A Logical Pattern Memory Pre-trained Model for Entailment Tree Generation

Li Yuan [1, 2], Yi Cai [1, 2 , *] Haopeng Ren [1, 2] , Jiexin Wan [1, 2,]

1School of Software Engineering, South China University of Technology
2 Key Laboratory of Big Data and Intelligent Robot (South China University of Technology) Ministry of Education

LREC-COLING 2024

The code and data are available at https:// https://github.com/YuanLi95/T5-LMPM

CONTENTS

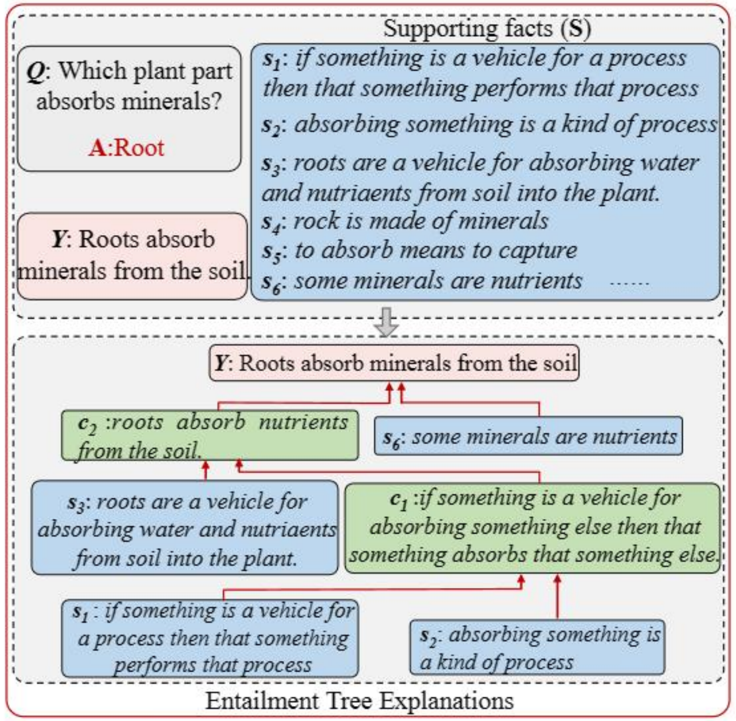


Figure 1: Illustration of Entailment Tree Generation. The top half presents the inputs, while the generated entailment tree is depicted in the bottom half. The tree consists of a hypothesis (pink), premises (blue), and generated intermediate conclusions (green).

Generating coherent and credible explanations poses a significant challenge in AI, and addressing it represents a giant leap towards the goal of **building reliable reasoning systems**.

In recent years, researchers have explored the utilization of **entailment trees for explanation generation**.

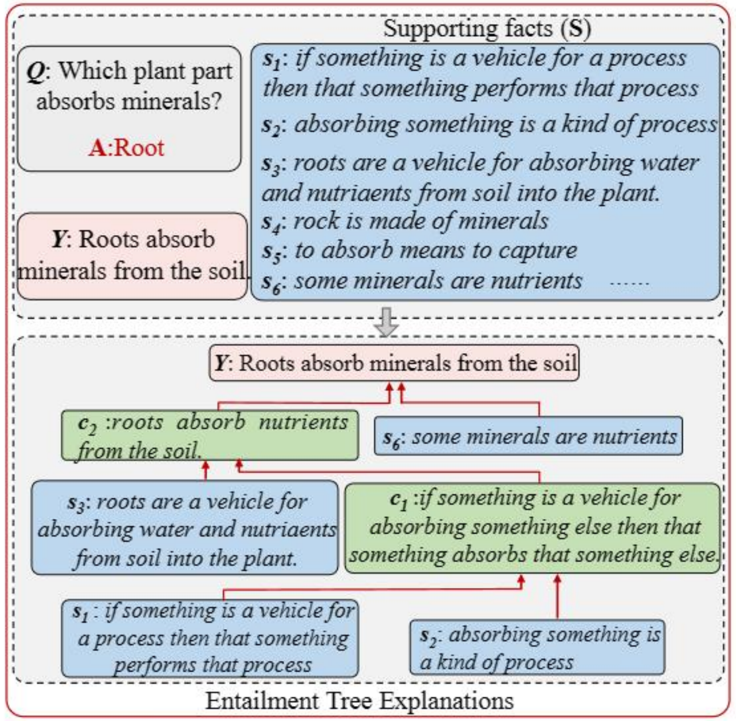


Figure 1: Illustration of Entailment Tree Generation. The top half presents the inputs, while the generated entailment tree is depicted in the bottom half. The tree consists of a hypothesis (pink), premises (blue), and generated intermediate conclusions (green).

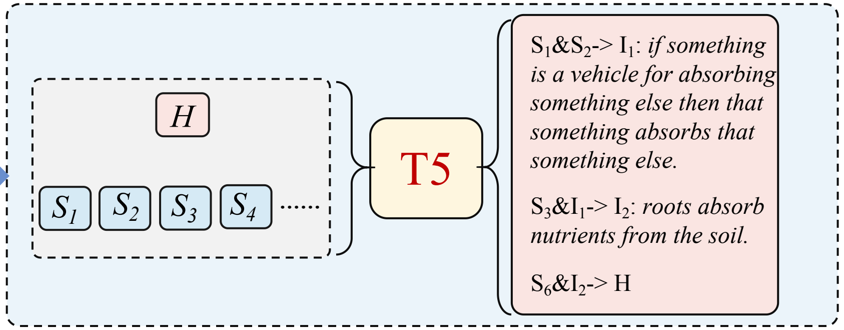
Generating coherent and credible explanations poses a significant challenge in AI, and addressing it represents a giant leap towards the goal of **building reliable reasoning systems**.

In recent years, researchers have explored the utilization of **entailment trees for explanation generation**.

The task of entailment tree generation **can be defined as follows**: given a hypothesis (summarizing from a question+answer pair) and a set of supporting facts, the goal is to derive an entailment tree where each non-leaf node is an intermediate conclusion generated from its children

Previous works

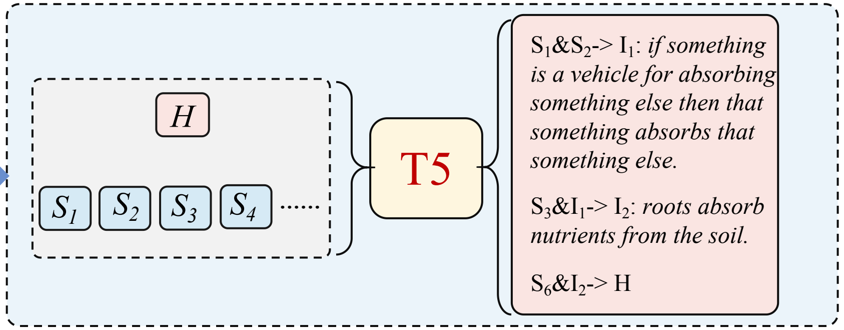
1. Single-step method[1]



Drawback：
Unreliable explanations for the hypothesis[2]

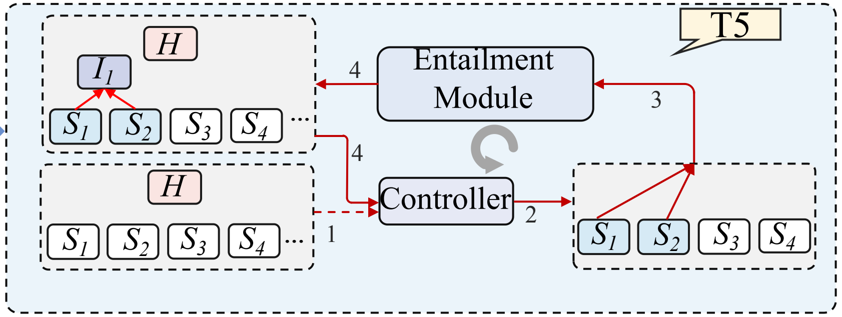
Previous works

1. Single-step method[1]



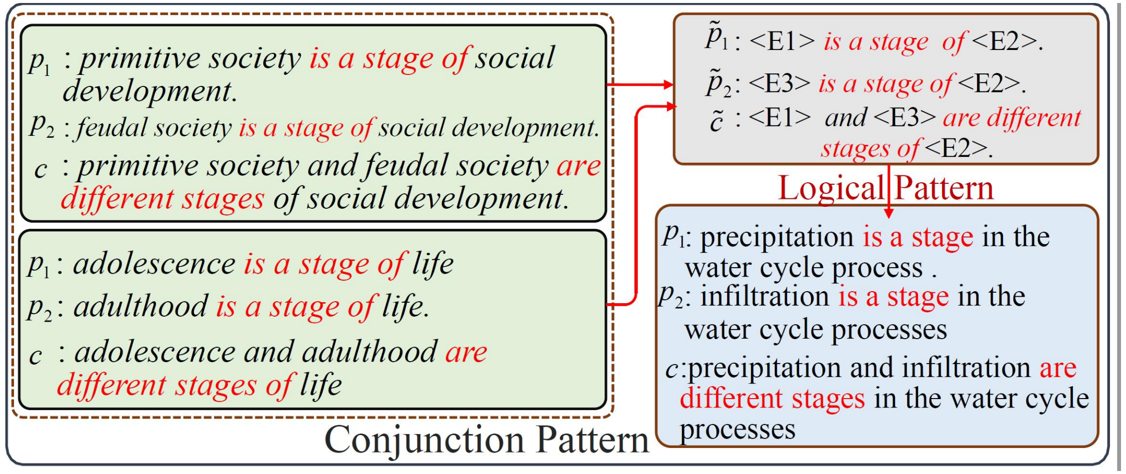
Drawback：
Unreliable explanations for the hypothesis[2]

2. multi-step generation method [3-6]



Drawback：
Iterative methods lies in their limited attention to the logical patterns between premises and conclusions

METGEN



Although, METGEN [3]stands out for its attempt to integrate **logical patterns** into the tree generation process.

METGEN's strategy for integrating logical patterns involves training the language model on a synthetic dataset constructed from Wikipedia (**Green boxes**).

Drawbacks:

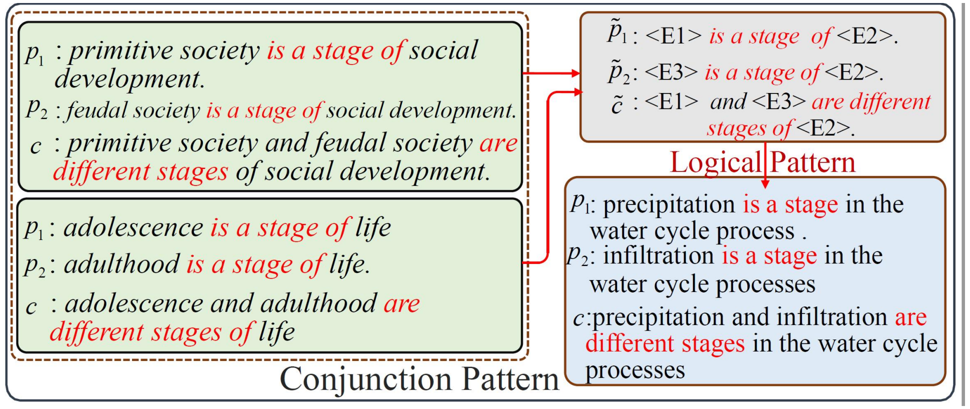
1. During training, the language model is forced to learn both logical patterns and textual features, which dilutes its focus on capturing logical patterns.

2. Using Wikipedia data as a training resource introduces a considerable amount of **domain-specific information that isn't logically relevant.**

Our Method

Key: Map the implicit **logical patterns onto the embedding space**.

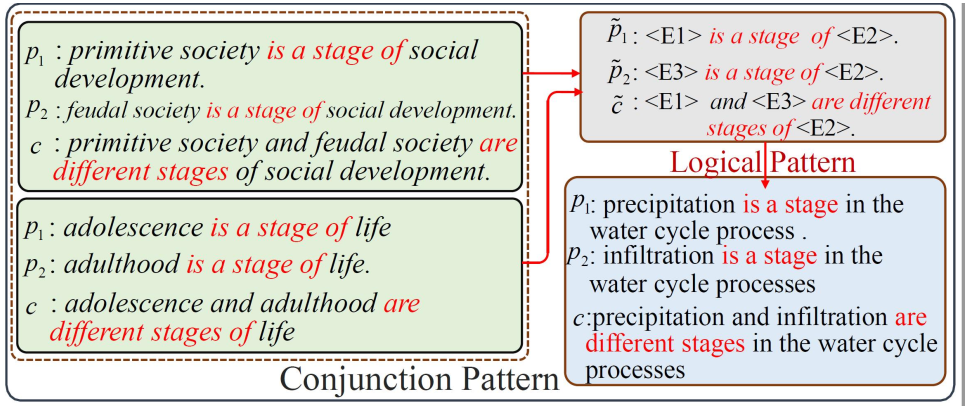
1. Entity abstraction techniques



Our Method

Key: Map the implicit **logical patterns onto the embedding space**.

1. Entity abstraction techniques



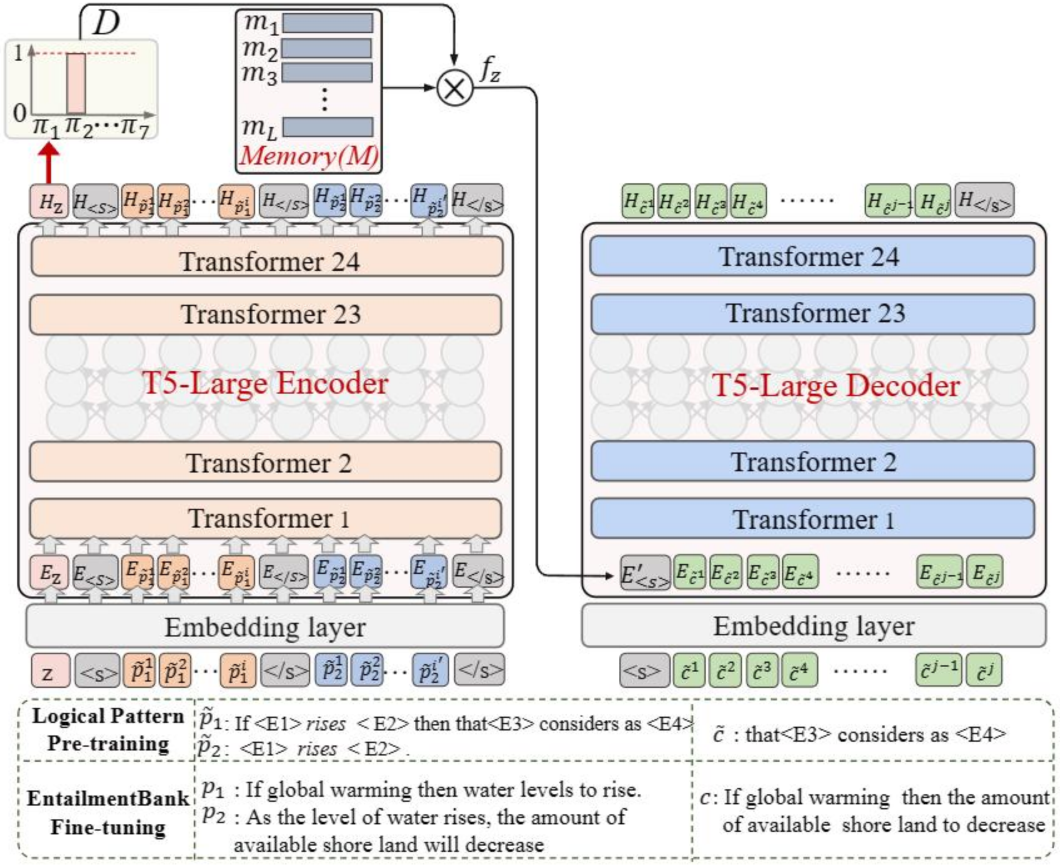


Figure 3: The overall architecture of LMPM

A Logical Pattern Memory Pre-trained Model

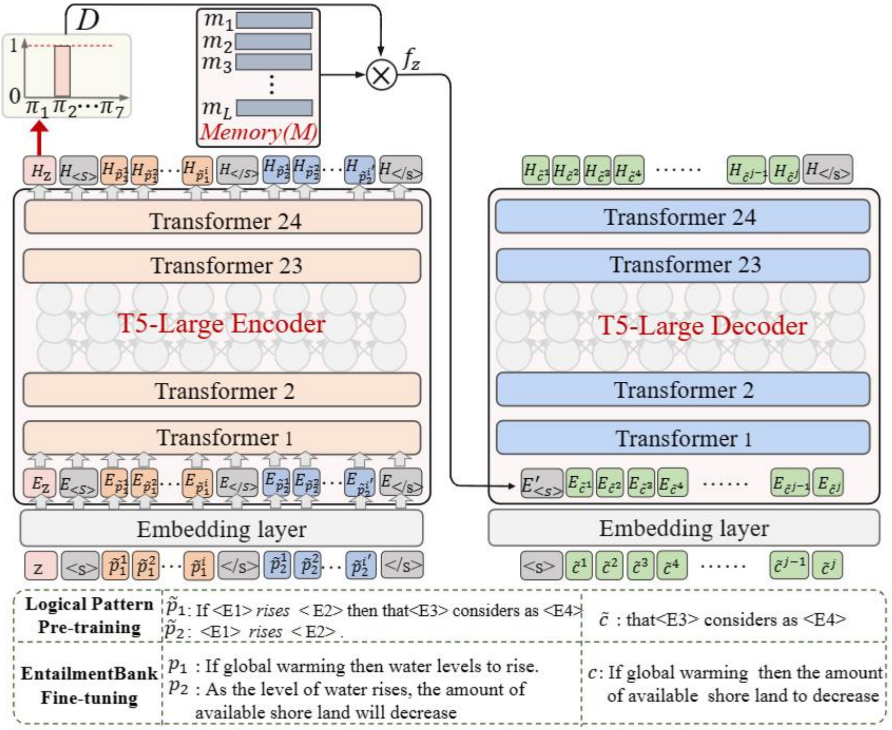


Figure 3: The overall architecture of LMPM

Step 1: Logical Pattern Pre-training

Enhance its ability to capture and learn latent logical pattern

$$I_{\text{logical}} = [<z>,<s>,\tilde{p}_1^1,...,\tilde{p}_1^l,</s>,\tilde{p}_2^1,...,\tilde{p}_2^{l'},</s>]$$

$$M = [m_1, ..., m_L] \in \mathbb{R}^{L \times d_m}$$

where, each element m_i in M indicates a specific logical pattern

Then, we introduce an address structure D, which employs a multi-layer perceptron architecture.

$$\gamma_i = w_z H_z + b_z$$

Gumbel-softmax

$$\alpha_i = \frac{\exp(\gamma_i + g_i)/\mathcal{T}}{\sum_{i=1}^{L} \exp(\gamma_i + g_i)/\mathcal{T}}$$

This structure takes Hz as input and produces a one-hot vector to help locate the potential logical pattern representation in *M*.

$$f_z = \sum_{i=1}^{L} \alpha_i M_i$$

Step 1: Logical Pattern Pre-training

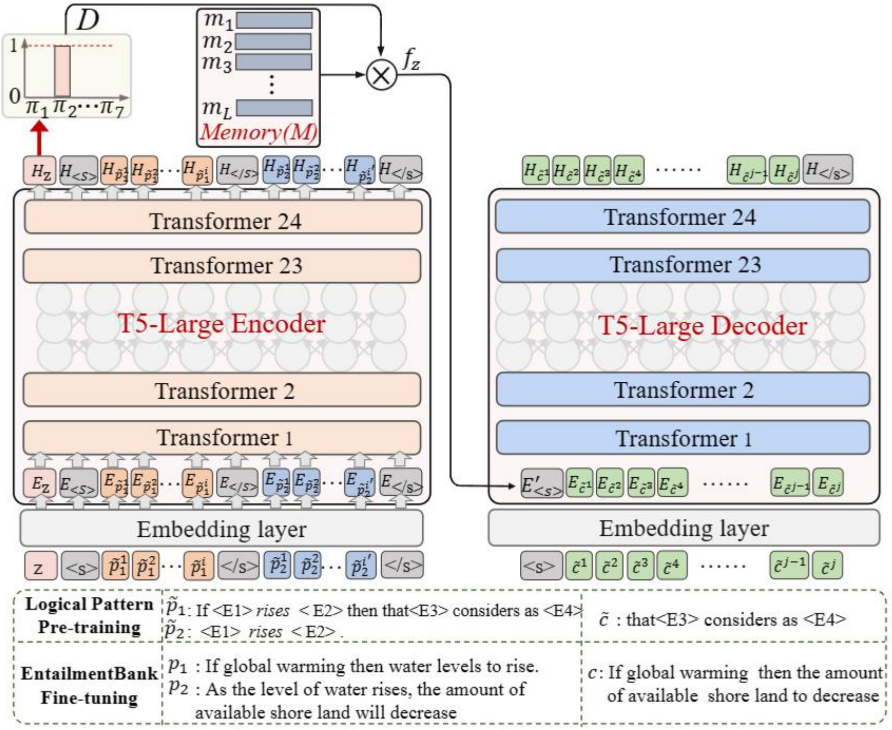


Figure 3: The overall architecture of LMPM

we integrate the selected logical pattern representation into the T5-decoder by adding it to the representation of the start special token <s>.

$$E'_{<s>} = f_z + E_{<s>}$$

Loss Function:

1. language modeling loss (LM)

$$\mathcal{L}_{LM} = -\sum_{t=1}^{|\tilde{c}|} \log p(\tilde{c}^t | f_z, \tilde{p}_1, \tilde{p}_2, \tilde{c}^{0:t-1})$$

2. To tackle the issue of vanishing latent variables a bag-of-word loss (BOW)

$$L_{BOW} = \sum_{t=1}^{|\tilde{c}|} \log p(\tilde{c}^t | f_z, \tilde{p}_1, \tilde{p}_2) = \sum_{t=1}^{|\tilde{c}|} \log \frac{e^{f(\tilde{c}^t)}}{\sum_{v \in V} e^{f(v)}}$$

$$\mathcal{L} = \mathcal{L}_{LM} + \mathcal{L}_{BOW}$$

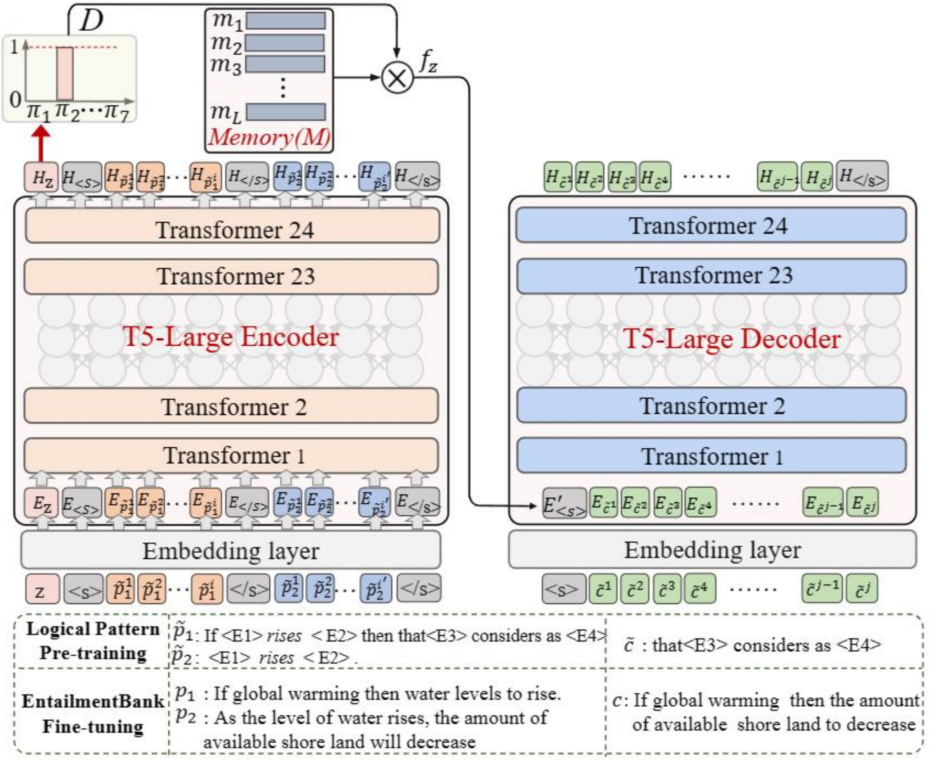
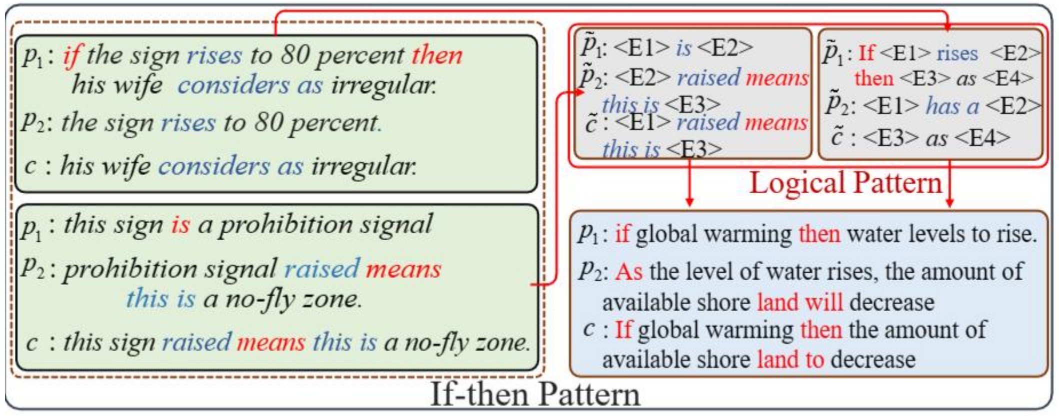


Figure 3: The overall architecture of LMPM

Step 2:EntailmentBank Fine-tuning:

Adapting the pre-trained logical pattern representations to the target EntailmentBank dataset.

The dataset includes some complex deductions that can not be adequately captured by a single logical pattern stored in M



$$\alpha_i = \frac{\exp(\gamma_i + g_i)/\mathcal{T}}{\sum_{i=1}^{L} \exp(\gamma_i + g_i)/\mathcal{T}}$$

$$\widetilde{\alpha_i} = \frac{\exp(\gamma_i)}{\sum_{i=1}^{L} \exp(\gamma_i)}$$

Softmax

Results

Method	Additional Data	Leaves F_1	Leaves AllCorrect	Steps F_1	Steps AllCorrect	Intermediates F_1	Intermediates AllCorrect	Overall AllCorrect
Task1								
EntailmentWriter(T5-large)	No	98.4	84.1	50.0	38.5	67.0	35.9	34.4
IRGR	No	97.6	89.4	50.2	36.8	62.1	31.8	32.4
RLET	No	**100.0**	**100.0**	55.0	41.2	67.2	36.7	35.1
METGEN+T5	564K	**100.0**	**100.0**	57.7	41.9	70.8	39.2	36.5
METGEN+LMPM(*ours*)	564K	99.76	99.41	**57.78**	**43.82†**	**72.78†**	**42.78†**	**38.54†**
Task2								
EntailmentWriter(T5-large)	No	**83.2**	35.0	39.5	24.7	**62.2**	28.2	23.2
IRGR	No	69.9	23.8	30.5	22.40	47.70	26.5	21.8
RLET	No	81.9	40.4	38.8	28.7	57.4	29.1	26.0
METGEN+T5	564K	82.7	46.1	41.3	29.6	61.4	32.4	27.7
METGEN+LMPM(*ours*)	564K	81.09	**47.06**	**42.56**	**31.38 †**	61.68	**34.32†**	**29.41†**
Task3								
EntailmentWriter(T5-large)	No	30.9	1.2	4.4	1.2	28.8	5.6	1.2
IRGR	No	**46.6**	10.0	11.3	8.2	38.7	**20.9**	8.2
RLET	No	46.20	**11.41**	**15.2**	**9.6**	**41.4**	17.6	9.4
METGEN+T5	564K	34.8	8.7	9.8	8.6	36.6	20.4	8.6
METGEN+LMPM(*ours*)	564K	35.3	9.23	10.28	9.23	37.8	20.33	**9.41**

Table 2: Automatic evaluation results on the EntailmentBank dataset (%). The best and second-best results are highlighted in bold and underlined, respectively. † refers to significance test $p - value < 0.05$. The “Additional Data" column denotes the size of the supplementary Wikipedia training data.

Mtrics	METGEN+T5	METGEN+LMPM
Validity	46	52
Logical	3.02	3.37
Readability	4.11	4.43
Reasonability	3.37	3.64

Table 3: Human evaluation results on 100 uniformly sampled questions from the test split.

Ablation Study

Method	Task 1 Inter	Task 1 Overall	Task 2 Inter	Task 2 Overall
METGEN+T5	39.2	36.5	32.4	27.7
METGEN+LMPM	42.78	38.54	34.32	29.41
w/o *LPP*	39.02	36.47	32.24	27.56
w/o *memory*	40.59	37.39	33.06	28.54
w/o *abstraction*	41.27	37.45	33.78	28.48

Table 4: Ablation study results. “Inter" and “Overall" denote Intermediates AllCorrect and Overall AllCorrect, respectively.

Three parts: logical pattern pre training (LPP), memory structure (memory), and dataset abstraction (abstraction)

Logical Pattern Data Size

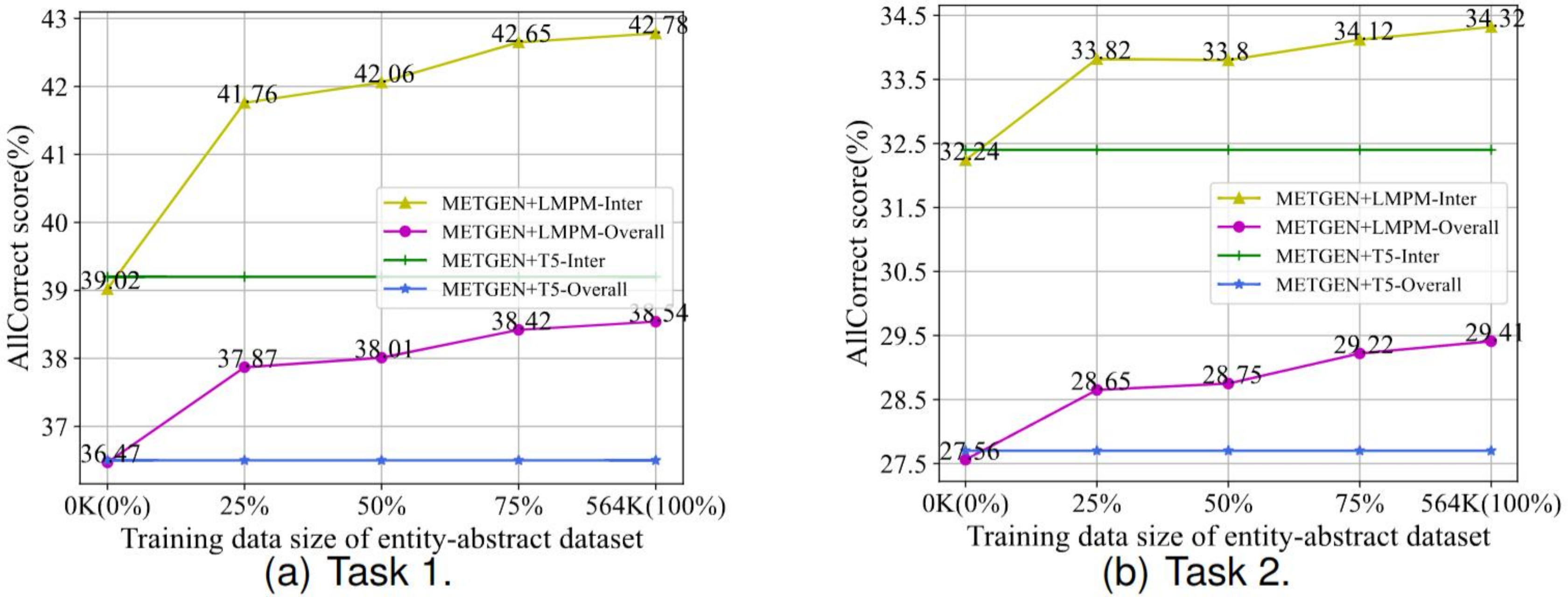


Figure 5: The impact of logical pattern pre-training data size on performance. The AllCorrect scores for Overall and Inter (Intermediates) are provided.

Additional Analysis

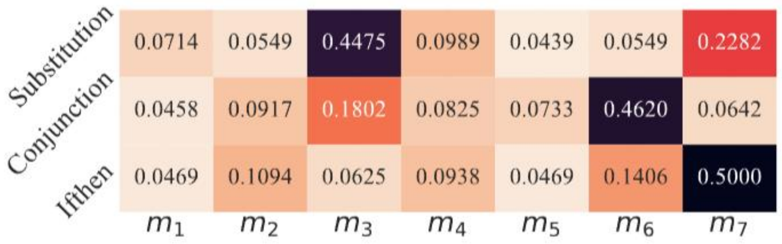


Figure 6: Probability distribution of the three logical patterns within the memory structure M.

We analyze a sample of **275 intermediate conclusions provided by METGEN**, along with their automatically annotated inference types. The distribution of these samples is as follows: **Substitution (153), Conjunction (71), and IF-then (51).**

1. We introduce a **logical pattern memory pre-trained model** (LMPM), which facilitates the generation of logically consistent intermediate conclusions during entailment steps.

2. LMPM employs an external memory to **learn and retain latent logical patterns** between premises and conclusions. This mechanism significantly enhances the language model's capacity to **capture and utilize logical patterns effectively**.

3. We propose to pre-train the LMPM model via a **constructed entity-abstract dataset**. This approach **mitigates the influence of irrelevant domain knowledge** in the original Wikipedia data and enables the model to be well-trained with less data.

4. we plan to **extend this method to other tasks**, such as the Multimodel QA, and to explore the incorporation of external knowledge to enhance generation

Thanks and QA

WeChat:

Email: seyuanli@mail.scut.edu.cn

Code : *https://github.com/YuanLi95/T5-LMPM*