ &= \sum_z q(z|x,y,R)\nabla_\theta \log p_\theta(z|x,R) \\ &= \mathbb{E}_{z\sim q(z|x,y,R)}[\nabla_\theta \log p_\theta(z|x,R)] \end{aligned} \tag{4}$$

From Eq. 4, the key point to compute the derivation is to calculate the posterior distribution $z \sim q(z|x,y,R)$. According to the Eq. 2, $q(z|x,y,R)$ is non-zero only when z satisfies $R \vdash z$ and $R \cup z \vDash y$. Therefore, $q(z|x,y,R)$ can be written as Eq. 5 shows, where $Z = \{z|R \vdash z, R \cup z \vDash y\}$.

$$q(z|x,y,R) = \begin{cases} 0 & z \notin Z \\ \dfrac{p_\theta(z|x)}{\sum_{z' \in Z} p_\theta(z'|x)}, & z \in Z \end{cases} \tag{5}$$

Although the expression of the distribution can be given directly, it is complicated to calculate $q(z|x,y,R)$ due to the summation $\sum_{z\in Z} p_\theta(z|x)$. An alternative way is to sample z^* from the posterior distribution $q(z|x,y,R)$

Algorithm 1: n-Step Metropolis-Hastings Sampler.

```
Input: Probability $p_\theta(\cdot|x)$; $Q(\cdot|z)$; Step $n$; $Z$.
Output: Sampled $z^*$
1  Initialize $z^{(0)} = \arg\max_z(p_\theta(z|x))$;
2  if $z^{(0)} \notin Z$ then
3      for $t = 1$ to $n$ do
4          One-Step Sample $z \sim Q(\cdot|z^{(t)})$;
5          if $t == 1$ then
6              $z^{(t+1)} = z$
7          else
8              Sample $u \sim \mathcal{U}(0,1)$;
9              $\alpha = \min\{\frac{p_\theta(z|x)Q(z^{(t)}|z)}{p_\theta(z^{(t)}|x)Q(z|z^{(t)})}, 1\}$;
10             if $u \le \alpha$ and $z \neq \varnothing$ then
11                 Accept: $z^{(t+1)} = z$
12             else
13                 Reject: $z^{(t+1)} = z^{(t)}$
14             end
15         end
16     end
17 else
18     return $z^{(0)}$
19 end
20 return $z^* = z^{(n)}$
```

through Metropolis-Hastings (M-H) sampling (Metropolis et al. 1953; Hastings 1970). Sample z^* is used as the pseudo label to train the neural network. We will introduce the n-step M-H sampler and the 1-step sampler, respectively.

Metropolis-Hastings Sampler. To sample z^* from $q(z|x, y, R)$, we regard $\pi(z) = \frac{p(z|x)}{\sum_{z' \in Z} p(z'|x)}, z \in Z$ as the desired stationary distribution, and define a proposal distribution Q. According to the Metropolis-Hastings algorithm, when given the current state z_i and the next sampled state z_j, we need to determine whether to accept z_j based on a specific acceptance ratio α. The acceptance ratio α can be calculated by π and Q, shown in Eq. 6. In the equation, the summation term happens to be offset, so the calculation is simplified. We can find that if z_i does not satisfy logical constraints, we always accept the new sample z_j because $p(z_i|x) = 0$. In practice, we set the initial α as 1 ($t = 1$) when the neural network cannot generate a possible $z \in Z$. Furthermore, to ensure the convergence of the Markov chain, the sampling process will iterate n times, the whole algorithm of which is shown in Alg. 1. The only remaining problem is to design Q and perform one-step sampling $z \sim Q(\cdot|z^{(t)})$.

$$\alpha(z_i, z_j) = \min\{\frac{\pi(z_j)Q(z_i|z_j)}{\pi(z_i)Q(z_j|z_i)}, 1\} = \begin{cases} 1 & z_i \notin Z \\ \min\{\frac{p(z_j|x)Q(z_i|z_j)}{p(z_i|x)Q(z_j|z_i)}, 1\}, & z_i \in Z \end{cases} \quad (6)$$

1-Step Sampler. We propose a random back search algorithm (one-step sampler) to construct Q and sample $\hat{z}$, which

Algorithm 2: 1-Step Sampler.

```
Input: Initial $z$; Ground Truth $y$; Rules $R$.
Output: Sampled $\hat{z}$
1  Counstruct tree: $root$ = CONSTRUCTTREE($z$);
2  Select atoms: $H$ = SELECTHEADS(list($root$));
3  Update root: UPDATE($H, y$);
4  while $H \neq \varnothing$ do
5      $H_{iter}$ = list();
6      for $h$ in $H$ do
7          Sample $B \sim \mathcal{U}(R|h)$;
8          Update child nodes: UPDATE($h$.child, $B$);
9          $H_{iter} = H_{iter}$ + SELECTHEADS($h$.child);
10     end
11     $H = H_{iter}$
12 end
13 return $\hat{z}$ = REDUCETOSEQUENCE($root$)
```

is shown in Alg. 2. In detail, we first construct a tree in reverse based on z (CONSTRUCTTREE). Then, the value of root node is updated by the ground truth y (UPDATE). Next, we uniformly sample and update (UPDATE) the values of child nodes B, and update atoms (SELECTHEADS) iteratively. Finally, the new root can be reduced to the sequence $\hat{z}$ with the same length of z (REDUCETOSEQUENCE). To further improve the efficiency of the sampling, we sample m intermediate variables and select the most probable one through $p_\theta(z|x)$. Based on such an algorithm, although it is still complex to compute the distribution matrix Q, $\frac{Q(z_i|z_j)}{Q(z_j|z_i)}$ can be calculated by Eq. 7 directly. $G(h, R)$ is a function to count how many rules in R that can imply h. h_i^k and h_j^k represent atoms in two z.

$$\frac{Q(z_i|z_j)}{Q(z_j|z_i)} = \frac{\prod_{h \in \{h_j^k | h_j^k \neq h_i^k\}} G(h, R)}{\prod_{h' \in \{h_i^k | h_i^k \neq h_j^k\}} G(h', R)} \quad (7)$$

Probabilistic Logic Regularization

Based on the back search algorithm, z^* is sampled as the pseudo label to train the neural network. As a result, the loss function can be computed by the cross-entropy loss $\mathrm{CE}(z, z^*)$. However, pseudo labels are noisy in the weakly supervised situation, so they cannot fully constrain the neural network through the cross-entropy loss. Therefore, we introduce a probabilistic logic regularization (PLR) to soften the reasoning process through probabilistic logic.

We assume that if input x is determined and contains the logic of $b_1 \wedge b_2 \rightarrow h$, then $p(h|x) = p(b_1|x)p(b_2|x) = 1.0$. Based on the assumption, we impose a constraint for each rule through conjunctive sentential decision diagrams (Darwiche 2011) adopt gradient semiring (Kimmig, den Broeck, and Raedt 2011; Manhaeve et al. 2018) to constrain the probabilistic logic, which is regarded as PLR. The gradient semiring is shown in Eq. 8, where p means $p_\theta(\cdot|x)$.

$$(p_h, \frac{\partial p_h}{\partial \theta}) = (p_{b_1}p_{b_2}, p_{b_1}\frac{\partial p_{b_2}}{\partial \theta} + p_{b_2}\frac{\partial p_{b_1}}{\partial \theta}) \quad (8)$$

Based on the gradient semiring of each node in a complete conjuntive sentential decision diagram, the potential

Model	Accuracy (%)
End2End	62.0 ± 0.0
DeepProbLog	63.5 ± 1.1
NGS	58.5 ± 0.0
WS-NeSyL	
$\boldsymbol{\lambda = 0}$	67.5 ± 0.0
$\boldsymbol{\lambda = 0.2}$	68.3 ± 0.9
$\boldsymbol{\lambda = 0.5}$	71.1 ± 0.1
$\boldsymbol{\lambda = 0.8}$	**72.9** ± 3.1

Table 1: Performance on MHTBD. λ is the hyper-parameter to reflect the proportion of PLR.

probabilistic constraint is $\forall h \in H, p_h = \prod_{b \in B} p_b$, where $B = \{b | b \wedge b^* \rightarrow h, b \in z^*\}$, $H = \{h | b^* \rightarrow h, h \in z^*\}$, h can be either an atom or a conjunction of atoms,and b^* is the conjunction of arbitrary literals. We use the differentiable mean square error to describe the constraint and PLR ($\mathcal{R}$) can be defined by in Eq. 9.

$$\mathcal{R} = \sum_{h \in H} \mathrm{MSE}(\prod_{b \in B} p_b, p_h) \tag{9}$$

As a result, the final objective function ($\mathcal{L}$) is shown in Eq. 10, where λ $(0 \leq \lambda \leq 1)$ is a hyper-parameter to adjust the proportion of the cross-entropy loss and PLR.

$$\mathcal{L} = (1 - \lambda) \times \mathrm{CE}(z, z^*) + \lambda \times \mathcal{R} \tag{10}$$

Experiments

Experimental Settings

Datasets. We experiment on three datasets: multi-hop TimeBank-Dense (MHTBD), multi-hop HWF (MHHWF), and CLUTRR (Sinha et al. 2019). **MHTBD** is a multi-hop version of TimeBank-Dense (Cassidy et al. 2014) with integrated multi-hop temporal reasoning instances, which includes a complex perception process (extracting information from text) but a simple reasoning process (with seven rules). In contrast, **MHHWF**, a multi-hop CV task modified on HWF (Sinha et al. 2019), has a simple perception process (recognizing handwritten digits from graphs) but a complex reasoning process since its mathematical operations are discretized into discrete logical rules. **CLUTRR**, a multi-hop relational reasoning dataset, includes a relatively complex perception process and reasoning process. Based on its original setting, it contains two generalization tasks and four robustness tasks. Generalization tasks are to benchmark the reasoning ability by forcing the model to be trained on low-hop instances (2-3/2-4 hops) and tested on high-hop inference (up to 10-hop reasoning). Robustness tasks are to evaluate the perception ability by training and testing models with different types of information organization (clean, supporting, irrelevant, and disconnected). In practice, we regroup the relationship defined in CLUTRR based on the work of Minervini et al. (2020).

Model	Training: k = 2-4		
	In.	Out.	Test
Pretrained	77.9 ± 0.3	50.0 ± 0.2	56.2 ± 0.1
DeepProbLog	90.2 ± 0.7	68.7 ± 0.7	73.5 ± 0.6
NGS	97.4 ± 0.2	91.5 ± 0.5	92.8 ± 0.4
WS-NeSyL			
$\boldsymbol{\lambda = 0}$	97.7 ± 0.4	93.7 ± 0.6	94.6 ± 0.5
$\boldsymbol{\lambda = 0.2}$	98.4 ± 0.5	95.2 ± 0.4	95.9 ± 0.4
$\boldsymbol{\lambda = 0.5}$	**98.5** ± 0.2	**96.2** ± 0.5	**96.7** ± 0.4
$\boldsymbol{\lambda = 0.8}$	97.4 ± 0.5	92.9 ± 0.9	93.9 ± 0.7

Table 2: Results on MHHWF. In. and Out. represent in-domain and out-of-domain tests, respectively. k is the number of hops of logical reasoning. λ is the hyper-parameter to represent the proportion of PLR.

Baselines. We adopt three baselines in this work, including the end-to-end neural network (**End2End**) and two NeSyL models: **DeepProbLog** (Manhaeve et al. 2018) and **NGS** (Li et al. 2020). In particular, we replace End2End with a pretrained model (**Pretrained**) based on a small amount of annotated data for MHHWF as End2End does not work. We adopt the same setting as Li et al. (2020) for NGS and only replace logical rules, while we adopt a different setting from Manhaeve et al. (2021) for DeepProbLog. For a fair comparison, we select the Bi-LSTM encoder and two LSTM attentional decoders on MHTBD and CLUTRR (Sinha et al. 2019), and use the CNN encoder and Multilayer Perceptrons decoders on MHHWF (Sinha et al. 2019) for baselines.

Results on MHTBD

Results on MHTBD are shown in Figure 1. WS-NeSyL outperforms all baselines more than three points. End2End and DeepProbLog can achieve better performances than NGS, while NGS almost fails on MHTBD. Compared with deterministic models, probabilistic models (including DeepProbLog and WS-NeSyL) perform better on MHTBD. Considering WS-NeSyL models, WS-NeSyL ($\lambda = 0.8$) achieves the best performance with an accuracy of 72.78%. Even when $\lambda = 0$ (without PLR), WS-SyNeL can also achieve a competitive performance (67.40%). In general, when the task has a simple reasoning process, WS-NeSyL can effectively use the back search algorithm to generate the reasoning process z, which is helpful for the neural network to perceive information from texts.

Results on MHHWF

Results on MHHWF are shown in Table 2. WS-NeSyL beats all baselines with a maximum accuracy of 96.70%, which performs well on both in-domain and out-of-domain evaluations. Besides, NGS has relatively good performance (92.84%), while the performance of DeepProbLog drops significantly (73.48%). Due to the complicated reasoning process in MHHWF, it is challenging for DeepProbLog to constrain probabilistic logic rules accurately. In addition, weakly supervised methods (NGS and WS-NeSyL) have advantages as they can effectively select logic rules. Further-

Model	Training: k = 2-3			Training: k = 2-4		
	In.	Out.	Test	In.	Out.	Test
End2End	76.1 ± 3.0	35.0 ± 0.5	40.2 ± 0.3	59.2 ± 3.9	37.4 ± 0.4	42.0 ± 0.8
DeepProbLog	**76.4** ± 3.3	31.9 ± 1.2	37.4 ± 1.3	**67.1** ± 2.9	37.0 ± 0.7	43.3 ± 1.2
NGS	70.0 ± 2.1	16.7 ± 7.1	23.3 ± 6.4	60.1 ± 9.6	36.7 ± 3.8	41.7 ± 4.2
WS-NeSyL						
$\lambda = 0$	68.8 ± 2.6	37.0 ± 0.3	41.0 ± 0.3	62.8 ± 7.4	40.2 ± 3.6	45.0 ± 2.1
$\lambda = 0.2$	70.8 ± 3.6	37.6 ± 0.9	41.8 ± 1.2	60.5 ± 4.6	41.2 ± 1.7	**45.3** ± 0.6
$\lambda = 0.5$	69.2 ± 2.4	38.0 ± 1.3	41.8 ± 1.0	59.2 ± 1.5	41.2 ± 1.1	45.0 ± 1.0
$\lambda = 0.8$	67.3 ± 3.5	**38.5** ± 0.9	**42.1** ± 0.6	58.9 ± 3.7	**41.3** ± 0.6	45.0 ± 0.7

Table 3: Generalization results on CLUTRR. In. and Out. represent in-domain and out-of-domain tests, respectively. k is the number of hops of logical reasoning. λ is the hyper-parameter to represent the proportion of PLR.

Model	Training: Clean					Training: Supporting				
	C	S	I	D	Test	C	S	I	D	Test
End2End	72.5 ± 1.8	68.1 ± 5.4	72.1 ± 4.7	65.6 ± 7.2	69.6 ± 3.5	71.2 ± 3.2	72.7 ± 5.0	75.8 ± 3.1	66.6 ± 3.1	71.6 ± 2.6
DeepProbLog	74.9 ± 3.9	55.7 ± 4.5	65.8 ± 4.1	68.2 ± 5.2	66.3 ± 3.3	75.5 ± 3.9	74.3 ± 1.6	75.5 ± 2.0	69.4 ± 4.2	73.7 ± 1.5
NGS	77.7 ± 10	64.9 ± 9.0	71.4 ± 9.8	71.2 ± 9.6	71.4 ± 8.3	67.2 ± 7.0	68.3 ± 2.9	67.3 ± 5.9	58.7 ± 5.8	65.4 ± 4.2
WS-NeSyL										
$\lambda = 0)$	71.2 ± 4.6	72.9 ± 2.9	75.2 ± 4.4	69.6 ± 3.8	72.2 ± 3.0	76.2 ± 2.9	73.6 ± 3.0	77.6 ± 2.7	70.5 ± 3.3	74.5 ± 1.3
$\lambda = 0.2$	71.6 ± 4.5	76.3 ± 3.4	76.3 ± 3.5	69.6 ± 3.5	73.4 ± 2.0	77.1 ± 3.7	76.7 ± 2.7	75.0 ± 2.0	69.7 ± 3.6	74.7 ± 1.0
$\lambda = 0.5$	74.5 ± 2.9	78.5 ± 2.3	77.8 ± 1.2	71.5 ± 3.7	75.6 ± 1.2	**78.7** ± 3.4	75.9 ± 3.1	77.2 ± 0.5	73.9 ± 3.5	76.4 ± 1.6
$\lambda = 0.8$	**79.0** ± 3.9	**81.5** ± 3.7	**81.8** ± 2.6	**78.9** ± 2.8	**80.3** ± 2.0	78.1 ± 4.0	**79.0** ± 2.5	**79.7** ± 3.4	**74.8** ± 3.0	**77.9** ± 1.8

Model	Training: Irrelevant					Training: Disconnected				
	C	S	I	D	Test	C	S	I	D	Test
End2End	48.8 ± 3.1	49.9 ± 2.5	52.5 ± 3.2	41.3 ± 2.6	48.2 ± 1.8	40.7 ± 2.7	40.3 ± 2.6	43.1 ± 4.9	41.2 ± 3.0	41.3 ± 1.9
DeepProbLog	54.6 ± 3.8	55.9 ± 3.5	60.0 ± 2.6	53.8 ± 6.5	56.1 ± 3.1	61.8 ± 5.0	55.2 ± 3.1	**63.5** ± 1.9	**64.5** ± 2.7	61.3 ± 2.4
NGS	52.0 ± 6.1	52.8 ± 2.7	55.3 ± 2.6	45.8 ± 5.7	51.5 ± 3.6	54.5 ± 8.5	53.9 ± 8.1	58.2 ± 9.4	58.6 ± 9.3	59.4 ± 10
WS-NeSyL										
$\lambda = 0$	57.4 ± 3.6	**56.8** ± 2.1	62.3 ± 2.5	50.4 ± 4.0	56.8 ± 2.2	62.1 ± 3.0	57.9 ± 3.8	57.0 ± 5.1	59.3 ± 3.7	59.1 ± 2.1
$\lambda = 0.2$	56.5 ± 1.3	54.8 ± 5.0	62.9 ± 2.2	54.2 ± 1.5	57.1 ± 1.0	58.4 ± 2.5	60.6 ± 2.1	57.0 ± 4.0	62.0 ± 1.0	59.3 ± 1.2
$\lambda = 0.5$	56.8 ± 3.3	56.5 ± 4.4	**66.7** ± 2.5	52.7 ± 5.1	58.3 ± 1.4	58.7 ± 1.1	61.5 ± 5.5	60.8 ± 3.3	63.6 ± 3.9	61.4 ± 2.7
$\lambda = 0.8$	**59.6** ± 3.5	56.8 ± 1.8	65.7 ± 6.3	**58.5** ± 2.5	**60.2** ± 2.6	**68.3** ± 6.3	**62.7** ± 6.0	62.8 ± 6.9	63.3 ± 4.0	**64.3** ± 4.2

Table 4: Robustness results on CLUTRR. C/S/I/D represent tests with Clean/Supporting/ Irrelevant/Disconnected instances. λ is the hyper-parameter to represent the proportion of PLR.

more, the sampling mechanism of WS-NeSyL is more stable than NGS's in logical reasoning, so the performance of WS-NeSyL is better than NGS. In conclusion, WS-NeSyL can adapt to cognitive tasks with the complex reasoning process.

Results on CLUTRR

Generalization Results. Generalization results on CLUTRR are shown in Table 3. WS-NeSyL outperforms all baseline models on both two evaluations, with maximum total accuracies of 42.07% ($\lambda = 0.8$) and 45.28% ($\lambda = 0.2$), respectively. In particular, WS-NeSyL performs better on out-of-domain evaluations (38.47% and 41.26%) because the weak supervision prevents overfitting of WS-NeSyL on the in-domain data. End2End does not show a significant difference in the performance on two tasks (40.16% and 41.99%) compared with the other three NeSyL models, although the second task provides more information about the reasoning process. This is evidence that End2End does not predict according to the reasoning process logically. Although NGS adopts a weakly supervised method, it can hardly learn the correct reasoning patterns on the first task (23.34%). One reason is that NGS cannot deal with the uncertainty in the logical reasoning, such as the disjunction rule of $Grandfather \wedge Son \rightarrow Father \vee Uncle$. Considering DeepProbLog, although the probabilistic logic can better describe the reasoning process of the neural network, the probabilistic loss alone is not enough to constrain the neural network efficiently. Overall, WS-NeSyL maintains relatively good reasoning ability even on cognitive tasks with complex perception and reasoning processes.

Robustness Results. Robustness results on CLUTRR are shown in Table 4. WS-NeSyL outperforms all three baselines on four tasks with accuracies of 80.28%, 77.90%, 60.21%, and 64.25%, respectively, as the back search mechanism can further enhance the perception ability of WS-

NeSyL by modifying the logic. End2End performs well on two noiseless tasks (69.63% and 71.60%), but the performance drops significantly on two noisy tasks (48.15% and 41.32%, respectively). DeepProbLog keeps stable performance as its symbolic reasoner is equivalent to providing the supervised label of the reasoning proof z. In conclusion, WS-NeSyL not only has extraordinary logical reasoning ability but also has better perception ability than other baselines under the same setting of the neural architecture.

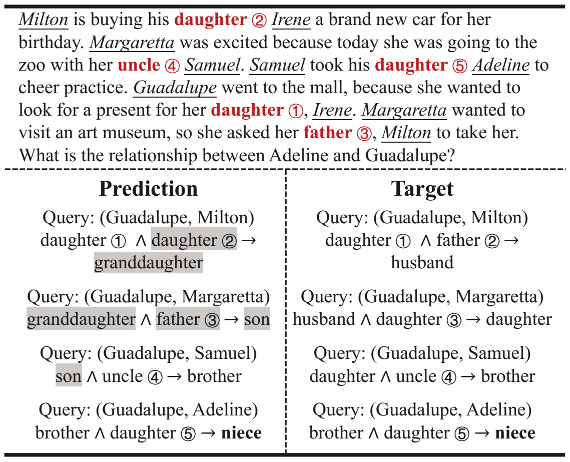
Analysis of PLR

We further explore the impacts of PLR on different datasets. On the one hand, PLR brings significant improvements on tasks benchmarking the perception ability (MHTBD and robustness evaluations of CLUTRR). Specifically, PLR brings at most a 5.38-point improvement on MHTBD and 7.11-point, 3.44-point, 3.44-point, and 5.16-point improvements on four robustness evaluations of CLUTRR, respectively. Meanwhile, with the increase of λ, the performance has always shown an upward trend. On the other hand, PLR does not bring significant improvements on those tasks that mainly evaluate the reasoning ability (MHHWF and generalization evaluations of CLUTRR). In particular, the performance of WS-NeSyL with PLR even drops when $\lambda = 0.8$, which is worse than WS-NeSyL without PLR.

Based on our analysis, PLR essentially provides local constraints for specific rules in the pseudo label. As a result, PLR can still provide reasonable constraints even under the noisy pseudo label as the logic rules always hold. On those tasks to benchmark the perception ability, PLR can alleviate negative influences of noisy pseudo labels on the neural network and benefit the correction of misperceived information. However, PLR essentially cannot provide sufficient constraints for the model to make the final prediction. This can be illustrated by setting $\lambda = 1$. WS-NeSyL will degenerate into weakly supervised DeepProbLog and cannot learn anything on any datasets. As a result, PLR alone cannot constrain the logical reasoning process on those tasks with more complex reasoning processes. In particular, when WS-NeSyL owns a strong perception ability (on MHHWF), an excessive proportion of PLR may even make the model unable to learn the reasoning efficiently.

Case Study and Error Analysis

To further understand WS-NeSyL, we perform a case study on CLUTRR and analyze the main error of WS-NeSyL. The case in Figure 4 gives a confusing example from CLUTRR. Although WS-NeSyL can predict the correct answer, it cannot completely recognize the whole reasoning chain correctly. In detail, we can observe that WS-NeSyL actually extracts words from the context but does not understand the true meaning of these kinships and the relationships of different kinships (such as the reversed relationship of $Daughter(x, y)$ and $Father(y, x)$). This means that the back search algorithm cannot correct such kind of perception error whose intermediate results match the logic rules. This is evidence that WS-NeSyL's ability to correct misperceived information is still limited. Furthermore, if WS-NeSyL perceives the wrong information, it may cater to the logic rules instead of modifying the information perceived.


Milton is buying his **daughter ②** *Irene* a brand new car for her birthday. *Margaretta* was excited because today she was going to the zoo with her **uncle ④** *Samuel*. *Samuel* took his **daughter ⑤** *Adeline* to cheer practice. *Guadalupe* went to the mall, because she wanted to look for a present for her **daughter ①**, *Irene*. *Margaretta* wanted to visit an art museum, so she asked her **father ③**, *Milton* to take her. What is the relationship between Adeline and Guadalupe?

Prediction	Target
Query: (Guadalupe, Milton) daughter ① ∧ daughter ② → granddaughter	Query: (Guadalupe, Milton) daughter ① ∧ father ② → husband
Query: (Guadalupe, Margaretta) granddaughter ∧ father ③ → son	Query: (Guadalupe, Margaretta) husband ∧ daughter ③ → daughter
Query: (Guadalupe, Samuel) son ∧ uncle ④ → brother	Query: (Guadalupe, Samuel) daughter ∧ uncle ④ → brother
Query: (Guadalupe, Adeline) brother ∧ daughter ⑤ → **niece**	Query: (Guadalupe, Adeline) brother ∧ daughter ⑤ → **niece**

Figure 4: Case Study of WS-NeSyL on CLUTRR. Shaded relations are mispredicted results.

Discussions

Comparing WS-NeSyL with NeSyL Mehtods

Although WS-NeSyL and NGS (Li et al. 2020) both adopt back search algorithms to weakly supervised the neural model, NGS cannot deal with uncertainty in logical reasoning, but WS-NeSyL can by nature due to its probabilistic framework. Besides, NGS adopts an approximate M-H sampler without a complete proposal distribution Q, so it essentially conducts one-step sampling and cannot meet the stable distribution condition in complex cognitive tasks. WS-NeSyL provides a multi-step sampler with a well-designed proposal distribution Q.

Inspired by DeepProbLog (Manhaeve et al. 2018), WS-NeSyL adopts the same probabilistic logic theory as DeepProbLog. However, WS-NeSyL imposes logical constraints locally, but DeepProbLog adopts a global probabilistic logic loss. Therefore, WS-NeSyL is capable of constraining local logical relations more refined than DeepProbLog.

Limitations of WS-NeSyL

There are two limitations of WS-NeSyL: (1) It is not flexible enough to automatically select different types of logic rules; (2) WS-NeSyL is difficult to cold start in complex logical reasoning scenarios because the model is hard to converge in weakly supervised manners. Therefore, it is challenging to apply WS-NeSyL to tasks with multiple inferences.

Conclusions and Future Work

In this paper, we propose WS-NeSyL that effectively enhances the perception and logical reasoning abilities to solve cognitive problems. Although WS-NeSyL has some limitations, experiments on three datasets have shown the powerful performance of WS-NeSyL. Future work will locate on applying NeSyL to more complex reasoning scenarios with multiple inferences through logic disentangling.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (2018YFC0830400), the ECNU-SJTU joint grant from the Basic Research Project of Shanghai Science and Technology Commission (19JC1410102), the Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), the Shanghai Pujiang Program (21PJ1407300), and the Shanghai Science and Technology Innovation Action Plan (20511102600).

References

Alvandi, P.; and Watt, S. M. 2019. Handwriting Feature Extraction via Legendre-Sobolev Matrix Representation. In *SYNASC*.

Cassidy, T.; McDowell, B.; Chambers, N.; and Bethard, S. 2014. An Annotation Framework for Dense Event Ordering. In *ACL*.

Chen, X.; Liang, C.; Yu, A. W.; Zhou, D.; Song, D.; and Le, Q. V. 2020. Neural Symbolic Reader: Scalable Integration of Distributed and Symbolic Representations for Reading Comprehension. In *ICLR*.

Dai, W.; Xu, Q.; Yu, Y.; and Zhou, Z. 2019. Bridging Machine Learning and Logical Reasoning by Abductive Learning. In *NeurIPS*.

Darwiche, A. 2011. SDD: A New Canonical Representation of Propositional Knowledge Bases. In *IJCAI*.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*.

Donadello, I.; Serafini, L.; and d'Avila Garcez, A. S. 2017. Logic Tensor Networks for Semantic Image Interpretation. In *IJCAI*.

Dong, H.; Mao, J.; Lin, T.; Wang, C.; Li, L.; and Zhou, D. 2019. Neural Logic Machines. In *ICLR*.

Dumancic, S.; Guns, T.; Meert, W.; and Blockeel, H. 2019. Learning Relational Representations with Auto-encoding Logic Programs. In *IJCAI*.

Evans, R.; and Grefenstette, E. 2018. Learning Explanatory Rules from Noisy Data. *J. Artif. Intell. Res.*

Gontier, N.; Sinha, K.; Reddy, S.; and Pal, C. 2020. Measuring Systematic Generalization in Neural Proof Generation with Transformers. In *NeurIPS*.

Hájek, P. 1998. *Metamathematics of Fuzzy Logic*, volume 4 of *Trends in Logic*. Kluwer.

Hastings, W. K. 1970. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*.

Johnson, J.; Hariharan, B.; van der Maaten, L.; Fei-Fei, L.; Zitnick, C. L.; and Girshick, R. B. 2017. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. In *CVPR*.

Kimmig, A.; den Broeck, G. V.; and Raedt, L. D. 2011. An Algebraic Prolog for Reasoning about Possible Worlds. In *AAAI*.

Li, Q.; Huang, S.; Hong, Y.; Chen, Y.; Wu, Y. N.; and Zhu, S. 2020. Closed Loop Neural-Symbolic Learning via Integrating Neural Perception, Grammar Parsing, and Symbolic Reasoning. In *ICML*.

Li, T.; and Srikumar, V. 2019. Augmenting Neural Networks with First-order Logic. In *ACL*.

Liu, J.; Cui, L.; Liu, H.; Huang, D.; Wang, Y.; and Zhang, Y. 2020. LogiQA: A Challenge Dataset for Machine Reading Comprehension with Logical Reasoning. In *IJCAI*.

Manhaeve, R.; Dumancic, S.; Kimmig, A.; Demeester, T.; and Raedt, L. D. 2018. DeepProbLog: Neural Probabilistic Logic Programming. In *NeurIPS*.

Manhaeve, R.; Dumančić, S.; Kimmig, A.; Demeester, T.; and Raedt, L. D. 2021. Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence*.

Marra, G.; Diligenti, M.; Giannini, F.; Gori, M.; and Maggini, M. 2020. Relational Neural Machines. In *ECAI*.

Metropolis, N.; Rosenbluth, A. W.; Rosenbluth, M. N.; Teller, A. H.; and Teller, E. 1953. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*.

Minervini, P.; Riedel, S.; Stenetorp, P.; Grefenstette, E.; and Rocktäschel, T. 2020. Learning Reasoning Strategies in End-to-End Differentiable Proving. In *ICML*.

Qu, M.; and Tang, J. 2019. Probabilistic Logic Neural Networks for Reasoning. In *NeurIPS*.

Ribeiro, M. T.; Guestrin, C.; and Singh, S. 2019. Are Red Roses Red? Evaluating Consistency of Question-Answering Models. In *ACL*.

Sampat, S. K.; Kumar, A.; Yang, Y.; and Baral, C. 2021. CLEVR_HYP: A Challenge Dataset and Baselines for Visual Question Answering with Hypothetical Actions over Images. In *NAACL-HLT*.

Sen, P.; and Saffari, A. 2020. What do Models Learn from Question Answering Datasets? In *EMNLP*.

Si, X.; Raghothaman, M.; Heo, K.; and Naik, M. 2019. Synthesizing Datalog Programs using Numerical Relaxation. In *IJCAI*.

Sinha, K.; Sodhani, S.; Dong, J.; Pineau, J.; and Hamilton, W. L. 2019. CLUTRR: A Diagnostic Benchmark for Inductive Reasoning from Text. In *EMNLP*.

Sourek, G.; Aschenbrenner, V.; Zelezný, F.; Schockaert, S.; and Kuzelka, O. 2018. Lifted Relational Neural Networks: Efficient Learning of Latent Relational Structures. *J. Artif. Intell. Res.*

Sugawara, S.; Stenetorp, P.; Inui, K.; and Aizawa, A. 2020. Assessing the Benchmarking Capacity of Machine Reading Comprehension Datasets. In *AAAI*.

Tsamoura, E.; Hospedales, T. M.; and Michael, L. 2021. Neural-Symbolic Integration: A Compositional Perspective. In *AAAI*.

Wang, P.; Stepanova, D.; Domokos, C.; and Kolter, J. Z. 2020. Differentiable learning of numerical rules in knowledge graphs. In *ICLR*.

Weber, L.; Minervini, P.; Münchmeyer, J.; Leser, U.; and Rocktäschel, T. 2019. NLProlog: Reasoning with Weak Unification for Question Answering in Natural Language. In *ACL*.

Xu, J.; Zhang, Z.; Friedman, T.; Liang, Y.; and den Broeck, G. V. 2018. A Semantic Loss Function for Deep Learning with Symbolic Knowledge. In *ICML*.

Yang, F.; Yang, Z.; and Cohen, W. W. 2017. Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In *NeurIPS*.

Yu, W.; Jiang, Z.; Dong, Y.; and Feng, J. 2020. ReClor: A Reading Comprehension Dataset Requiring Logical Reasoning. In *ICLR*.

Zadeh, L. A. 1965. Fuzzy Sets. *Inf. Control.*

Zhelezniakov, D.; Zaytsev, V.; and Radyvonenko, O. 2021. Online Handwritten Mathematical Expression Recognition and Applications: A Survey. *IEEE Access*.