Natural Logic at the Core: Dynamic Rewards for Entailment Tree Generation

Jihao Shi, Xiao Ding*, Kai Xiong, Hengwei Zhao, Bing Qin, and Ting Liu
Research Center for Social Computing and Interactive Robotics
Harbin Institute of Technology, China
{jhshi, xding, kxiong, hwzhao, qinb, tliu} @ir.hit.edu.cn

Abstract

Entailment trees are essential for enhancing interpretability and transparency in tasks like question answering and natural language understanding. However, existing approaches often lack logical consistency, as they rely on static reward structures or ignore the intricate dependencies within multi-step reasoning. To address these limitations, we propose a method that integrates natural logic principles into reinforcement learning, enabling dynamic reward computation to guide entailment tree generation. Our approach ensures logical consistency across reasoning steps while improving interpretability and generalization. Experiments on EntailmentBank demonstrate significant improvements over state-of-the-art methods, highlighting the effectiveness of natural logic in structured reasoning.

1 Introduction

In recent years, the generation of entailment trees (Dalvi et al., 2021; Yuan et al., 2024; Song et al., 2024) and the application of reinforcement learning (Liu et al., 2022; Chen et al., 2024) for knowledge selection have garnered considerable attention within the domain of automated reasoning. These methodologies aspire to model the structure and inference of knowledge, playing a crucial role in applications such as natural language understanding (Dalvi et al., 2021), automated theorem proving (Yang et al., 2022), and question answering (Hong et al., 2023). However, extant approaches frequently encounter a significant limitation: the lack of global logical constraints that direct the reasoning process. Whether incrementally constructing the entailment tree (Dalvi et al., 2021; Bostrom et al., 2022; Hong et al., 2023) or utilizing reinforcement learning (Liu et al., 2022; Chen et al., 2024) to select pertinent knowledge, these methods often lack a comprehensive framework that guarantees logical consistency throughout the reasoning process. Consequently, the model is vulnerable to deviating from the correct reasoning pathway, which may result in inaccuracies that compromise the overall correctness of the final output.

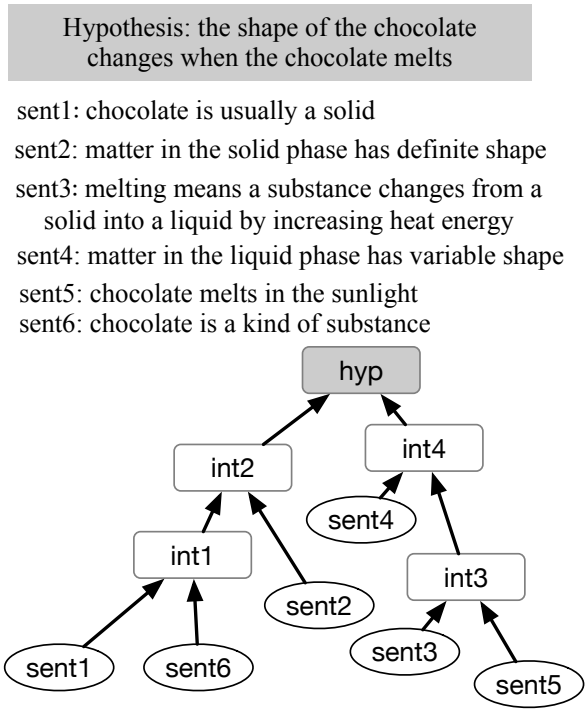

Figure 1: Example tree from EntailmentBank. Given a hypothesis h (a declarative sentence derived from a question-answer pair) and a set of facts (or corpus), the objective is to produce a structured explanation that clearly outlines the reasoning process from the facts to the hypothesis.

To address this issue, we propose a novel approach, **NLDR**, which uses **N**atural **L**ogic-based **D**ynamic **R**ewards to facilitate entailment tree construction. Unlike prior methods that rely on fixed reward structures (Chen et al., 2024; Liu et al., 2022), our approach employs entailment scores derived from natural logic as dynamic rewards to guide the model's reasoning process. These dy-

*Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2025, pages 17372–17382
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

namic rewards enable the model to assess the logical consistency of each reasoning step, providing feedback that steers it towards the optimal reasoning path while helping to avoid local optima.

One of the primary strengths of natural logic (Lakoff, 1970; MacCartney and Manning, 2009; Angeli et al., 2016) is its ability to capture a wide range of logical relationships inherent in natural language. By leveraging these relationships, our methodology provides nuanced logical guidance throughout the process of entailment tree generation. Figure 1 illustrates an example of how a hypothesis and a set of facts lead to the structured explanation of a reasoning process. As illustrated in Figure 1, our method constructs an entailment tree by progressively combining relevant facts to reach the hypothesis. At each step of reasoning, the model selects a set of premises from the candidate sentences and generates an intermediate conclusion. The logical consistency of this inference is evaluated using natural logic operations (e.g., entailment, alternation, contradiction), and the resulting logical relation is fed into a deterministic finite automaton (DFA) to determine whether the step contributes to a valid reasoning path. The natural logic-based dynamic reward is computed from this evaluation and used to guide the reinforcement learning process. This mechanism enables the model to receive fine-grained feedback at each reasoning step, ensuring that the constructed entailment tree adheres to a globally consistent logical structure rather than only optimizing for the final output. Employing entailment scores derived from natural logic as reward values enables the model to assess the logical consistency of each reasoning step, ensuring that the reasoning trajectory remains both precise and consistent. This dynamic reward mechanism also allows the model to evaluate the logical soundness of intermediate steps. Moreover, the incorporation of natural logic-based rewards enhances the interpretability and transparency of the reasoning process, as these rewards directly reflect clear and explainable logical relationships.

The principal contributions of this work are delineated as follows:

- We present a logic-guided dynamic reward mechanism, wherein the entailment scores derived from natural logic are employed as rewards to direct the reasoning process, thereby ensuring logical consistency and enhancing the generation of entailment trees.
- We extend the scope of reinforcement learning to encompass not only step selection but also intermediate generation, thereby permitting the model to assess and refine the logical soundness of the reasoning process at every stage, rather than solely at the final conclusion.

2 Related Work

2.1 Entailment Tree Generation

Entailment trees, first introduced by Dalvi et al. (2021), provide a structured framework for modeling multi-step logical reasoning in NLP tasks, particularly question answering (QA). These trees illustrate how hypotheses logically follow from a series of substantiated premises. Since then, several methods have emerged to enhance entailment tree generation, including approaches by Bostrom et al. (2022) and Ribeiro et al. (2022), which focus on reconstructing entailment trees from natural language inputs. Advances have also been made in integrating entailment trees with neuro-symbolic reasoning (Tafjord et al., 2022; Weir and Van Durme, 2022), enabling more interpretable and flexible models. Recent work, such as that by Yang et al. (2022) and Sprague et al. (2022), has further expanded this research by introducing methods for handling incomplete data and enhancing the reliability of logical deductions.

Additionally, reinforcement learning (RL) has been integrated into entailment tree generation to refine and optimize the reasoning process. Methods like RLET (Liu et al., 2022) and SEER (Chen et al., 2024) use RL to direct reasoning and improve the transparency and interpretability of answers. Other studies, such as those by Hong et al. (2023) and Nathani et al. (2023), have explored Monte Carlo methods and multi-faceted feedback to enhance the reliability of multi-step reasoning. Despite these advancements, challenges remain in developing dynamic and adaptive reward functions that effectively assess the logical consistency of reasoning steps. Our research addresses these challenges by introducing a dynamic reward mechanism informed by natural logic, ensuring the reasoning process remains both logically consistent and aligned with the correct solution.

2.2 Natural Logic

Natural Logic (Van Benthem, 1988; Sanchez, 2016; Van Benthem, 1995; Nairn et al., 2006; MacCart-

Relation: Name	Example	Set Theoretic Definition
$\equiv$: Equivalence	mistake $\equiv$ error	$x = y$
$\sqsubseteq$: Forward Entailment	sunflower $\sqsubseteq$ flower	$x \subset y$
$\sqsupseteq$: Reverse Entailment	flower $\sqsupseteq$ sunflower	$x \supset y$
$\wedge$: Negation	usual $\wedge$ unusual	$x \cap y = \varnothing \wedge x \cup y = U$
$\Updownarrow$: Alternation	sunflower $\Updownarrow$ rose	$x \cap y = \varnothing \wedge x \cup y \neq U$
$\smile$: Cover	mammal $\smile$ nonhuman	$x \cap y \neq \varnothing \wedge x \cup y = U$
$\#$: Independence	banana $\#$ luck	All other cases

Table 1: A set $\mathcal{R}$ of seven logical relations proposed by MacCartney and Manning (MacCartney and Manning, 2009).

r	$\equiv$	$\sqsubseteq$	$\sqsupseteq$	$\smile$	$\Updownarrow$	$\wedge$	$\#$
$\delta(r)$	$\equiv$	$\sqsupseteq$	$\sqsubseteq$	$\Updownarrow$	$\smile$	$\wedge$	$\#$

Table 2: The projection function δ projects an input relation r into a different relation under downward monotonicity.

ney and Manning, 2009; Icard, 2012; Angeli et al., 2016; Shi et al., 2021) is a reasoning framework grounded in the syntax of natural language, offering a direct alternative to traditional formal systems like first-order logic. Tracing its origins back to Aristotle's syllogisms (Rose, 1965), natural logic eliminates the need for translating natural language into formal representations, enabling inferences to be drawn directly from linguistic surface structures. A major milestone in natural logic was the introduction of a set of seven logical relations by MacCartney and Manning (2009): $\mathcal{L} = \{\equiv, \sqsubseteq, \sqsupseteq, \wedge, \Updownarrow, \smile, \#\}$. These relations, detailed in Table 1, capture diverse logical interactions between linguistic expressions, thereby expanding the expressive capacity of natural logic. Another cornerstone of this framework is the principle of monotonicity (MacCartney and Manning, 2009; Valencia, 1991; Van Benthem et al., 1986; Icard III and Moss, 2014), which delineates how logical relationships are affected by linguistic context. In upward monotonic contexts, logical relationships are preserved, whereas in downward monotonic contexts, they may be inverted, as shown in Table 2.

The natural logic proof process (MacCartney and Manning, 2009; Angeli and Manning, 2014; Feng et al., 2020; Shi et al., 2025) can be understood as a structured sequence of steps. Initially, text spans are aligned between sentence pairs to identify corresponding elements. Subsequently, the logical relations between these aligned spans are then determined, followed by the application of contextual monotonicity to adjust these relationships. Ultimately, the aggregated logical relations are used to infer the overall relationship between the sentence pair.

3 Method

We formulate structured reasoning as a reinforcement learning (RL) task, aiming to learn an optimal reasoning policy. In RL, an agent makes sequential decisions to maximize the expected cumulative rewards through interactions with an environment. Here, the language model π serves as the agent's policy. The reasoning process begins from an initial state s_0. At each step t, the agent observes the current state s_t, receives a reward r_t, selects an action based on π, and transitions to the next state s_{t+1}. This process continues until a terminal state is reached. The reasoning process terminates when the model either generates an intermediate conclusion whose semantic similarity to the final hypothesis exceeds a predefined threshold (BLEURT(int_n,h) > 1) or reaches a predefined maximum reasoning depth (set to 20) without further valid inference steps. A trajectory $\xi = \{s_0, a_0, s_1, a_1, \ldots, s_n\}$ represents the sequence of states and actions throughout the interaction. The RL objective is to optimize the policy π to maximize the expected cumulative reward. At each reasoning step t, we define the state s_t as a tuple consisting of three components: the hypothesis hyp, the set of reasoning steps taken so far $\mathcal{T}_t$, and the set of available candidate sentences $\mathcal{U}_t$. Formally, we express the state as: $s_t = \{h, \mathcal{T}_t, \mathcal{U}_t\}$.

3.1 Reinforcement Component

Policy We represent our policy π with a generative model that can directly sample actions from the space $\mathcal{A}(s_t)$. This design enables the policy to explore a much wider range of promising actions (i.e., allowing arbitrary combinations of premises such as x_2 & x_4 & int_1) during reinforcement learning rather than being confined to paired premises (Hong et al., 2022; Liu et al., 2022; Yuan et al., 2024). To further accelerate the RL training process, we initially use the policy π to generate the top-k candidate actions:

$$a_t^i \sim \pi(a|s_t), \tag{1}$$

where the input is a linearized state s_t (i.e., the concatenation of hyp, $\mathcal{T}_t$, and $\mathcal{U}_t$), a_t^i denotes the i-th candidate actions at time step t. Subsequently, we renormalize the probabilities over these k actions to obtain a distribution and then sample from it to

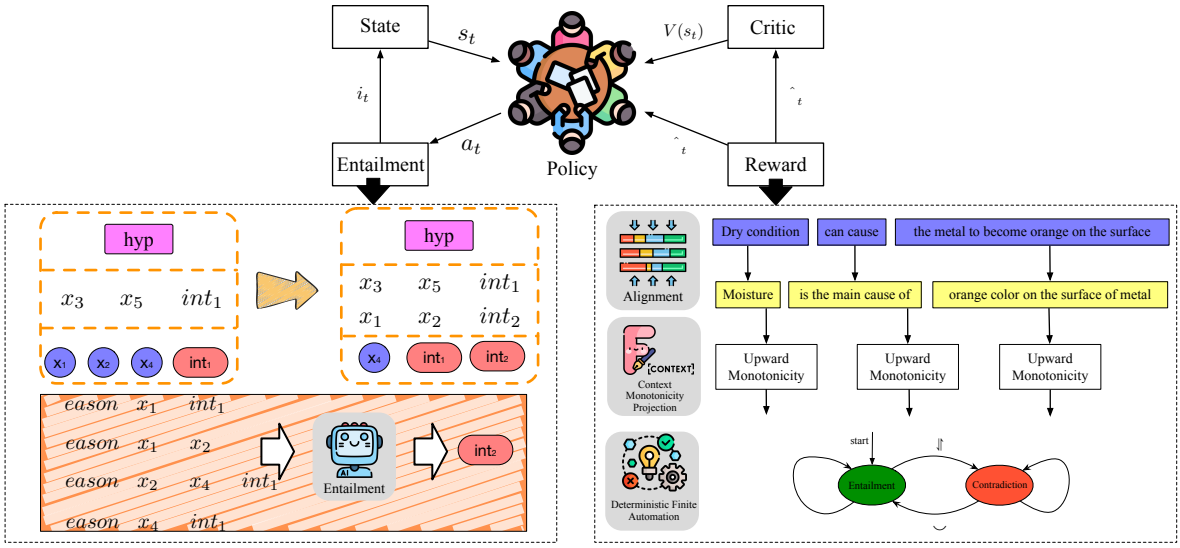

Figure 2: Overview of the proposed **NLDR** framework for entailment tree generation.

select the action a_t for the current reasoning step:

$$\pi'(a_t^i|s_t) = \frac{\pi(a_t^i|s_t)}{\sum_{i=1}^{k} \pi(a_t^i|s_t)}, \quad a_t \sim \pi'(a|s_t). \tag{2}$$

Reward To this end, we propose a natural logic-guided reward function that assigns different rewards for correct steps. We utilize Llama 3.1-8b-Instruct model (AI@Meta, 2024) to perform semantic chunking and the DeBERTa (He et al., 2023) model to predict NatOps. Here, NatOps refers to the set of natural logic operations, such as equivalence, forward entailment, negation, and alternation, as defined in MacCartney and Manning (2009). These operations are used to assess the logical relationships between aligned spans and form the foundation for computing dynamic rewards.

3.2 Natural Logic Guided Reward

During this phase, NLDR constructs a proof represented as a sequence of logical relations $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$. These relations are subsequently passed as input into a deterministic finite automaton (DFA), which generates an interpretable reasoning trace. For example, as illustrated in the bottom-right of Figure 2, a reasoning trace might take the form "$\mathrm{s}_{valid} \xrightarrow{\wedge} \mathrm{s}_{invalid} \xrightarrow{\equiv} \mathrm{s}_{invalid} \xrightarrow{\equiv} \mathrm{s}_{invalid}$". The details of the natural logic reasoning process are outlined below:

Chunking and Alignment The first step in the reasoning pipeline involves segmenting the hypothesis h_i and $fact_p$ into discrete chunks. Unlike traditional syntactic chunkers, which rely heavily on pre-defined linguistic rules, our approach leverages the capabilities of instruction-tuned large language models (LLMs) to perform semantic chunking. Specifically, the hypothesis and fact are fed into the LLM with a prompt instructing it to split the text into smaller, independently verifiable units. We perform the chunking by prompting an LLM to "*Split the text into smaller chunks that can be inferred by natural logic*". Each chunk is expected to encapsulate a single, atomic piece of information that can be mapped directly to evidence in the subsequent stages. We then use constrained decoding to ensure the desired output format.

After chunking, the next critical task is aligning the spans between the hypothesis and the generated fact. Span alignment is conceptualized as an optimal matching problem, where we aim to establish the best possible correspondence between spans from the hypothesis and those from the fact. To solve this problem, we utilize the Hungarian algorithm (Crouse, 2016), which efficiently finds a global optimal alignment. This step ensures accurate pairwise associations, a prerequisite for reliable logical relation prediction.

Logical Relation Prediction With aligned span pairs $\{(b_1, \tilde{b}_1), \ldots, (b_n, \tilde{b}_n)\}$, the next step is to predict the logical relation between each pair. We leverage a fine-tuned DeBERTa model to compute the probabilistic distribution p_n^r over the set of natural logical relations $\mathcal{L}$ for each aligned pair:

$$p_n^r = softmax(\text{DeBERTa}(b_n, \tilde{b}_n)), \tag{3}$$

where n denotes the number of aligned span pairs between the hypothesis and the intermediate. For each pair $(b_n, \tilde{b}_n)$, the DeBERTa model outputs a probability distribution over the set of natural logic relations.

These predicted logical relations are further refined through contextual monotonicity projection, which adjusts the initial predictions based on the monotonicity properties of the context. To implement this, we fine-tune a monotonicity classifier following the approach described by Rozanova et al. (Rozanova et al., 2021). The classifier predicts whether a given logical relation should be upward or downward within the specific contextual settings. This process yields a projected set of logical relations with updated probabilities:

$$\hat{p}_n^{k'} = p_n^k \mathbf{1}\left(\delta(k) = k'\right), \tag{4}$$

where $\delta(k)$ represents the monotonicity projection function, and k' refers to the projected label of the original relation k under contextual monotonicity. The sequence of projected logical relations is then used to guide the state transitions within the DFA, facilitating the final reasoning outcome.

Deterministic Finite Automaton The Deterministic Finite Automaton (DFA) acts as a lightweight yet powerful mechanism for interpreting the sequence of logical relations. It transitions through a predefined set of states based on the projected logical relations, ultimately arriving at a final classification. Unlike previous methods such as NaturalLI (Angeli and Manning, 2014), which employ three states (entailment, contradiction, and unknown), our approach simplifies the DFA to operate with only two states: entailment and contradiction. This design choice reduces complexity, enhances computational efficiency, and aligns better with the requirements of downstream answer selection tasks.

The probability of the DFA's final outcome is determined by aggregating the probabilities of the projected logical relations along the reasoning path. Specifically, the final outcome's probability is computed as the product of the individual relation probabilities:

$$\hat{y}_i = \prod_i^n \hat{p}_n^{k'} \tag{5}$$

3.3 Optimization

To address the challenges of training instability and sample inefficiency in reinforcement learning (Zhou et al., 2023; Roit et al., 2023), we employ the proximal policy optimization (PPO) algorithm (Schulman et al., 2017) to optimize the policy π, parameterized by θ. The policy loss function is formulated as follows:

$$\mathcal{L}_\pi = \mathbb{E}_t[\min\Big(r_t(\theta)\hat{A}_t, \text{clip}\big(r_t(\theta), 1-\epsilon, 1+\epsilon\big)\hat{A}_t\Big) + c\mathcal{H}(\pi'_\theta)], \tag{6}$$

where $r_t(\theta)$ is the importance sampling ratio given by:

$$r_t(\theta) = \frac{\pi'_\theta(a_t|s_t)}{\pi'_{\theta_{old}}(a_t|s_t)}. \tag{7}$$

Here, π' represents the probabilities normalized by Equation (2), and θ, θ_{old} denote the parameters of the new and old policies, respectively. The hyperparameter ϵ defines the clipping range, which stabilizes training by preventing excessively large policy updates. Additionally, c is the entropy coefficient, and $\mathcal{H}(\pi'_\theta)$ is the entropy bonus, which promotes sufficient exploration:

$$\mathcal{H}(\pi'_\theta) = -\pi'_\theta(a_t|s_t)\log\pi'_\theta(a_t|s_t). \tag{8}$$

Furthermore, $\hat{A}_t$ denotes the estimated advantage function for state s_t, computed as:

$$\hat{A}_t = r_t + \gamma V(s_{t+1}) - V(s_t), \tag{9}$$

where r_t is the reward calculated by Equation (5), $V(s_t)$ and $V(s_{t+1})$ are the state-value functions, and γ is the discount factor.

To ensure accurate return estimation and guide the policy towards effective updates, we train the critic by minimizing the squared advantage function:

$$\mathcal{L}_V = \mathbb{E}_t\left[(\hat{A}_t)^2\right]. \tag{10}$$

4 Experiments

4.1 Dataset

We used the EntailmentBank dataset (Dalvi et al., 2021), which contains 1,840 expert-annotated entailment trees. Each tree represents the reasoning process for deriving a hypothesis from a question-answer pair in the ARC dataset (Clark et al., 2018), with leaf nodes sourced from the WorldTreeV2 corpus (Xie et al., 2020). Table 3 summarizes the dataset statistics. Following Dalvi et al. (2021), we define three progressively challenging explanation

	Train	Dev	Test	All
Questions	1313	187	340	1840
Entailment Steps	4175	597	1109	5881

Table 3: Summary statistics for the EntailmentBank dataset splits.

tasks: (a) constructing a valid entailment tree using only relevant sentences, (b) generating the tree with both relevant and some irrelevant sentences, and (c) creating the tree using the complete corpus. The goal is to generate entailment trees that trace reasoning from facts to the final hypothesis.

4.2 Baselines

We compare our approach with several state-of-the-art methods in entailment tree generation. EntailmentWriter (Dalvi et al., 2021) generates entire trees and intermediate conclusions in a single pass using a sequence-to-sequence model, available in T5-11B and T5-Large versions. RLET (Liu et al., 2022) uses reinforcement learning for iterative, single-step reasoning to guide tree generation. MetGen (Hong et al., 2022) iteratively refines reasoning steps with multiple modules. NLProofs (Yang et al., 2022) uses an independent verifier to ensure logical soundness in proof step generation. FAME (Hong et al., 2023) employs a Monte Carlo planning approach for strategic action selection. SEER (Chen et al., 2024) uses reinforcement learning with fixed reward structures for guiding tree generation. METGEN+Rhetorical (Zhang et al., 2024) bridges reasoning types with rhetorical relations, and METGEN+LMPM (Yuan et al., 2024) incorporates external memory to store logical patterns for pre-training. SPEH (Song et al., 2024) introduces step feasibility perception and error handling for iterative tree generation. We use publicly available models and default settings for a fair comparison across methods.

4.3 Metrics

Entailment tree evaluation involves two steps. First, nodes from the predicted tree (T_{pred}) are aligned with the gold tree (T_{gold}) based on $sent_*$ labels and Jaccard similarity for intermediate nodes. In the second step, we evaluate the generated tree along three dimensions, following prior work (Dalvi et al., 2021; Hong et al., 2022; Yang et al., 2022):

- **Leaves:** This aspect examines the accuracy of the leaf facts utilized in the predicted tree. The F_1 score is calculated by comparing the predicted leaf facts with those in the gold tree. Additionally, we report the `AllCorrect` metric, which indicates a perfect match. Specifically, the `AllCorrect` score is set to 1 if the F_1 score is 1 (i.e., all predicted leaf facts match perfectly), and 0 otherwise.

- **Steps:** This dimension focuses on the structural correctness of the entailment steps. We measure accuracy using both the F_1 and `AllCorrect` scores. An entailment step is considered correct if the identifiers of its child nodes align precisely with those in the corresponding step of the gold tree.

- **Intermediates:** Correctness of intermediate nodes is evaluated using F_1 and `AllCorrect`. A predicted intermediate is considered correct if its semantic similarity to the corresponding gold intermediate, as measured by BLEURT-Large-512 (Sellam et al., 2020), exceeds a threshold of 0.28.

Finally, the `Overall AllCorrect` metric assigns a score of 1 if all leaves, steps, and intermediates in the predicted tree match perfectly with those in the gold tree. Any mismatch results in a score of 0.

4.4 Implementation Details

Previous studies (Chen et al., 2024; Hong et al., 2022) have consistently utilized T5-large (Raffel et al., 2020) as the base model. For a fair comparison, our policy is also built with a T5-large model with NVIDIA A100 GPUs. The pre-trained language models are derived from Huggingface Transformers (Wolf et al., 2020). For our experimental setup, we adopt the optimal hyperparameters from Chen et al. (2024).

5 Experimental Results and Analysis

5.1 Main Results

As shown in Table 4 and Table 5, our method, NLDR, outperforms all baseline models on the strictest metric, "`Overall AllCorrect`", across all three tasks.

The results in Table 4 clearly demonstrate the performance of NLDR on Task 1 and Task 2 of the EntailmentBank dataset, showing significant improvements over several baseline methods. In Task 1, NLDR achieves an absolute improvement

Method	Leaves		Steps		Intermediates		Overall
	F1	AllCorrect	F1	AllCorrect	F1	AllCorrect	AllCorrect
Task1 (no-distractor)							
IRGR (Ribeiro et al., 2022)	97.6	89.4	50.2	36.8	62.1	31.8	32.4
EntailmentWriter (Dalvi et al., 2021)	98.4	84.1	50.0	38.5	67.0	35.9	34.4
METGEN+Rhetorical (Zhang et al., 2024)	**100.0**	**100.0**	56.9	42.1	68.8	37.1	34.7
RLET (Liu et al., 2022)	**100.0**	**100.0**	54.6	40.7	66.9	36.3	34.8
METGEN (Hong et al., 2022)	**100.0**	**100.0**	57.7	41.9	70.8	39.2	36.5
SPEH (Song et al., 2024)	**100.0**	**100.0**	57.4	42.7	73.3	39.4	37.7
METGEN+LMPM (Yuan et al., 2024)	99.8	99.4	57.8	43.8	72.8	42.8	38.5
NLProofS (Yang et al., 2022)	97.8	90.1	55.6	42.3	72.4	40.6	38.9
FLD (Morishita et al., 2023)	99.0	92.7	55.5	42.2	73.4	41.3	39.2
FRVA (Fan et al., 2024)	98.2	94.0	57.8	44.4	**73.5**	42.4	40.3
SEER (Chen et al., 2024)	**100.0**	**100.0**	67.6	52.6	70.3	42.6	40.6
NLDR (Ours)	**100.0**	**100.0**	**71.7**	**55.3**	73.3	**46.2**	**42.6**
Task2 (distractor)							
IRGR (Ribeiro et al., 2022)	69.9	23.8	30.5	22.4	47.7	26.5	21.8
EntailmentWriter (Dalvi et al., 2021)	83.2	35.0	39.5	24.7	62.2	28.2	23.2
RLET (Liu et al., 2022)	81.0	39.0	38.5	28.4	56.3	28.6	25.7
METGEN (Hong et al., 2022)	82.7	46.1	41.3	29.6	61.4	32.4	27.7
METGEN+Rhetorical (Zhang et al., 2024)	83.3	49.7	42.1	30.3	61.0	31.8	28.2
METGEN+LMPM (Yuan et al., 2024)	81.1	47.1	42.6	31.4	61.7	34.3	29.4
FLD (Morishita et al., 2023)	88.4	53.6	45.6	33.8	67.9	36.1	32.6
NLProofs (Yang et al., 2022)	90.3	58.8	47.2	34.4	70.2	37.8	33.3
FRVA (Fan et al., 2024)	**91.3**	**60.5**	48.0	35.8	**71.1**	**39.1**	34.4
SEER (Chen et al., 2024)	86.4	53.5	56.8	39.7	66.3	38.3	34.7
NLDR (Ours)	83.6	51.5	**60.2**	**41.2**	66.9	38.2	**36.5**

Table 4: Automatic evaluation results of Task 1 and Task 2 on the EntailmentBank test split (%). All baseline results come from published papers.

of 2.0% over the strongest baseline, SEER. SEER's performance is often limited by its rigid reward assignments, which do not adapt to the complexities of nuanced logical inferences. In contrast, NLDR uses a dynamic reward mechanism based on natural logic reward, which allows the model to adapt during the inference process. This flexibility leads to more precise and context-sensitive entailment tree generation, enhancing the model's ability to handle complex logical dependencies.

Similarly, in Task 2, NLDR demonstrates a 1.8% improvement over SEER. The dynamic reward structure of NLDR helps it more effectively manage abstract reasoning, especially in situations where SEER's static rewards may fail to capture the subtleties of the inference process. For example, in cases involving more ambiguous or indirect entailments, NLDR adjusts its reward function to guide the model toward more accurate conclusions, whereas SEER struggles with such scenarios. This highlights NLDR's robustness across diverse reasoning tasks and its ability to handle nuanced inference challenges that other methods may overlook.

When compared to the RL-based method RLET, NLDR shows significant improvements, outperforming RLET by 7.8%, 10.8%, and 6.3% on Tasks 1, 2, and 3, respectively. Additionally, NLDR surpasses SEER by 2.0%, 1.8%, and 0.3% across the same tasks. The key advantage of NLDR lies in its dynamic reward function, which is grounded in natural logic and better aligns with the logical dependencies inherent in multi-step reasoning. This ensures that each step in the entailment tree generation process remains logically consistent and that the reward adapts to the quality of reasoning at each stage. On the other hand, RLET relies on a static reward structure, which can fail to account for the complexities of reasoning across the entire tree, leading to less accurate predictions. Furthermore, while SEER also focuses on structured reasoning, its static reward function may not capture the evolving nature of multi-step logical inferences, making it less effective than NLDR in handling tasks that require deep and flexible reasoning.

In Task 3, as shown in Table 5, NLDR also outperforms advanced models such as GPT-4 with Chain-of-Thought (CoT) and Tree of Thoughts (ToT). Task 3 requires the model to maintain logical consistency across multiple reasoning steps, which can be particularly challenging for models

Method	Leaves		Steps		Intermediates		Overall
	F1	AllCorrect	F1	AllCorrect	F1	AllCorrect	AllCorrect
EntailmentWriter (Dalvi et al., 2021)	30.9	1.2	4.4	1.2	28.8	5.6	1.2
NLProofs (Yang et al., 2022)	43.2	8.2	11.2	6.9	42.9	17.3	6.9
FRVA (Fan et al., 2024)	44.0	8.9	11.6	7.5	43.2	17.9	7.5
IRGR (Ribeiro et al., 2022)	46.6	10.0	11.3	8.2	38.7	**20.9**	8.2
FLD (Morishita et al., 2023)	43.6	9.7	12.1	8.3	43.0	20.1	8.3
METGEN+Rhetorical (Zhang et al., 2024)	36.8	8.5	10.6	8.5	37.7	20.3	8.5
METGEN+T5 (Hong et al., 2022)	34.8	8.7	9.8	8.6	36.6	20.4	8.6
RLET (Liu et al., 2022)	38.3	9.1	11.5	7.1	34.2	12.1	6.9
METGEN+LMPM (Yuan et al., 2024)	35.3	9.2	10.3	9.2	37.8	20.3	9.4
GPT4-ReAct (Yao et al., 2023b)	45.8	12.9	14.1	10.5	43.5	21.5	10.5
GPT4-CoT (Wei et al., 2022)	44.1	12.1	15.4	10.8	43.1	20.6	10.8
GPT4-ToT (Yao et al., 2023a)	43.3	12.0	15.8	11.0	43.9	20.0	11.0
FAME (Hong et al., 2023)	43.4	13.8	16.6	12.4	40.6	19.9	11.9
SEER (Chen et al., 2024)	47.1	13.8	17.4	12.9	45.1	18.8	12.9
NLDR (Ours)	**48.1**	**14.4**	**17.5**	**13.2**	**47.8**	20.0	**13.2**

Table 5: Automatic evaluation results of Task 3 on the EntailmentBank test split (%). All baseline results come from published papers.

Method	Steps	Intermediates	Overall
NLDR (Ours)	**55.3**	**46.2**	**42.6**
w/o dynamic reward	52.6	42.6	40.6
w/o logic reward	52.1	43.2	41.5

Table 6: Ablation results of Task 1 on EntailmentBank test set (%).

like GPT-4 with CoT. While GPT-4 with CoT can perform step-by-step reasoning, it struggles when faced with longer or more deeply nested reasoning chains, often losing track of earlier steps or failing to maintain logical consistency. Similarly, ToT, although useful for structured reasoning, faces limitations in dynamically adapting to the specific requirements of each reasoning step in complex tasks. In contrast, NLDR's dynamic reward mechanism allows it to adjust its approach at each step, effectively managing the complexity of multi-step inferences. By continuously optimizing its reward based on the logical structure of the entailment, NLDR achieves a 2.2% improvement over both GPT-4 with CoT and ToT in Task 3. For tasks that require deep logical analysis or multi-step inference, NLDR maintains higher accuracy by adapting to the evolving context, whereas GPT-4 with CoT may struggle with complex reasoning dependencies. ToT, while structured, lacks the flexibility to adjust dynamically to the changing needs of each inference step, which is precisely where NLDR excels.

5.2 Ablation Study

In this section, we conduct an ablation study to evaluate the impact of different components of the NLDR method on its performance. We systematically analyze the effect of removing the dynamic reward mechanism and the natural logic-based reward function. As shown in Table 6, we tested two key variations.

First, the **w/o dynamic reward** setup disables the dynamic reward mechanism and replaces it with a fixed reward value (1 or -1) for correct or incorrect steps. The results show a significant drop in performance, indicating that the dynamic nature of the reward mechanism is crucial for capturing the complex dependencies between intermediate nodes and the root node in the entailment tree. Without the dynamic reward, the model struggles to adjust its reasoning process appropriately, leading to a decrease in reasoning accuracy.

The second variation, **w/o logic reward**, removes the natural logic-based reward function and substitutes it with a reward signal derived from a neural network-based model, specifically the DeBERTa model's entailment score. While the DeBERTa-based reward still offers some guidance, it lacks the fine-grained logical reasoning that natural logic provides. As a result, the model's ability to generate high-quality entailment trees is compromised, and its performance is lower compared to the full NLDR model, which incorporates both natural logic and dynamic rewards.

5.3 Reward Function Comparison Experiment

To further evaluate the performance of different dynamic reward functions, we compared NLDR with several pre-trained language models used to compute dynamic reward scores: DeBERTaV3 (He

Method	Steps	Intermediates	Overall
NLDR (Ours)	**55.3**	**46.2**	**42.6**
DeBERTaV3 (He et al., 2023)	52.1	43.2	41.5
Llama 3.1-8B-Instruct (AI@Meta, 2024)	54.4	42.9	41.2
DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025)	53.8	45.3	41.8

Table 7: Reward function comparison experiment of Task 1 on EntailmentBank test set (%).

et al., 2023), Llama 3.1-8B-Instruct (AI@Meta, 2024), and DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025). The results, shown in Table 7, clearly highlight the superiority of the NLDR method, which utilizes natural logic-based rewards, over the neural network-based approaches. Unlike the neural network-based methods, the natural logic approach provides a more precise and logically grounded reward signal, leading to significantly improved reasoning accuracy. These results demonstrate the significant advantage of using natural logic as a reward function for generating high-quality entailment trees in complex reasoning tasks.

6 Conclusion

In this paper, we introduced the NLDR method, which leverages natural logic to compute dynamic entailment scores between intermediate nodes and the root node in an entailment tree, offering a more nuanced and adaptable reward function for reinforcement learning compared to traditional fixed-value rewards. Our experimental results demonstrate that NLDR not only enhances the quality of entailment tree generation but also exhibits significant improvements in logical consistency, reasoning ability, and overall model accuracy. Future work could focus on further optimizing the reward function design, improving model interpretability and scalability, and exploring the application of this approach to a broader range of natural language inference scenarios, ultimately advancing the development of intelligent reasoning systems.

Acknowledgements

We thank the anonymous reviewers for their constructive comments, and gratefully acknowledge the support of the New Generation Artificial Intelligence of China (2024YFE0203700), National Natural Science Foundation of China under Grants U22B2059 and 62176079.

Limitations

While the approach presented in this paper offers a novel integration of natural logic to dynamically generate entailment tree rewards, there are a few limitations. First, the model's reliance on T5 as the backbone architecture may limit its generalizability to other pre-trained models. Future work could explore how the proposed method performs with alternative architectures. Additionally, while the dynamic reward function more accurately reflects logical relationships, it may still be influenced by inherent biases in the underlying language model, which could affect the consistency of the entailment scores. Finally, the current evaluation is conducted on a relatively limited set of datasets, which may not fully capture the complexity of entailment tasks across different domains. Expanding the evaluation to a wider range of datasets would provide a more comprehensive assessment of the model's robustness.

References

AI@Meta. 2024. Llama 3 model card.

Gabor Angeli, Neha Nayak Kennard, and Christopher D Manning. 2016. Combining natural logic and shallow reasoning for question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 442–452.

Gabor Angeli and Christopher D Manning. 2014. Naturalli: Natural logic inference for common sense reasoning. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 534–545.

Kaj Bostrom, Zayne Sprague, Swarat Chaudhuri, and Greg Durrett. 2022. Natural language deduction through search over statement compositions. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4871–4883.

Guoxin Chen, Kexin Tang, Chao Yang, Fuying Ye, Yu Qiao, and Yiming Qian. 2024. SEER: facilitating structured reasoning and explanation via reinforcement learning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 5901–5921. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

David F Crouse. 2016. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. 2021. Explaining answers with entailment trees. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7358–7370.

Yue Fan, Hu Zhang, Ru Li, Yujie Wang, Hongye Tan, and Jiye Liang. 2024. Frva: Fact-retrieval and verification augmented entailment tree generation for explainable question answering. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 9111–9128.

Yufei Feng, Quan Liu, Michael Greenspan, Xiaodan Zhu, et al. 2020. Exploring end-to-end differentiable natural logic modeling. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1172–1185.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. In *The Eleventh International Conference on Learning Representations, ICLR*.

Ruixin Hong, Hongming Zhang, Xintong Yu, and Changshui Zhang. 2022. Metgen: A module-based entailment tree generation framework for answer explanation. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1887–1905.

Ruixin Hong, Hongming Zhang, Hong Zhao, Dong Yu, and Changshui Zhang. 2023. Faithful question answering with monte-carlo planning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3944–3965.

Thomas F Icard. 2012. Inclusion and exclusion in natural language. *Studia Logica*, 100:705–725.

Thomas F Icard III and Lawrence S Moss. 2014. Recent progress on monotonicity. *Linguistic Issues in Language Technology*, 9:167–194.

George Lakoff. 1970. Linguistics and natural logic. *Synthese*, 22(1):151–271.

Tengxiao Liu, Qipeng Guo, Xiangkun Hu, Yue Zhang, Xipeng Qiu, and Zheng Zhang. 2022. Rlet: A reinforcement learning based approach for explainable qa with entailment trees. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7177–7189.

Bill MacCartney and Christopher D Manning. 2009. An extended model of natural logic. In *Proceedings of the eight international conference on computational semantics*, pages 140–156.

Terufumi Morishita, Gaku Morio, Atsuki Yamaguchi, and Yasuhiro Sogawa. 2023. Learning deductive reasoning from synthetic corpus based on formal logic. In *International Conference on Machine Learning*, pages 25254–25274. PMLR.

Rowan Nairn, Cleo Condoravdi, and Lauri Karttunen. 2006. Computing relative polarity for textual inference. In *Proceedings of the fifth international workshop on inference in computational semantics (icos-5)*.

Deepak Nathani, David Wang, Liangming Pan, and William Wang. 2023. Maf: Multi-aspect feedback for improving reasoning in large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6591–6616.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Danilo Neves Ribeiro, Shen Wang, Xiaofei Ma, Rui Dong, Xiaokai Wei, Henghui Zhu, Xinchi Chen, Peng Xu, Zhiheng Huang, Andrew Arnold, et al. 2022. Entailment tree explanations via iterative retrieval-generation reasoner. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 465–475.

Paul Roit, Johan Ferret, Lior Shani, Roee Aharoni, Geoffrey Cideron, Robert Dadashi, Matthieu Geist, Sertan Girgin, Leonard Hussenot, Orgad Keller, et al. 2023. Factually consistent summarization via reinforcement learning with textual entailment feedback. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6252–6272.

Lynn E Rose. 1965. Aristotle's syllogistic and the fourth figure. *Mind*, pages 382–389.

Julia Rozanova, Deborah Ferreira, Mokanarangan Thayaparan, Marco Valentino, and André Freitas. 2021. Supporting context monotonicity abstractions in neural nli models. In *Proceedings of the 1st and 2nd Workshops on Natural Logic Meets Machine Learning (NALOMA)*, pages 41–50.

Victor Sanchez. 2016. *Studies on natural logic and categorial grammar*. Ph.D. thesis, University of Amsterdam.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. Bleurt: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892.

Jihao Shi, Xiao Ding, Li Du, Ting Liu, and Bing Qin. 2021. Neural natural logic inference for interpretable question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3673–3684.

Jihao Shi, Xiao Ding, Siu Cheung Hui, Yuxiong Yan, Hengwei Zhao, Ting Liu, and Bing Qin. 2025. Final: Combining first-order logic with natural logic for question answering. *IEEE Transactions on Knowledge and Data Engineering*.

Junyue Song, Xin Wu, and Yi Cai. 2024. Step feasibility-aware and error-correctable entailment tree generation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15296–15308.

Zayne Sprague, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. 2022. Natural language deduction with incomplete information. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8230–8258.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2022. Entailer: Answering questions with faithful and truthful chains of reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2078–2093.

Víctor Manuel Sánchez Valencia. 1991. *Studies on natural logic and categorial grammar*. Universiteit van Amsterdam.

Johan Van Benthem. 1988. The semantics of variety in categorial grammar. In *Categorial grammar*, page 37. John Benjamins.

Johan Van Benthem. 1995. *Language in Action: categories, lambdas and dynamic logic*. MIT Press.

Johan Van Benthem et al. 1986. *Essays in logical semantics*, volume 29. Springer.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Nathaniel Weir and Benjamin Van Durme. 2022. Dynamic generation of grounded logical explanations in a neuro-symbolic expert system. *arXiv preprint arXiv:2209.07662*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Zhengnan Xie, Sebastian Thiem, Jaycie Martin, Elizabeth Wainwright, Steven Marmorstein, and Peter Jansen. 2020. Worldtree v2: A corpus of science-domain structured explanations and inference patterns supporting multi-hop inference. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5456–5473.

Kaiyu Yang, Jia Deng, and Danqi Chen. 2022. Generating natural language proofs with verifier-guided search. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 89–105.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023b. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Li Yuan, Yi Cai, Haopeng Ren, and Jiexin Wang. 2024. A logical pattern memory pre-trained model for entailment tree generation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 759–772.

Longyin Zhang, Bowei Zou, and Aiti Aw. 2024. Empowering tree-structured entailment reasoning: Rhetorical perception and llm-driven interpretability. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 5783–5793.

Jinfeng Zhou, Zhuang Chen, Bo Wang, and Minlie Huang. 2023. Facilitating multi-turn emotional support conversation with positive emotion elicitation: A reinforcement learning approach. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.