Parsing Natural Language into Propositional and First-Order Logic with Dual Reinforcement Learning

Anonymous ACL submission

Abstract

Semantic parsing converts natural language paraphrases into structured logical expressions. In this paper, we consider two such formal representations: Propositional Logic (PL) and First-order Logic (FOL). Due to the insufficiency of annotated data in this field, we use dual reinforcement learning (RL) to make full use of labeled and unlabeled data. We further propose a brand new reward mechanism to avoid the trouble of manually defining the reward in RL. To utilize the training data efficiently and make the learning process consistent with humans, we integrate curriculum learning into our framework. Experimental results show that the proposed method outperforms competitors on different datasets. In addition to the technical contribution, we construct a Chinese-PL/FOL dataset to make up for the lack of data in this field. We aim to release our code as well as the dataset to aid further research in related tasks.

1 Introduction

Semantic parsing is the task of mapping natural language paraphrases into logical expressions. In the past few years, parsing unstructured text into logical expressions such as lambda calculus (Dong and Lapata, 2016, 2018; Zhao et al., 2019), SQL (Chang et al., 2020; Wang et al., 2020) and SPARQL (Shi et al., 2020; Das et al., 2021) have been extensively studied.

As two major logical forms of text representation, Propositional Logic (PL) and First-order Logic (FOL) have gradually come to the fore in recent years. Due to its strong logical reasoning ability and interpretability, PL/FOL plays an increasingly important role in natural language processing (NLP) tasks. Ma et al. (2020) improve the interpretability of RL policies by representing states and actions using FOL. Kimura et al. (2021) train a neural network by introducing directly interpretable FOL facts and logical operators to achieve fast convergence for the policy in RL. It is worth noting that all of these tasks have a prerequisite, i.e., parsing natural language paraphrases into PL/FOL. The results of parsing directly affect the performance of downstream tasks. Thus, it is crucial to have a strong semantic parser for PL/FOL.

Chinese-PL/FOL
Natural Language Paraphrase There is a natural number that is odd and not a multiple of 3. **Symbolic Definition** $A(x)$: x is odd; $B(x, y)$: x is a multiple of y; x: natural number; a: 3 **Logical Expression** $\exists x(A(x) \wedge (\neg B(x, a)))$

English-FOL
Natural Language Paraphrase The volunteers include executives and professionals. **Logical Expression** exists x1.(volunteer(x1) & exists x2. (executive(x2) & professional(x2) & include(x1, x2)))

Table 1: Examples of the Chinese-PL/FOL and English-FOL dataset. Note that symbol definition only appears in the Chinese-PL/FOL dataset.

Some researches have been conducted on the conversion between natural language and PL/FOL. One of the most typical methods is to treat the parsing task as a sequence-to-sequence (Seq2seq) generation problem (Singh et al., 2020; Levkovskyi and Li, 2021). However, this approach still has the following issues: 1) A large amount of labeled data is required. Although Levkovskyi and Li (2021) propose to generate data by templates, it leads to a lack of diversity of the data. 2) The previous works (Singh et al., 2020; Levkovskyi and Li, 2021) only consider unidirectional generation, i.e., from natural language to PL/FOL, and ignore that bidirectional generation can further enhance the performance of the models. 3) The training strategy of the existing models is simple and ignores the association between training samples.

To address these issues, we propose an effective framework for parsing natural language into

PL/FOL named **Dual-(m)T5** (in Section 3.1), and construct a new dataset called **Chinese-PL/FOL** (in Section 4) to evaluate the framework. Inspired by He et al. (2016); Cao et al. (2019), we model the learning of logical expressions and natural language generation as a dual task, which fully exploits labeled and unlabeled data. The prime task is parsing **N**atural **L**anguage into **L**ogical **E**xpressions (NL2LE) and the dual task is an inverse of the prime task, which aims to generate **N**atural **L**anguage given **L**ogical **E**xpressions (LE2NL). Each task requires a generation model, and both of the models are jointly trained via RL since the training process is non-differentiable.

We further design a brand new reward mechanism in RL. For logical expression, we propose a validity reward reflecting the structure of PL/FOL, which is an effective signal indicating whether the generated logical expression is well-structured. Different from the previous work (Cao et al., 2019), where only a rule-based validity reward is employed, we further propose a model-based validity reward, which can avoid the trouble of manually defining the validity reward in RL. To reduce information loss in the process of dual reinforcement learning, reconstruction reward is exploited to estimate the similarity between the input of the prime model and the output of the dual model.

Furthermore, due to the difference in the difficulty of training samples, we incorporate curriculum learning (Bengio et al., 2009) into our dual reinforcement learning framework, which makes the training process of models closer to the humans'. Experimental results show that our framework effectively parses natural language into PL/FOL and consistently improves performance compared to competitors.

The main contributions of this paper are summarized as follows:

- We propose an effective learning framework based on curriculum learning and dual reinforcement learning, which enables bidirectional conversion between natural language and PL/FOL. Experimental results show that the proposed method outperforms competitors on different datasets.

- We further propose a novel validity reward built upon the structure of PL/FOL, which is an effective signal indicating whether the generated logical expression is well-structured. A new strategy is proposed for training a scoring model that automatically computes the validity reward, which avoids the trouble of manually defining the reward in previous works.

- In addition to the technical contribution, we release a new Chinese-PL/FOL dataset that contains 1,263 Chinese-PL pairs and 1,464 Chinese-FOL pairs to make up for the lack of data and aid further research in this field.

2 Overview

In this section, we introduce the preliminary of PL/FOL and then formalize the problem definition.

2.1 Preliminary

FOL represents entities and actions in natural language through quantified variables and consists of predicates which take variables as arguments and attach semantics to variables (Blackburn and Bos, 2005), while PL is a relatively simple logical expression and does not deal with quantified variables. Formally, a predicate $P(v_1; v_2; ...; v_n)$ in PL/FOL is an n-ary function of variables v_i that are combined through logical connectives: *logical and* ($\wedge$), *logical or* ($\vee$), *logical not* ($\neg$), *logical implication* ($\rightarrow$), *logical equivalent* ($\leftrightarrow$). What's more, there are two types of quantifiers for FOL: *universal* ($\forall$) which specifies that sub-formula within its scope is true for all instances of the variable and *existential* ($\exists$) which asserts existence of at least one instance represented by a variable under which the sub-formula holds true.

2.2 Problem Definition

As shown in Table 1, given a natural language paraphrase s, the goal of this paper is to generate the corresponding logical expression e. Compared with the English-FOL dataset, the Chinese-PL/FOL dataset additionally includes the symbolic definition. We believe that the introduction of symbolic definition is beneficial because it guarantees the uniqueness of PL/FOL, while the predicates in the English-FOL dataset are not clear. However, the English-FOL dataset still has its place, since the symbolic definition is not always available in real-world scenarios. Therefore, we conduct experiments on both of the datasets. To facilitate the following description, we will take the English-FOL dataset as an example. The only difference is that the input of the models should include the symbol definition on the Chinese-PL/FOL dataset.

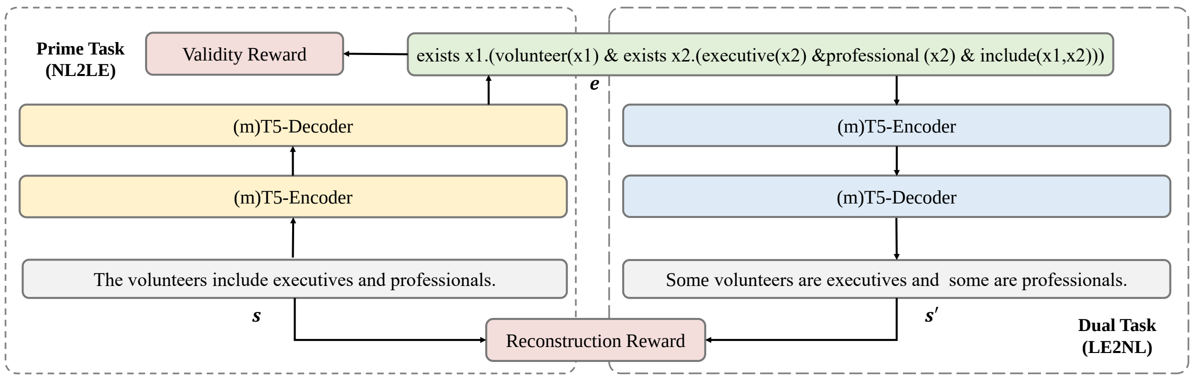
Prime Task (NL2LE)
Validity Reward
exists x1.(volunteer(x1) & exists x2.(executive(x2) &professional (x2) & include(x1,x2)))
e
(m)T5-Decoder
(m)T5-Encoder
The volunteers include executives and professionals.
s
(m)T5-Encoder
(m)T5-Decoder
Some volunteers are executives and some are professionals.
s′
Reconstruction Reward
Dual Task (LE2NL)

Figure 1: The architecture of dual reinforcement learning. The prime task and dual task form a closed cycle. The validity reward is used to estimate the quality of the generated logical expression and consists of both the rule-based and model-based reward. The reconstruction reward is exploited to estimate the similarity between the input s of the prime model and the output s' of the dual model.

3 Methdology

In this section, we present the details of the dual reinforcement learning algorithm for conversion between natural language and PL/FOL.

3.1 Framework

As shown in Figure 1, our framework consists of two sub-modules: the prime module NL2LE and the dual module LE2NL. The prime module adopts T5 (Raffel et al., 2019) / mT5 (Xue et al., 2021), a (multilingual) pre-trained text-to-text transformer, to generate the logical expression given a natural language sentence. The dual module uses another (m)T5 model to produce the sentence given a logical expression. These two modules in a closed-loop are trained by a reinforcement learning (RL) method based on policy gradient (Sutton et al., 2000). In RL, the state is denoted by the input of the prime module, i.e., natural language sentence s. The action in the prime and dual modules is defined as the logical expression and natural language sentence generation, respectively. The policy is denoted as the parameters of the (m)T5 models in the two modules.

3.2 Prime Module

The prime module (NL2LE) aims to transform natural language into PL/FOL. Given a sentence s, the encoder of (m)T5 is exploited to encode the input into a vector representation and the decoder learns to generate logical expression e depending on the encoding vector.

To ensure whether the generated logical expression is well-formed, we design a validity reward. Different from the previous work (Cao et al., 2019), where only rule-based validity reward is designed, we additionally introduce model-based validity reward by training a scoring model that can automatically score the intermediate logical expression (in Section 3.4.3).

Specifically, given a sentence s, the NL2LE model generates k possible logical expressions $e_1, e_2, \cdots, e_k$ via nucleus sampling (Holtzman et al., 2020). For each e_i, we can obtain a validity reward $R_e^{val}(e_i)$ composed of rule-based and model-based reward. For the rule-based reward, we estimate the quality of the generated logical expression by checking whether the logical expression is a complete tree without parentheses mismatching. Formally,

$$R_e^{rule\_val}(e_i) = \begin{cases} 1 & \textit{paired parentheses} \\ 0 & \textit{otherwise} \end{cases} \qquad (1)$$

which returns 1 when e_i has no error as mentioned above, and returns 0 otherwise. For the model-based reward, a scoring model is trained in advance. Given a logical expression e_i, the scoring model will give a reward $R_e^{model\_val}(e_i) \in [0, 1]$. Thus, the final validity reward is computed as:

$$R_e^{val}(e_i) = \alpha R_e^{rule\_val}(e_i) + (1-\alpha) R_e^{model\_val}(e_i) \qquad (2)$$

where a hyper-parameter $\alpha \in [0, 1]$ is exploited to balance these two rewards.

After feeding e_i into the LE2NL model (in Section 3.3), we get a reconstruction reward $R_s^{rec}(s \mid e_i)$ which forces the generated sentence as similar to the original sentence s as possible. A hyper-parameter β is exploited to balance these two rewards in r_i,

$$r_i = \beta R_e^{val}(e_i) + (1-\beta) R_s^{rec}(s \mid e_i) \quad (3)$$

where $\beta \in [0, 1]$. Let Θ_{NL2LE} denote all the parameters of the NL2LE model. By utilizing policy gradient (Sutton et al., 2000), the stochastic gradients of Θ_{NL2LE} is computed as:

$$\nabla_{\Theta_{NL2LE}} \hat{E}[r] = \frac{1}{k} \sum_{i=1}^{k} r_i \nabla_{\Theta_{NL2LE}} \log P(e_i \mid s; \Theta_{NL2LE}) \quad (4)$$

3.3 Dual Module

The dual module (LE2NL) is an inverse of the prime module, which aims to generate natural language sentences given PL/FOL expressions. Formally, the input is the logical expression e_i generated in the prime task, and the model is expected to output the original sentence s. Reconstruction reward is used to estimate the similarity between the input of the prime model and the output of the dual model. Here, we take log-likelihood as a reconstruction reward. Let Θ_{LE2NL} denote all the parameters of the LE2NL model. The reconstruction reward can be formulated as:

$$R_s^{rec}(s \mid e_i) = \log P(s \mid e_i; \Theta_{LE2NL}) \quad (5)$$

By utilizing policy gradient, the stochastic gradients of Θ_{LE2NL} is computed as:

$$\nabla_{\Theta_{LE2NL}} \hat{E}[r] = \frac{1-\beta}{k} \sum_{i=1}^{k} \nabla_{\Theta_{LE2NL}} \log P(s \mid e_i; \Theta_{LE2NL}) \quad (6)$$

3.4 Training Details

In this section, we supplement our training details, and answer these questions:

- How to avoid training collapse in the process of dual reinforcement learning? (Section 3.4.1)
- Since the amount of the labeled data is limited, how to use the labeled data effectively? (Section 3.4.2)
- What is the scoring model in Section 3.2 and how is it trained? (Section 3.4.3)

3.4.1 Supervisor Guidance & Reward Baseline

In practice, we find that if the models are trained with only the rewards from dual reinforcement learning, the training process will easily collapse. To keep the training process stable and prevent the models from crashing, we fine-tune both of the models with the labeled data before dual reinforcement learning starts. What's more, after each update according to Eq.(4) and Eq.(6), the models are trained with the labeled data again, i.e., both of the models are trained with dual reinforcement learning and supervised learning alternately.

Besides supervisor guidance, to cope with high variance in reward signals, we generate k intermediate outputs as mentioned in Section 3.2 and re-define reward signals via a reward baseline to stabilize the training process. We investigate different reward baseline choices, and it performs best when we use the average of rewards within samples per input. Thus, the final validity reward R_e^{val} and reconstruction reward R_s^{rec} are as follows:

$$R_e^{val}(e_i) = R_e^{val}(e_i) - \frac{1}{k} \sum_{i=1}^{k} R_e^{val}(e_i) \quad (7)$$

$$R_s^{rec}(s \mid e_i) = R_s^{rec}(s \mid e_i) - \frac{1}{k} \sum_{i=1}^{k} R_s^{rec}(s \mid e_i) \quad (8)$$

3.4.2 Curriculum Learning

Intuitively, there is a difference in the difficulty of training samples. To utilize the training data effectively and make the learning process consistent with humans, we integrate curriculum learning (Bengio et al., 2009) into the training process. The curriculum is arranged by sorting each sample into training sets according to a specific ranking standard. Here, we consider the length of logic expressions as an indicator of the learning order, i.e., the longer the logical expression, the more difficult it is. We first sort the training samples according to the length of the logical expressions. At each training step t, a batch of training samples is obtained from the top $f(t)$ portions of the entire sorted training samples. Following Platanios et al. (2019), $f(t)$ is defined as:

$$f(t) = \min\left(1, \sqrt{\frac{t\left(1-c_0^2\right)}{T} + c_0^2}\right) \quad (9)$$

where c_0 represents the models start training using the $c_0\%$ easiest training samples, and T represents the duration of curriculum learning. Note that curriculum learning is only used in the process of supervised learning, and does not appear in the process of dual reinforcement learning.

3.4.3 Training of the Scoring Model

Inspired by Shen et al. (2021), we design a **Generation & Classification** method to train the scoring model as follows:

Before the training of the scoring model starts, a NL2LE model should be fine-tuned first. The process of fine-tuning can refer to **Supervisor Guidance**. After that, given a natural language sentence s, the fine-tuned NL2LE model generates k possible logical expressions $e_1, e_2, \cdots, e_k$ via nucleus sampling. Since the NL2LE model has been fine-tuned, the generated logical expressions will be similar or equal to the ground truth e. We denote $\{\langle s, e_i \rangle\}$ as $\mathcal{P}$, where the logical expression e_i equals to ground truth, and the others as $\mathcal{N}$.

For each $\langle s, e_i \rangle$ pair, we get the last layer hidden states of the NL2LE model's encoder $h_{i1}^{encoder}, \cdots, h_{in}^{encoder}$, and decoder $h_{i1}^{decoder}, \cdots, h_{im}^{decoder}$. Then, the scoring model is defined as follows:

$$\overline{h}_i^{encoder} = \frac{1}{n} \sum_{j=1}^{n} h_{ij}^{encoder} \tag{10}$$

$$\overline{h}_i^{decoder} = \frac{1}{m} \sum_{j=1}^{m} h_{ij}^{decoder} \tag{11}$$

$$u_i = \overline{h}_i^{encoder} W_1 + b_1 \tag{12}$$

$$v_i = \overline{h}_i^{decoder} W_2 + b_2 \tag{13}$$

$$R_e^{model\_val}(e_i) = P(e_i \mid s) \tag{14}$$

$$= sigmoid\left([u_i; v_i; |u_i - v_i|]\, W_3 + b_3\right) \tag{15}$$

where $W_{1|2|3}$ and $b_{1|2|3}$ are trainable parameters and $[\cdot;\cdot]$ is the concatenation operation. The training loss L of the scoring model is cross-entropy loss between the model's output $P(e_i \mid s)$ and labels,

$$L = -\frac{1}{|\mathcal{P} \cup \mathcal{N}|} \left(\sum_{e_i \in \mathcal{P}} \log P(e_i \mid s) + \sum_{e_i \in \mathcal{N}} (1 - \log P(e_i \mid s)) \right) \tag{16}$$

Note that the weight of the NL2LE model is fixed, and only the scoring model is updated during the backpropagation.

The details of the dual reinforcement learning are provided in Algorithm 1, and the training process of the scoring model is provided in Algorithm 2.

4 Dataset Collection

To make up for the lack of data in this field and to verify the effectiveness of our proposed framework, we construct a dataset containing natural language and PL/FOL pairs. One way to do so is to define templates first, and then obtain samples by filling slots (Levkovskyi and Li, 2021). However, the resulting dataset is limited by the lack of diversity of templates. Instead, in this work, we use crowdsourcing to avoid this problem.

Since the task requires professional knowledge about PL/FOL, the crowdsourcing team consists of 8 Chinese graduate students who have a deep understanding of PL/FOL. If the crowd workers are required to construct data without any reference, this will introduce inevitable troubles and labeling errors since PL/FOL is not intuitive to humans. Aiming to reduce nontrivial human labor and ensure the quality of the dataset, our crowdsourcing process consists of the following steps:

1. We first obtain PL/FOL exercise sets and exam papers that require students to convert natural language into PL/FOL from Baidu Wenku[1], one of the largest online platforms for sharing documents in China.

2. The crowd workers are asked to organize these exercises in a uniform format. Each sample consists of three parts: natural language sentence s, symbolic definition d, and logical expression e, as shown in Table 1.

3. After that, we de-duplicate and annotate the data so that each sample is annotated by two other crowd workers. Note that PL/FOL is not visible at this stage, and the crowd workers are asked to provide the PL/FOL according to natural language sentences and symbolic definitions. We keep the sample only when both of the crowd workers have the same answer as the initial one.

[1] https://wenku.baidu.com/

	PL	FOL	TOTAL
Training	871	1,037	1,908
Validation	128	145	273
Test	264	282	546
Total	1263	1464	2727

Table 2: Statistics of the Chinese-PL/FOL Dataset

In this way, we obtain a total of 2,727 samples consisting of 1,263 PL and 1,464 FOL with the corresponding natural language paraphrases and symbolic definitions. To establish human performance, we ask an additional 3 undergraduate and 2 graduate students who have acquired basic knowledge of PL/FOL to provide logical expressions given natural language sentences and symbolic definitions from the entire test set. We follow Levkovskyi and Li (2021) and take Exact Match (EM) as an evaluation measure. The detailed statistics of the dataset are shown in Table 2.

5 Experiments

In this section, we evaluate our framework on the English-FOL (Levkovskyi and Li, 2021) and our Chinese-PL/FOL datasets.

5.1 Dataset

English-FOL It is generated by pre-defined templates and contains natural language paraphrases paired with FOL. We follow the training/validation/test splits as Levkovskyi and Li (2021).

Chinese-PL/FOL The details of our dataset have been introduced in Section 4. Since symbol definition only appears in this dataset, we concatenate it with the original input, i.e. natural language paraphrase in the prime module and logical expression in the dual module. Due to the small amount of training data, PL and FOL are trained together.

Unlabeled Data Since neither the English-FOL nor our Chinese-PL/FOL dataset provides unlabeled data, to test our framework in a semi-supervised setting, we design two schemes: In scheme **A**, we keep a part of the training set as fully labeled data and leave the rest as unlabeled data where only natural language paraphrases should be used. In scheme **B**, off-the-shelf paraphrase generation models are leveraged, which can generate synonymous sentences from existing natural language paraphrases in the datasets. According to our observation, since the paraphrase generation models are trained on paraphrase generation datasets that are different from the datasets we use, the generated synonymous sentence is not particularly similar to the original one, and the logical expressions corresponding to the two sentences are different in most cases. Thus, it is reasonable to treat the generated sentences as unlabeled data. Refer to Appendix.A for the details of the paraphrase generation models and hyperparameter settings.

5.2 Overall Results

We compare our method with competitors on different datasets in Table 3. T5 is used on the English-FOL dataset and mT5 is used on the Chinese-PL/FOL dataset. *B.Unlabel* represents unlabeled data obtained by scheme **B** in Section 5.1.

From the results, we conclude that: **1)** Our models outperform the competitors on both of the datasets, which shows the effectiveness and robustness of our models. The performance of the models is not affected by the different languages, which shows the versatility of our models. **2)** Even without additional unlabeled data, our models outperform the competitors only with supervised learning, e.g., Dual-T5-**base** gets much better performance than T5-**base** on the English-FOL dataset. (Actually, Dual-T5-**base** even has better performance than T5-**large**). **3)** By introducing the unlabeled data, the performance of the models is further improved. The improvement on the Chinese-PL/FOL dataset is more obvious than that on the English-FOL dataset. We believe that this is because the amount of the labeled data in the English-FOL dataset is enough for the training of the models, and it is difficult for the models to use the unlabeled data for further improvement, while the models can make full use of the unlabeled data to compensate for the lack of the labeled data on the Chinese-PL/FOL dataset. To confirm the performance of the models in a semi-supervised setting, we conduct further experiments in section 5.4.

5.3 Effectiveness on Small Data

To investigate the effectiveness of our method on small data, we set different labeled ratios for the training of the models. Note that unlabeled data is not used in this setting, and we only vary the ratio of the labeled data kept on the English-FOL dataset from 20% to 100%.

In Table 4, we can find that our models have better performance over all labeled ratios. Specifically, Dual-T5-**base** still has a better performance than

Method	English-FOL	Chinese-PL/FOL		
		PL	FOL	TOTAL
Human Performance	-	87.94	79.92	84.07
text2log (Levkovskyi and Li, 2021)	89.54	-	-	-
T5-small (Raffel et al., 2019) / mT5-small (Xue et al., 2021)	89.95	64.02	58.87	61.35
T5-base (Raffel et al., 2019) / mT5-base (Xue et al., 2021)	91.30	70.08	61.35	65.57
T5-large (Raffel et al., 2019)	92.59	-	-	-
Dual-(m)T5-small **(Ours)**	90.98	64.39	61.70	63.00
Dual-(m)T5-base **(Ours)**	92.63	70.83	62.77	66.67
Dual-(m)T5-small + *B.Unlabel* **(Ours)**	91.03	70.83	63.48	67.03
Dual-(m)T5-base + *B.Unlabel* **(Ours)**	**92.82**	**75.00**	**68.44**	**71.61**

Table 3: EM on the test set of the English-FOL and Chinese-PL/FOL datasets. T5 is used on the English-FOL dataset and mT5 is used on the Chinese-PL/FOL dataset. *B.Unlabel* represents the unlabeled data obtained by scheme B in Section 5.1.

Method	Labeled Ratio				
	20%	40%	60%	80%	100%
text2log (Levkovskyi and Li, 2021)	47.19	77.06	85.31	88.53	89.54
T5-small (Raffel et al., 2019)	77.02	84.57	86.56	89.87	89.95
T5-base (Raffel et al., 2019)	84.58	88.05	89.34	90.81	91.30
T5-large (Raffel et al., 2019)	85.87	88.79	89.56	91.97	92.59
Dual-T5-small **(Ours)**	78.26	85.60	88.51	90.94	90.98
Dual-T5-base **(Ours)**	**86.97**	**89.82**	**90.64**	**92.23**	**92.63**

Table 4: EM on the test set of the English-FOL dataset. It varies the ratio of the labeled data.

T5-**large** over all labeled ratios, which indicates that even without using additional unlabeled data, our model is capable of steadily outperforming larger scale models. The performance of text2log (Levkovskyi and Li, 2021) decreases rapidly with the reduction of the labeled data, which proves that the robustness of the model in previous works needs to be enhanced.

5.4 Experiments on Semi-supervised Setting

To investigate whether the unlabeled data benefits our framework and how much unlabeled data may lead to the best result, we perform the following experiment. We fix the ratio of the labeled data as 20% and change the ratio of the unlabeled data to the rest of the data on the English-FOL dataset, i.e., scheme **A** mentioned in Section 5.1. The results are shown in Table 5.

Method	Unlabeled Ratio	FOL
Dual-T5-small	0%	78.26
+ *A.Unlabel*	25%	**79.29**(+1.03)
+ *A.Unlabel*	50%	79.18(+0.92)
+ *A.Unlabel*	75%	78.45(+0.19)
+ *A.Unlabel*	100%	77.85(-0.41)
Dual-T5-base	0%	86.97
+ *A.Unlabel*	25%	88.95(+1.98)
+ *A.Unlabel*	50%	89.18(+2.21)
+ *A.Unlabel*	75%	**89.34**(+2.37)
+ *A.Unlabel*	100%	88.98(+2.01)

Table 5: EM on the test set of the English-FOL dataset. It fixes the ratio of the labeled data as 20% and varies the ratio of the unlabeled data to the rest data. *A.Unlabel* represents the unlabeled data obtained by scheme A in Section 5.1.

We can find that the performance of the models doesn't improve constantly when the amount of unlabeled data is increased, which is consistent with the previous work (Cao et al., 2019). We conclude that the performance of the model should be determined by two factors: **1)** the relative proportion of labeled and unlabeled data, **2)** the number of parameters in the model. A proper ratio of unlabeled data is crucial, and a model with more parameters tends to perform better with more unlabeled data.

On the contrary, when a model with few parameters is trained with a large amount of unlabeled data by dual reinforcement learning, it may converge to a wrong equilibrium state to adapt to the unlabeled data and forget what has been learned from the labeled data, which leads to poor performance.

5.5 Ablation Analysis

To enhance the performance of our framework, we introduce dual reinforcement learning and curriculum learning. To evaluate the effectiveness of each of them, we perform an ablation analysis on the Chinese-PL/FOL dataset. The results are shown in Table 6.

Method	PL	FOL	TOTAL
Dual-mT5-small	**64.39**	**61.70**	**63.00**
w/o curriculum	64.02	60.64	62.27
w/o dual	63.64	59.57	61.54
Dual-mT5-base	**70.83**	**62.77**	**66.67**
w/o curriculum	70.45	62.06	66.12
w/o dual	70.08	61.70	65.75

Table 6: EM of Dual-mT5 ablations on the test set of the Chinese-PL/FOL dataset.

From the results, we conclude that: **1)** Both dual reinforcement learning and curriculum learning are helpful for the task since the effect of the models becomes worse in the absence of any learning method. **2)** Dual reinforcement learning has a greater impact on model performance than curriculum learning when labeled data is limited. **3)** Curriculum learning has a greater impact on FOL than PL. This is intuitive because FOL expressions are more complex than PL expressions, and curriculum learning helps the models gradually adapt to difficult samples.

6 Related Works

Parsing Natural Language into PL/FOL Logic expressions are commonly written in standardized mathematical notation, and learning this notation typically requires many years of experience. Barker-Plummer et al. (2009) study why students find translating natural language sentences into FOL hard and systematically categorize the problems encountered by students. Bansal (2015) proposes a rule-based framework that leverages the Part-of-speech structure of natural language sentences. Limited to the manually defined rules and a small amount of experimental data, the system can only work under a specific setting. With the development of deep learning, neural approaches alleviate the need for manually defining lexicons. Singh et al. (2020) examine the capability of neural models on parsing FOL from natural language sentences. They propose to disentangle the representations of different token categories while generating FOL and use category prediction as an auxiliary task. Unfortunately, they do not release the dataset they construct. Levkovskyi and Li (2021) release a dataset containing English-FOL sentence pairs and set up a baseline encoder-decoder model, but the dataset is not challenging for it is generated by templates, and the vanilla model can get a high score.

Dual Learning Dual learning is first proposed to improve neural machine translation (NMT) (He et al., 2016). The author makes full use of monolingual corpus to improve the effectiveness of the model through dual learning. Xia et al. (2017) introduce a probabilistic duality term to serve as a data-dependent regularizer to better guide the dual supervised learning. Since then, the idea of dual learning has been applied in various tasks, such as Question Answering/Generation (Tang et al., 2017), Open-domain Information Extraction/Narration (Sun et al., 2018), Semantic Parsing with lambda calculus (Cao et al., 2019, 2020), and Emotion-Controllable Response Generation (Shen and Feng, 2020). We are the first to introduce the curriculum and dual reinforcement learning in conversion between natural language and PL/FOL to the best of our knowledge.

7 Conclusion

In this paper, we introduce Dual-(m)T5, an effective framework based on curriculum and dual reinforcement learning, which enables bidirectional conversion between natural language and PL/FOL. We also propose a brand new reward mechanism to avoid manually defining the reward in RL. Experimental results show that the proposed method outperforms competitors on the datasets. In addition to the technical contribution, a new Chinese-PL/FOL dataset is constructed to make up for the lack of data in this field. In the future, we will further supplement our dataset since the size of the current dataset is not large. We will also exploit more lightweight models to accelerate the training process of dual reinforcement learning.

References

Naman Bansal. 2015. *Translating Natural Language Propositions to First Order Logic*. Ph.D. thesis, INDIAN INSTITUTE OF TECHNOLOGY KANPUR.

Dave Barker-Plummer, Richard Cox, and Robert Dale. 2009. Dimensions of difficulty in translating natural language into first order logic. *International Working Group on Educational Data Mining*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 41–48. ACM.

Patrick Blackburn and Johannes Bos. 2005. *Representation and inference for natural language: A first course in computational semantics*. Center for the Study of Language and Information Amsterdam.

Ruisheng Cao, Su Zhu, Chen Liu, Jieyu Li, and Kai Yu. 2019. Semantic parsing with dual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 51–64, Florence, Italy. Association for Computational Linguistics.

Ruisheng Cao, Su Zhu, Chenyu Yang, Chen Liu, Rao Ma, Yanbin Zhao, Lu Chen, and Kai Yu. 2020. Unsupervised dual paraphrasing for two-stage semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6806–6817, Online. Association for Computational Linguistics.

Shuaichen Chang, Pengfei Liu, Yun Tang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Zero-shot text-to-sql learning with auxiliary task. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7488–7495. AAAI Press.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay-Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. *arXiv preprint arXiv:2104.08762*.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–742, Melbourne, Australia. Association for Computational Linguistics.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 820–828.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Daiki Kimura, Masaki Ono, Subhajit Chaudhury, Ryosuke Kohita, Akifumi Wachi, Don Joven Agravante, Michiaki Tatsubori, Asim Munawar, and Alexander Gray. 2021. Neuro-symbolic reinforcement learning with first-order logic. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3505–3511.

Oleksii Levkovskyi and Wei Li. 2021. Generating predicate logic expressions from natural language. In *SoutheastCon 2021*, pages 1–8. IEEE.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Zhihao Ma, Yuzheng Zhuang, Paul Weng, Dong Li, Kun Shao, Wulong Liu, Hankz Hankui Zhuo, and HAO Jianye. 2020. Interpretable reinforcement learning with neural symbolic logic.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. 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 1162–1172, Minneapolis, Minnesota. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Jianhao Shen, Yichun Yin, Lin Li, Lifeng Shang, Xin Jiang, Ming Zhang, and Qun Liu. 2021. Generate & rank: A multi-task framework for math word problems. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2269–2279.

Lei Shen and Yang Feng. 2020. CDL: Curriculum dual learning for emotion-controllable response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 556–566, Online. Association for Computational Linguistics.

Jiaxin Shi, Shulin Cao, Liangming Pan, Yutong Xiang, Lei Hou, Juanzi Li, Hanwang Zhang, and Bin He. 2020. Kqa pro: A large diagnostic dataset for complex question answering over knowledge base. *arXiv e-prints*, pages arXiv–2007.

Hrituraj Singh, Milan Aggrawal, and Balaji Krishnamurthy. 2020. Exploring neural models for parsing natural language into first-order logic. *arXiv preprint arXiv:2002.06544*.

Jianlin Su. 2021. Roformer-sim: Integrating retrieval and generation into roformer. Technical report.

Mingming Sun, Xu Li, and Ping Li. 2018. Logician and orator: Learning from the duality between language and knowledge in open domain. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2119–2130, Brussels, Belgium. Association for Computational Linguistics.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, Online. Association for Computational Linguistics.

Yingce Xia, Tao Qin, Wei Chen, Jiang Bian, Nenghai Yu, and Tie-Yan Liu. 2017. Dual supervised learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3789–3798. PMLR.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Zijian Zhao, Su Zhu, and Kai Yu. 2019. A hierarchical decoding model for spoken language understanding from unaligned data. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 7305–7309. IEEE.

A Implemention Details

We use Pytorch[2] library for implementing an auto-differentiable graph of our computations. We leverage the HuggingFace's implementation of (m)T5[3]. For **supervisor guidance**, (m)T5-small/base are trained with an AdamW optimizer (Loshchilov and Hutter, 2018) initialized with a learning rate of 1e-3/1e-4 with a decay rate of 1e-3/1e-2 respectively. For **dual reinforcement learning**, models are trained with an AdamW optimizer initialized with a learning rate of 1e-5 with a decay rate of 1e-3 for (m)T5-small/base. The batch size is set to 8 for **supervisor guidance** and 2 for **dual reinforcement learning**, and the max input and output sentence length are set to 128. Training runs until the performance on validation set does not improve.

We use PEGASUS (Zhang et al., 2020) fine-tuned for paraphrasing[4] for english paraphrasing, and RoFormer-Sim (Su, 2021) [5] for chinese paraphrasing. For each natural language paraphrase in the datasets, we generate one synonymous sentence as unlabeled data. (Generating top-k synonymous sentences for each natural language paraphrase, where $k \geq 1$, may lead to poor performance because the unlabeled data is too similar to each other).

Our models run on a computer with Intel(R) Xeon(R) Gold 6230R CPU, 2 GeForce RTX 3090, 64GB of RAM, and Ubuntu 20.04.

[2]https://pytorch.org

[3]https://huggingface.co/models

[4]https://huggingface.co/tuner007/pegasus_paraphrase

[5]https://github.com/ZhuiyiTechnology/roformer-sim

B Algorithm

Algorithm 1 Dual Reinforcement Learning

Input: Supervised dataset $\mathcal{L} = \{\langle s, e\rangle\}$; Unsupervised dataset $\mathcal{U} = \{s'\}$; number of nucleus sampling k; hyper parameters α and β; curriculum training batches T
Output: NL2LE model

```
 1: Fine-tune NL2LE model with $\langle s, e\rangle$ from $\mathcal{L}$ and curriculum learning based on Eq.(9)
 2: Fine-tune LE2NL model with $\langle e, s\rangle$ from $\mathcal{L}$
 3: repeat
 4:   Get mini-batch $\{s\}$ from $\mathcal{L} \cup \mathcal{U}$
 5:   for all $s \in \{s\}$ do
 6:     NL2LE model generates $k$ logical expressions $\{e_i\}$ for $s$ via nuclelus sampling
 7:     for all $e_i \in \{e_i\}$ do
 8:       Obtain validity reward for $e_i$ w.r.t. Eq. (2)
 9:       Obtain reconstruction reward for $e_i$ w.r.t. Eq. (5)
10:       Compute total reward for $e_i$ w.r.t. Eq.(3)
11:     end for
12:   end for
13:   Compute gradient of $\Theta_{NL2LE}$ w.r.t. Eq.(4)
14:   Compute gradient of $\Theta_{LE2NL}$ w.r.t. Eq.(6)
15:   Update $\Theta_{NL2LE}$ and $\Theta_{LE2NL}$ with gradient
16:   Get mini-batch $\{\langle s, e\rangle\}$ from $\mathcal{L}$
17:   Fine-tune NL2LE model with $\{\langle s, e\rangle\}$
18:   Fine-tune LE2NL model with $\{\langle e, s\rangle\}$
19: until NL2LE model converges
```

Algorithm 2 Training Scoring Model

Input: Supervised dataset $\mathcal{L} = \{\langle s, e\rangle\}$; number of nucleus sampling k; Fine-tuned NL2LE model
Output: scoring model

```
 1: $\mathcal{P} \leftarrow \{\}, \mathcal{N} \leftarrow \{\}$
 2: for all $\langle s, e\rangle \in \mathcal{L}$ do
 3:   Given $s$, fine-tuned NL2LE model generates $k$ logical expressions $\{e_i\}$ via nucleus sampling
 4:   for all $e_i \in \{e_i\}$ do
 5:     if $e_i$ equals $e$ then
 6:       $\mathcal{P} \leftarrow \mathcal{P} \cup \{\langle s, e_i\rangle\}$
 7:     else
 8:       $\mathcal{N} \leftarrow \mathcal{N} \cup \{\langle s, e_i\rangle\}$
 9:     end if
10:   end for
11: end for
12: repeat
13:   Update scoring model w.r.t. Eq.(16)
14: until scoring model converges
```