Eng Int Syst (2026) 5: 683–696
© 2026 CRL Publishing Ltd

Engineering
Intelligent Systems

SymLoop: A Symbolic Constraint–Integrated Framework for Consistency Preservation and Reasoning Control in English Text Generation

Xiaoyi Liu*

Wuhan Technology and Business University, Wuhan, 430065, China

Large language models (LLMs) have demonstrated remarkable performance in English text generation. However, their prevailing next-token prediction paradigm struggles to internalize deep causal reasoning rules, often resulting in outputs that are formally well-structured yet logically fragmented. Existing approaches suffer from notable limitations: declarative prompting methods lack deep integration with logical rules; external solver-based approaches rely on sequential pipelines that impede real-time correction; and constrained decoding methods disrupt linguistic fluency due to discrete interventions. To address these challenges, we propose SymLoop, a symbolic constraint-integrated framework for consistency preservation and reasoning control in English text generation. SymLoop establishes a three-stage neuro-symbolic closed loop of extraction - injection - verification. Specifically, we introduce a Context-Adaptive Symbolic Constraint Extraction (CASCE) algorithm that dynamically captures implicit logical entailments; a Bayesian Symbolic Constraint Injection (BSCI) algorithm that softens discrete rules into probabilistic priors within the latent space; and a Hierarchical Symbolic Verification Feedback Loop (HSVFL) that enables end-to-end logical correction through reinforcement learning. Experimental results show that SymLoop achieves a logical consistency accuracy of 0.94 and a BERTScore of 0.89 on ProofWriter, and outperforms the strongest baseline by 12% on EntailmentBank. The reasoning latency is 48 ms/token, yielding a 2.6× speedup over classical constrained decoding methods. This work introduces a new paradigm for integrating probabilistic modeling with logical reasoning in neural language models, providing technical support for applications such as academic writing and the drafting of legal documents.

Keywords: large language models; neuro-symbolic integration; logical consistency; English text generation; variational inference; SymLoop

1. INTRODUCTION

With the rapid development of large language models, their capabilities in English text generation have advanced substantially. However, as a hypotactic language, English relies heavily on complex clause nesting and precise discourse markers to ensure logical rigor [1]. While the next-token prediction paradigm effectively reproduces surface-level fluency, it fails to internalize deep causal reasoning rules, leading to generations that are syntactically acceptable yet logically incoherent [2]. This phenomenon reveals a fundamental gap between continuous neural representations and the discrete rules of symbolic logic [3]. Consequently, the enforcement of symbolic reasoning constraints while preserving idiomatic English expression has emerged as a key scientific challenge for trustworthy artificial intelligence.

Existing studies can be broadly categorized as having three approaches: prompting-based methods, external solver-

*Email: xiaoyiliuedu@163.com

based methods, and constrained decoding [4]. Although declarative prompting approaches (e.g., SATLM) enhance constraint awareness through structured prompts, they lack deep integration with logical rules [5]. External solver–based methods (e.g., LINC, Logic-LM) adopt a "translate–solve" paradigm that enables post hoc error detection; however, their effectiveness depends on intermediate translation accuracy, and their sequential pipelines hinder real-time correction. Constrained decoding approaches (e.g., NeuroLogic) impose hard predicate-logic constraints during generation, but discrete interventions often compromise the natural fluency of English text [6]. Currently, there are no end-to-end mechanisms capable of dynamically parsing complex contextual entailments in English and injecting verifiable symbolic constraints into the decoding process in real time [7].

To overcome these limitations, we propose SymLoop, a symbolic constraint loop framework for consistency preservation and reasoning control in English generation. SymLoop constructs a three-stage neuro-symbolic collaborative loop consisting of extraction, injection, and verification. Specifically, SymLoop introduces a Context-Adaptive Symbolic Constraint Extraction (CASCE) algorithm that employs graph neural networks to probe attention responses to discourse connectives and extract entailment structures in real time. We further propose a Bayesian Symbolic Constraint Injection (BSCI) algorithm that leverages variational inference to transform deterministic logic into probabilistic distributions in latent space, achieving a deep integration of linguistic fluency and logical rigor. Finally, we design a Hierarchical Symbolic Verification Feedback Loop (HSVFL) that incorporates the Z3 solver to perform dual validation of facts and logic, and applies reinforcement learning to selectively penalize generations that are formally correct but propositionally inconsistent.

At the theoretical level, this work offers a variational inference-based paradigm to address the long-standing dichotomy between probabilistic modeling and logical reasoning. Methodologically, SymLoop's dynamic constraint mechanism overcomes the limitations of static rules, substantially improving robustness in high logical-density tasks. From an application perspective, this study lays a foundation for building trustworthy artificial intelligence systems.

2. RELATED WORK

2.1 Application Scenarios and Evaluation Frameworks

The application of large language models to English text generation has expanded to professional domains such as academic writing, legal documentation, and financial analysis. These scenarios impose stringent requirements in regard to logical consistency [8]. As a hypotactic language, English expresses logical relations primarily through explicit discourse connectives (e.g., therefore, however) and complex subordinate structures, making logical discontinuities evident via syntactic markers. In contrast, paratactic languages such as Chinese rely more heavily on semantic inference and contextual cues to convey logical relationships [9]. As a result, logical errors in English texts are more easily perceived, placing higher demands on the logic-control capabilities of generation systems.

Mainstream evaluation datasets such as LogiQA and ReClor focus on short-text reasoning, while SummEval and FRANK focus on detecting contradictions in sentences, with dataset sizes ranging from tens of thousands to hundreds of thousands of instances [10]. However, these datasets fail to adequately capture long-range logical dependencies arising from cross-sentence causal chains and nested conditional clauses in English discourse [11]. For example, complex sentence constructions involving multiple layers of conditions, and exceptions are underrepresented in existing annotations. In addition, the role of English-specific grammatical features, such as tense systems and the subjunctive mood, in logical reasoning lacks systematic evaluation benchmarks [12].

Regarding evaluation metrics, n-gram overlap-based measures such as BLEU and ROUGE primarily assess lexical similarity and cannot adequately detect logical contradictions caused by misuse of discourse markers or tense inconsistency within conditional clauses. Although large-model-based evaluation methods enable semantic judgment, they remain limited in quantitatively verifying the logical completeness of complex sentence structures [13]. Human evaluation can identify deep logical issues, but this is costly and lacks standardized criteria, particularly for assessing cross-paragraph logical coherence where subjectivity is pronounced.

2.2 Overview of Mainstream Technical Approaches

The first category of methods guides reasoning through prompt engineering. Chain-of-Thought (CoT) techniques require models to explicitly output intermediate reasoning steps and have significantly improved performance in tasks such as mathematical reasoning [14]. However, in English text generation, polysemy and flexible coreference may obscure logical fallacies, allowing models to produce reasoning chains that are formally valid yet causally unsound. Moreover, in long-form generation, long-distance dependencies in complex English sentences can lead to error accumulation, while prompt engineering relies heavily on manually designed templates and cannot adapt to different domains and writing styles.

The second category introduces external knowledge or verification mechanisms. Retrieval-Augmented Generation (RAG) reduces factual hallucination by dynamically querying knowledge bases, and performs well in closed-domain tasks [15]. Nevertheless, the mapping of complex English sentences containing multiple conditions into structured queries is inherently challenging, requiring precise handling of conditional clauses, tense entailments, and causal directionality. The sequential, post hoc verification pipeline also makes it difficult to correct logical errors in already-generated sentences [16]. Furthermore, external knowledge bases offer limited coverage and provide insufficient support for implicit, multi-step logical reasoning.

The third category attempts to inject logical constraints directly into the latent representation space of models by modifying loss functions or introducing decoding biases [17]. This approach does not require external modules and is easily integrated into existing architectures. However, when mapping discrete symbolic logic into continuous representations, it struggles to precisely capture the deterministic semantics of English logical operators, such as differences in negation scope and quantifier binding, and generally has limited interpretability [18].

2.3 Comparison with Most Related Work

Training frameworks based on logical consistency losses incorporate external fact bases during pretraining to force generated sentences to remain consistent with knowledge triples, effectively reducing hallucination and demonstrating the value of explicit logical supervision for long-range consistency [19]. However, such methods depend on static knowledge bases and are ill-suited for handling dynamic contextual logic. In contrast, SymLoop's CASCE algorithm extracts entailment structures of English discourse markers in real time from attention mechanisms and integrates HSVFL to enable dynamic intervention during the reasoning process.

Symbolic rule vectorization methods embed first-order logic formulas as differentiable tensors and perform well in structured reasoning tasks; however, they face limitations when handling the syntactic diversity of English expressions [20]. The BSCI algorithm proposed in SymLoop adopts a Bayesian variational framework to model constraints as vocabulary-level probabilistic priors, balancing the rigor of symbolic semantics with the flexibility of natural language expression, and is therefore more suitable for open-domain English generation.

Verifiable reasoning frameworks (e.g., LINC, Logic-LM) couple generators with formal verifiers to achieve high-confidence validation in tasks such as mathematics and code generation [21]. However, their reliance on sequential post-generation rejection sampling makes it difficult to correct logical errors in already-generated sequences, and their natural language parsing depends on manually crafted templates. Conversely, SymLoop's HSVFL performs real-time verification during sentence-by-sentence generation and employs PPO-based policy gradients to transform verification outcomes into training signals for end-to-end logical correction. This mechanism penalizes generations that are formally correct but propositionally inconsistent, while constraint extraction is adaptively handled by CASCE.

2.4 Research Gaps and Positioning of This Work

In summary, existing methods for improving logical consistency in large language models remain insufficient in several respects. First, models exhibit limited awareness of dynamically evolving contextual logic during generation, making it difficult to handle nested conditionals and subjunctive mood shifts. Second, the integration of symbolic logic with neural networks remains incomplete, often resulting in precision loss or constraints on end-to-end training. Third, there is a lack of dedicated research on logic constraint verification and correction mechanisms tailored to English-specific linguistic properties, such as hypotactic structures, long-distance dependencies, and tense systems.

To address these challenges, this paper proposes the SymLoop framework, which comprises three core algorithms: CASCE, BSCI, and a HSVFL. By enhancing English generation controllability in regard to dynamic parsing, probabilistic injection, and closed-loop verification, SymLoop aims to provide a new technical pathway and methodological approach for future research.

3. METHODOLOGY

3.1 Problem Formulation

This study aims to construct a neuro-symbolic hybrid system that preserves logical consistency during English text generation. Formally, given an input context sequence $x = (x_1, \ldots, x_m)$, where $x_i \in \mathcal{V}$ denotes a token from the English vocabulary, the system is required to generate a target sequence $y = (y_1, \ldots, y_n)$, such that the output is not only linguistically fluent but also satisfies verifiable logical consistency constraints.

The input x may consist of a problem statement, a partial document, or dialogue history. The output y must simultaneously satisfy linguistic constraints $\mathcal{C}_{lang}$ (including grammatical well-formedness and semantic coherence) and logical constraints $\mathcal{C}_{logic}$, which require propositional relations to conform to formal logic rules, such as causal entailment transitivity and negation consistency. For logical relations explicitly marked in English through discourse connectives (e.g., "therefore", "unless"), the system must ensure semantic-syntactic alignment.

We assume that the training data are drawn from a joint distribution $P(x, y)$, where each sample pair $(x_i' y_i)$ satisfies logical consistency requirements. The dataset is divided into a training set $\mathcal{D}_{train} = \{(x_i, y_i)\}_{i=1}^{N}$ and a validation set $\mathcal{D}_{val}$. Ideally, the model learns the conditional distribution $P_\theta(y|x)$, with parameters θ obtained by minimizing the expected risk:

$$\mathcal{R}(\theta) = \mathbb{E}_{(x,y)\ P}\left[\mathcal{L}(y, \widehat{y}_\theta) + \lambda \cdot \mathcal{V}(\widehat{y}_\theta, \mathcal{C}_{logic})\right], \quad (1)$$

where $\mathcal{L}$ denotes the language generation loss (e.g., cross-entropy), and $\mathcal{V}$ is a logical violation penalty function that quantifies deviations between the generated sequence and the logical constraint set.

Traditional neural language models struggle to directly optimize objectives involving discrete logical constraints. This work is motivated by the need to address this problem: how to inject the deterministic constraints of symbolic logic into the generation process while maintaining end-to-end differentiability, and how to ensure logical completeness when handling complex English syntactic structures, such as nested conditional clauses and the subjunctive mood.

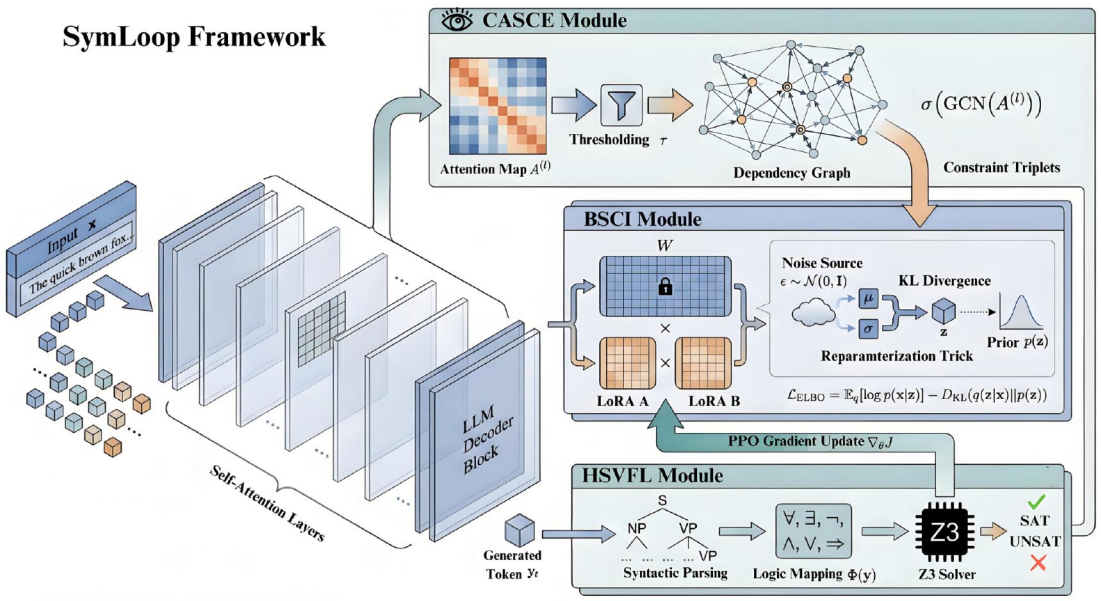


Figure 1 Overview of the proposed SymLoop framework.

3.2 SymLoop Framework Design

As illustrated in Figure 1, the SymLoop framework adopts an end-to-end neuro-symbolic collaborative architecture to mitigate logical drift and factual hallucination in long-form English generation. After the input context x is encoded by a large language model backbone, it is sequentially processed by three collaborative modules. The CASCE module dynamically parses entailment structures in English discourse from attention mechanisms; the BSCI module transforms discrete logical rules into probabilistic priors in latent space; and the HSVFL module performs posterior verification using the Z3 solver and propagates corrective signals. The system ultimately generates a target sequence y that satisfies both English syntactic norms and logical completeness. Through dynamic constraint injection and feedback optimization, SymLoop forms a closed-loop architecture consisting of extraction, injection, and verification.

During the initial generation stage, the CASCE module operates as a perceptual unit. Rather than relying on static rule libraries, it monitors the model's attention responses to English discourse markers (e.g., therefore, unless) and syntactic dependency relations, dynamically extracting logical entailment structures from hidden representations to construct an instantaneous constraint set. Subsequently, the BSCI module softens these discrete constraints and injects them into the generation process. Based on variational inference, BSCI converts deterministic logical rules into prior distributions in latent space and guides the decoding process toward logically consistent manifolds by optimizing a KL-divergence term, while preserving the flexibility of English expression.

Finally, the HSVFL module functions as a correction and optimization unit. It establishes a dual-layer verification mechanism that integrates factual validation with logical reasoning based on the Z3 solver. Generated text is translated into formal logical expressions and subjected to satisfiability checking. Verification outcomes are mapped into reinforcement learning reward signals, which are used to update model parameters via policy gradient methods. This process penalizes those generation paths exhibiting deep logical contradictions and promotes the coordinated integration of continuous neural representations with symbolic logic rules.

3.3 Module Details

3.3.1 CASCE

Motivation

Static rule templates struggle to accommodate semantic dynamics caused by polysemous words in English, such as "since", which can express both temporal and causal relations depending on context. The CASCE module is designed to address this context-dependency challenge by perceiving attention flows in real time and accurately extracting context-specific logical entailment structures from unstructured representations.

Principle

The core mechanism of CASCE is attention-based causal probing. Self-attention weights in Transformer architectures implicitly encode logical dependencies among tokens. Instead of relying on simple numerical thresholding, this module employs a graph neural network (GNN) as a probe, treating the attention matrix as a weighted graph. Graph convolution is used to aggregate neighborhood information, enabling the identification of token pairs with strong causal associations.

Implementation

As illustrated in Figure 2, the attention matrix from the l-th layer, denoted as $A^{(l)}$, is fed into a lightweight graph convolutional network (GCN). The resulting logical adjacency matrix is filtered via thresholding to construct a dynamic constraint set $\mathcal{C}_{dynamic}$:

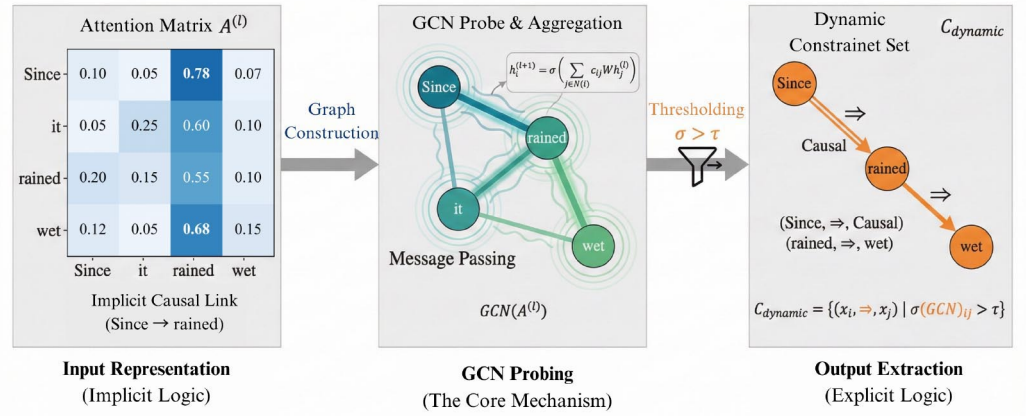


Figure 2 The hierarchical processing flow of the CASCE module.

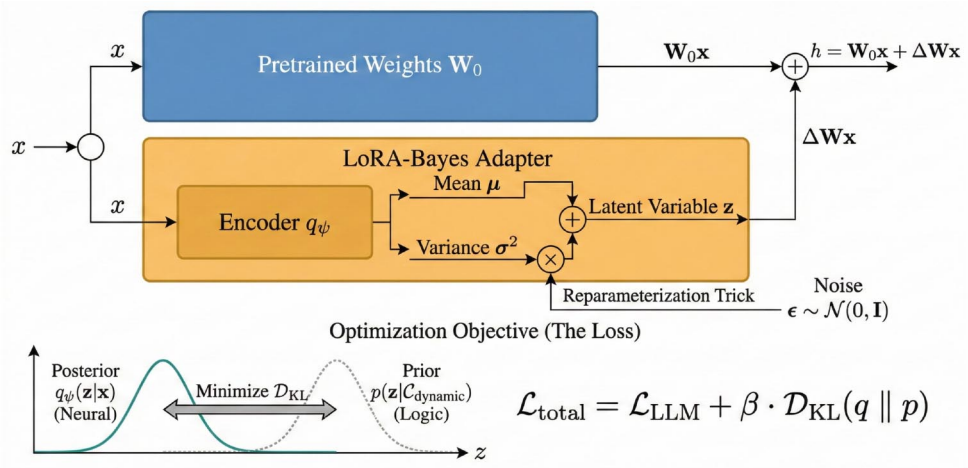


Figure 3 Illustration of the Bayesian Symbolic Constraint Injection (BSCI) algorithm.

$$\mathcal{C}_{dynamic} = \{(x_i, \Rightarrow, x_j) | \sigma(GCN(A^{(l)}))_{ij} > \tau\}. \quad (2)$$

This step maps continuous neural activation patterns into explicit symbolic logical triplets.

3.3.2 BSCI

Motivation

Directly imposing discrete hard constraints disrupts gradient propagation in neural networks and often leads to the generation of rigid or unnatural text. The BSCI module aims to resolve the conflict between symbolic logic and continuous optimization objectives by introducing probabilistic soft constraints that preserve logical rigor while maintaining fluency in English expression.

Principle

BSCI is grounded in variational inference. We model the ideal logically consistent generation state as a latent variable z, whose prior distribution $p(z)$ is defined by the constraint set $\mathcal{C}_{dynamic}$. The central objective is to construct a variational posterior distribution $q_\psi(z|x)$ that approximates this logic-informed prior in probability space.

Implementation

We introduce a low-rank Bayesian adapter (LoRA-Bayes) into the decoder. As shown in Figure 3, the model optimizes the evidence lower bound (ELBO), yielding a total loss function comprising the standard language model loss $\mathcal{L}_{LLM}$ and a constraint injection term represented by the KL divergence:

$$\mathcal{L}_{total} = \mathcal{L}_{LLM}(y|x) + \beta \cdot D_{KL}(q_\psi(z|x)||p(z|\mathcal{C}_{dynamic})). \quad (3)$$

By leveraging the reparameterization trick, the model is guided to explore parameter subspaces that are both grammatically plausible and consistent with the logical prior.

3.3.3 HSVFL

Motivation

Logical drift is a common issue in long-form text generation, necessitating a post hoc correction mechanism. The HSVFL module is designed to establish a closed-loop verification system that penalizes logical fallacies using external objective criteria, compensating for the limited self-reflection capability of generative models.

Principle

HSVFL is based on solver-feedback–driven reinforcement learning. Text generation is modeled as a Markov Decision Process (MDP), where logical correctness serves as the environmental reward. At the lower level, factual consistency is verified, while at the higher level, formal reasoning is performed using the Z3 Solver, forcing the policy to converge

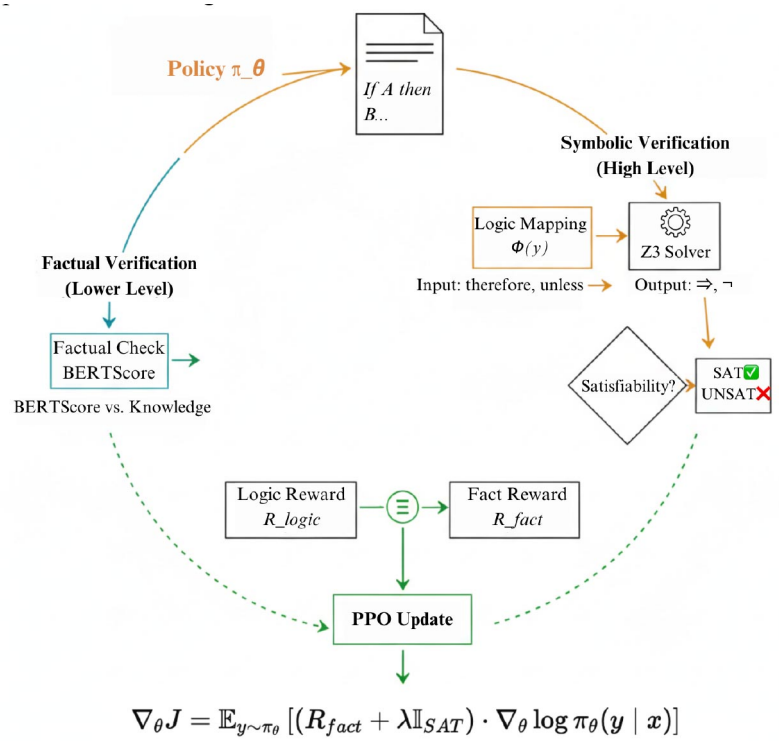


Figure 4 The Hierarchical Symbolic Verification Feedback Loop (HSVFL).

toward logically consistent regions. Note that the solver is invoked only during training; at inference time, logical constraints are internalized into model parameters.

Implementation

As shown in Figure 4, the generated text $y$is translated into a logical formula $\Phi(y)$. Specifically, dependency parsing is employed to identify main-subordinate clause structures; discourse connectives (e.g., therefore, unless) are mapped to entailment operators ($\Rightarrow$, $\lceil$) and tense consistency is encoded as temporal constraints, resulting in a first-order logic expression. The Z3 Solver evaluates its satisfiability (SAT/UNSAT) and produces a discrete logical reward R_{logic}. Combined with a factual reward R_{fact}, which is computed via BERTScore similarity against retrieved external knowledge, the policy π_θ is updated using the PPO algorithm:

$$\nabla_\theta J = \mathbb{E}_{y\ \pi_\theta}\left[(R_{fact} + \lambda \mathbb{I}_{Z3(\Phi)=SAT}) \cdot \nabla_\theta \log \pi_\theta(y|x)\right]. \tag{4}$$

This closed-loop mechanism ensures that the model continuously learns from logical errors and progressively improves its reasoning behavior.

3.4 Objective Function and Optimization

To strongly integrate the rigor of symbolic logic with the expressive capacity of neural networks, this study adopts a two-stage hybrid optimization strategy. In the first stage, the model learns logic-informed priors in latent space via variational inference (the BSCI module). In the second stage, reinforcement learning (the HSVFL module) is employed to correct reasoning policies using discrete solver feedback. The overall optimization objective $\mathcal{J}(\theta)$aims to maximize the posterior probability while satisfying logical constraints and is formulated as a weighted combination of two sub-objectives:

$$\mathcal{J}(\theta) = \underbrace{\mathcal{L}_{ELBO}(\theta)}_{\substack{\text{Stage I: Variational} \\ \text{supervision loss}}} + \gamma \cdot \underbrace{\mathcal{L}_{RL}(\theta)}_{\substack{\text{Stage II: Reinforcement} \\ \text{feedback objective}}} \tag{5}$$

where θ denotes the learnable parameters and γbalances the two training stages. We adopt a sequential training strategy, in which $\mathcal{L}_{ELBO}$ is first optimized until convergence, after which its parameters are fixed and $\mathcal{L}_{RL}$ is optimized.

3.4.1 Stage I: Variational Supervision Loss

Corresponding to the BSCI module, we optimize the evidence lower bound (ELBO) to jointly account for linguistic fluency and logical consistency. The variational loss $\mathcal{L}_{ELBO}$ consists of a reconstruction term and a regularization term:

$$\mathcal{L}_{ELBO}(\theta) = \mathcal{L}_{NLL}(\theta) + \beta \cdot \mathcal{L}_{KL}(\theta) \tag{6}$$

where $\mathcal{L}_{NLL}$ enforces grammatical correctness of the generated text y and is defined as the negative log-likelihood:

$$\mathcal{L}_{NLL}(\theta) = -\frac{1}{T}\sum_{t=1}^{T} \log P_\theta(y_t|y_t, x, z) \tag{7}$$

The KL regularization term $\mathcal{L}_{KL}$ measures the divergence between the model posterior $q_\phi(z|x)$ and the logic-informed prior $p(z|\mathcal{C}_{dynamic})$:

$$\mathcal{L}_{KL}(\theta) = D_{KL}(q_\phi(z|x)||p(z|\mathcal{C}_{dynamic})) \tag{8}$$

Table 1 Overview of experimental datasets.

Dataset Name	ProofWriter	EntailmentBank
Primary Purpose	Core evaluation and ablation; verification of symbolic mapping accuracy and HSVFL correction effectiveness	Generalization and robustness evaluation; assessment of CASCE contextual constraint extraction and consistency preservation
Task Type	English logical proof generation	English multi-step natural language entailment
Data Modality	English text + first-order logic forms	Scientific English text + entailment tree structures
Data Scale	˜570k total samples (7:1.5:1.5)	1,313 / 187 / 340 (number of problems)
Key Characteristics	Features: synthetic sentence patterns with explicit logical markers (if–then, unless); Objective: generate English reasoning chains conforming to symbolic rules	Features: scientific English with rich discourse connectives and nested clauses; Objective: generate fluent and logically self-consistent explanatory paragraphs
Source	https://proofwriter.apps.allenai.org/about	https://github.com/allenai/entailment_bank

Both distributions are modeled as Gaussian, allowing the KL divergence to be computed in closed form. To prevent posterior collapse, we apply a linear annealing strategy to the coefficient β:

$$\mathcal{L}_{KL} = \frac{1}{2}\sum_{j=1}^{d}\left(\sigma_{\phi,j}^2 + (\mu_{\phi,j} - \mu_{logic,j})^2 - 1 - \log(\sigma_{\phi,j}^2)\right) \tag{9}$$

By introducing noise $\epsilon\mathcal{N}(0, I)$ through the reparameterization trick, end-to-end gradient back propagation is enabled.

3.4.2 Stage II: Reinforcement Feedback Objective

Corresponding to the HSVFL module, reinforcement learning with PPO is employed to penalize logical fallacies based on solver feedback. The total reward function integrates factual consistency, logical satisfiability, and policy regularization:

$$R_{total}(x, y) = R_{fact}(y) + \eta \cdot \mathbb{I}_{SAT}(\Phi(y)) - \rho \cdot D_{KL}(\pi_\theta || \pi_{ref}) \tag{10}$$

where $\mathbb{I}_{SAT}$is an indicator function determined by the Z3 Solver (SAT/UNSAT). Based on this reward, we compute the clipped PPO objective to ensure stable updates:

$$\mathcal{L}_{RL}(\theta) = \mathbb{E}_t\left[\min(r_t(\theta)A_t,\ clip(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)\right] \tag{11}$$

The probability ratio $r_t(\theta)$ between the new and old policies is defined as:

$$r_t(\theta) = \frac{\pi_\theta(y_t|x)}{\pi_{old}(y_t|x)} \tag{12}$$

After incorporating the value function loss $\mathcal{L}_{VF}$ and the entropy regularization term $\mathcal{S}$, the final PPO loss becomes:

$$\mathcal{L}_{PPO}^{final} = -\mathcal{L}_{RL}(\theta) + c_1\mathcal{L}_{VF}(\theta) - c_2\mathcal{S}[\pi_\theta] \tag{13}$$

In our experiments, the logical reward weight is set to $\eta = 2.0$, and the clipping threshold is set to $\epsilon = 0.2$.

3.4.3 Optimization Settings

We use the AdamW optimizer, with parameter updates defined as:

$$\theta_{t+1} = \theta_t - \alpha\frac{m_t}{\sqrt{v_t} + \delta} - \lambda_{wd}\theta_t \tag{14}$$

All experiments are conducted on a workstation equipped with four NVIDIA RTX 4090 GPUs (24 GB each). To mitigate memory constraints, gradient accumulation is employed to simulate a global batch size of 128, and mixed-precision training (FP16) is enabled. The learning rate is set to $\alpha = 2 \times 10^{-5}$ in Stage I and $\alpha = 1 \times 10^{-6}$ in Stage II, combined with cosine decay scheduling and weight decay $\lambda_{wd} = 0.01$. These settings ensure robust joint optimization between continuous representations and discrete logical constraints.

4. EXPERIMENTS AND RESULTS

4.1 Experimental Setup

To evaluate the effectiveness of the proposed extraction-injection-verification collaborative mechanism in English text generation, we select two complementary datasets: ProofWriter, which assesses logical-syntactic control, and EntailmentBank, which evaluates generalization in natural discourse. Table 1 summarizes the key characteristics of the datasets. We focus on English datasets because English, as a prototypical hypotactic language, externalizes logical relations through explicit discourse markers, making it well suited for verifying the effectiveness of symbolic constraint injection. The extension of the framework to other hypotactic languages, such as German, is left for future work.

Proof Writer provides a standardized corpus with explicit logical connectives, enabling validation of the symbolic-probabilistic mapping in BSCI and the solver-based correction capability of HSVFL. Entailment Bank, consisting of scientific texts with deeply nested clauses, is used to assess CASCE's ability to extract implicit entailment structures. This dataset combination spans rule-based reasoning and

Table 2 Evaluation metrics.

Metric Name	Computation Formula	Core Interpretation	Reference for Value Interpretation
LCA	$LCA = \frac{1}{N}\sum \mathbb{I}(Z3(\Phi(y)) = SAT)$	Core metric. Uses the Z3 solver to verify satisfiability of generated text mapped into symbolic logic, detecting deep contradictions.	Values approaching 1.0 are optimal. Baseline models on natural discourse are typically < 0.75; significant gains indicate effective symbolic constraint injection.
RCC	$RCC = \frac{1}{N}\sum \mathbb{I}(Path_{gen} \equiv Path_{gold})$	Process-level metric. Compares generated intermediate reasoning paths with gold standards, preventing "correct answer but incorrect reasoning."	Higher is better. In 5-hop reasoning, values > 0.7 indicate strong long-range causal control.
BERTScore (Semantic Similarity)	$R_{BERT} = mean(\cos(e_{gen}, e_{ref}))$	Language-level metric. Measures semantic embedding similarity to assess idiomaticity and semantic faithfulness of English expression.	Range [0, 1]. Scores > 0.85 indicate high semantic fidelity and ensure logic constraints do not degrade fluency.

natural discourse generation, allowing the evaluation of robustness against long-range logical drift.

Preprocessing is tailored to the neuro-symbolic architecture. For ProofWriter, explicit mappings are constructed between English functional expressions (e.g., if-then) and first-order logic operators ($\Rightarrow$, $\neg$), yielding aligned corpora to calibrate the variational priors in BSCI. For EntailmentBank, dependency parsing is performed using Stanford CoreNLP to extract discourse cues and construct adjacency matrices for initializing CASCE edge weights. This standardized processing strengthens model sensitivity to both explicit and implicit logical markers, ensuring that structured priors are effectively utilized during generation.

Experiments are conducted on four NVIDIA RTX 4090 GPUs (24 GB) using PyTorch 2.1 with CUDA 11.8. The backbone model is LLaMA-2-7B, fine-tuned with LoRA and optimized using AdamW (Batch = 128). The learning rates for the two training stages are 2×10^{-5} and 1×10^{-6}, respectively. Core module settings include a two-layer GCN for CASCE, a KL annealing coefficient $\beta = 0.1$ for BSCI, and a logical reward weight $\eta = 2.0$ for HSVFL. All experiments fix the random seed to 42, and results are reported as mean $\pm$ standard deviation over five independent runs.

We construct a logic-semantic dual-dimensional evaluation framework (Table 2). Logic Consistency Accuracy (LCA) is adopted as the primary metric, directly using the Z3 solver to assess symbolic satisfiability, serving as the gold standard for verifying constraint injection. Reasoning Chain Correctness (RCC) further enforces correctness of intermediate inference steps. BERTScore is used to monitor semantic fidelity, ensuring that improved logical rigor does not compromise idiomaticity or fluency.

4.2 Baselines

To objectively evaluate the performance of the proposed approach, we categorize the comparison models into classical constraint-based decoding methods and recent state-of-the-art (SOTA) neuro-symbolic approaches. Among the classical methods, we select NeuroLogic Decoding [22] as a representative baseline. This method introduces hard predicate-logic constraints during decoding by forcibly pruning candidate tokens to ensure logical validity. Although this method pioneered logic-controlled generation, such discrete and rigid interventions often disrupt the natural fluency of English expression. As a result, it struggles to balance idiomaticity and logical completeness in complex long-form generation and is unable to handle soft semantic entailments.

For the most relevant SOTA approaches, we compare against three frontier works published in 2023. LINC [23] and Logic-LM [24] both adopt a translate-solve paradigm, invoking external solvers for reasoning, whereas SATLM [25] enhances constraint awareness through declarative prompting. In contrast to these methods, which typically rely on sequential pipeline architectures, are highly sensitive to intermediate translation accuracy, and are prone to cascading failures, our work targets end-to-end deep integration. Rather than employing externally attached solvers, we introduce the BSCI variational injection mechanism, which softens symbolic constraints into probabilistic priors in latent space. This design enables real-time balancing between the flexibility of English expression and the rigor of logical rules during generation.

In summary, existing baselines commonly face a trade-off between logical rigor and linguistic flexibility: hard constraints degrade fluency, while sequential solving pipelines are prone to error accumulation. This tension motivates our introduction of Bayesian soft constraints and a reinforcement-based feedback loop. Accordingly, our method explores a new pathway for resolving this trade-off, with the goal of improving LCA while maintaining or enhancing BERTScore relative to hard-constraint baselines, thereby better reconciling formal correctness with semantic fluency in English generation.

4.3 Quantitative Results

To evaluate the performance of the proposed method in terms of logical consistency and language quality in English generation, Table 3 reports a quantitative comparison

Table 3 Performance comparison of ProofWriter and EntailmentBank

Model / Method	ProofWriter (Controlled Logic)			EntailmentBank (Natural Discourse)		
	LCA (↑)	RCC (↑)	BERTScore	LCA (↑)	RCC (↑)	BERTScore
NeuroLogic	0.78 ± 0.02	0.74 ± 0.03	0.81 ± 0.01	0.65 ± 0.03	0.58 ± 0.04	0.79 ± 0.02
LINC	0.88 ± 0.01	0.85 ± 0.02	0.86 ± 0.01	0.71 ± 0.04	0.63 ± 0.05	0.84 ± 0.01
Logic-LM	0.89 ± 0.01	0.86 ± 0.01	0.87 ± 0.01	0.73 ± 0.03	0.65 ± 0.04	0.85 ± 0.01
SATLM	0.82 ± 0.02	0.79 ± 0.02	0.88 ± 0.01	0.69 ± 0.02	0.61 ± 0.03	0.87 ± 0.01
SymLoop (Ours)	0.94 ± 0.01*	0.91 ± 0.01*	0.89 ± 0.01	0.82 ± 0.02*	0.76 ± 0.03*	0.88 ± 0.01

*Note: * indicates statistically significant improvement over the second-best model ($p < 0.05$).*

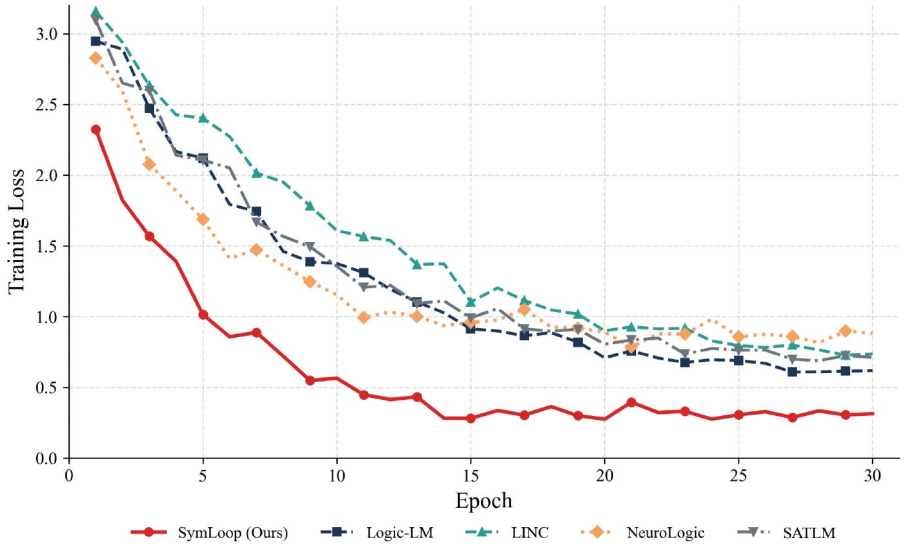


Figure 5 Training loss curves of different methods.

between SymLoop and four baseline models on ProofWriter (controlled logic) and EntailmentBank (natural discourse). All results are reported as the mean ± standard deviation over five independent runs, with statistical significance assessed using a t-test ($p < 0.05$).

As shown in Table 3, SymLoop demonstrates competitive and consistently superior performance across both datasets. On ProofWriter, NeuroLogic maintains an LCA of 0.78; however, its token-pruning mechanism leads to a relatively low BERTScore (0.81), indicating degraded naturalness of English expression. Pipeline-based methods such as LINC and Logic-LM benefit from external solvers, achieving LCA values up to 0.89. Nevertheless, on EntailmentBank, characterized by complex subordinate clauses in real-world discourse, their LCA drops to 0.73, suggesting that the translate-solve paradigm accumulates errors when handling hypotactic English structures.

In contrast, SymLoop leverages the dynamic perception of CASCE and the soft constraint injection of BSCI to achieve an LCA of 0.82 on EntailmentBank, representing a substantial improvement in logical consistency. Meanwhile, its BERTScore remains high at 0.88, indicating that Bayesian soft constraints effectively enhance logical rigor without sacrificing fluency.

We further analyze convergence behavior and inference efficiency. Figure 5 illustrates the training loss curves. External-solver-based methods (Logic-LM, LINC) rely on data-driven learning of logic mappings and typically converge after 20–25 epochs. Constraint decoding methods (NeuroLogic) converge faster albeit to a greater final loss due to the restricted optimization space imposed by hard constraints. In contrast, SymLoop benefits from the structured priors provided by CASCE and variational guidance from BSCI, achieving faster loss reduction, and converging after approximately 15 epochs. These results indicate superior training efficiency and reduced adaptation cost for complex logical tasks.

As shown in Table 4, NeuroLogic incurs substantial inference latency (125 ms/token) due to complex constraint checking at each decoding step. SOTA pipeline methods are similarly constrained by serial external solver calls. By contrast, SymLoop internalizes logical constraints into model parameters, eliminating the need for Z3 invocation during inference. As a result, SymLoop achieves an average latency of 48 ms/token, approximately 2.6× faster than NeuroLogic, validating the practicality and efficiency of the proposed end-to-end closed-loop design. Notably, the Z3 solver is used only during training to compute reinforcement learning rewards; at inference time, logical constraints are fully embedded in the model parameters, requiring no external calls.

4.4 Qualitative Results

To intuitively investigate the logical mechanisms underlying model decisions, Figure 6 visualizes two representative generation cases of the proposed method on EntailmentBank (natural discourse) and ProofWriter (controlled logic).

Table 4 Inference efficiency and computational overhead.

Model / Method	Parameter Size	Additional Inference Mechanism	Average Latency (ms/token)	Speedup (vs. Classic)
Base LLaMA-2	7B	None	42	–
NeuroLogic	7B	Discrete constraint checking	125	1.0×
Logic-LM	7B	External calls + translation	95	1.3×
Ours (SymLoop)	7.1B	Lightweight GCN	48	2.6×

Note: LINC and SATLM adopt inference mechanisms similar to Logic-LM (external solver invocation) and exhibit comparable efficiency; thus, Logic-LM is selected as a representative.

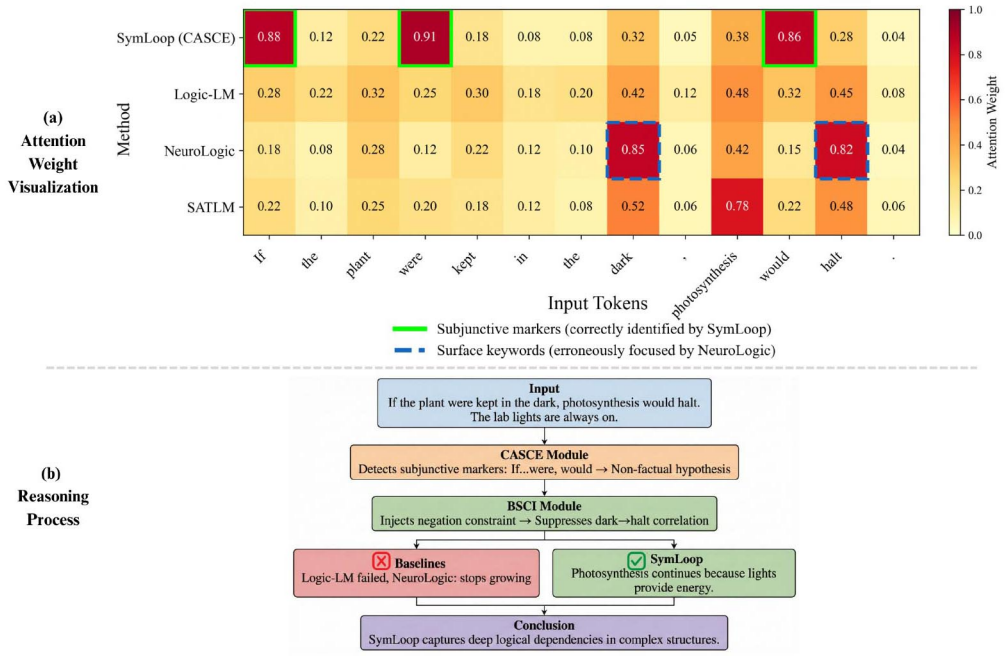


Figure 6 Successful case study on EntailmentBank. (a) Attention weight distribution across methods. (b) Reasoning process of SymLoop.

In the successful case on EntailmentBank (Figure 6), the input contains a highly misleading counterfactual conditional: "If the plant were kept in the dark, photosynthesis would halt", together with the factual statement "The lab lights are always on". The baseline model Logic-LM fails due to inaccurate translation of the subjunctive mood, leading to solver malfunction, while NeuroLogic, constrained by surface-level keyword matching, incorrectly infers that the plant stops growing. In contrast, SymLoop correctly generates: "Photosynthesis continues because the lights provide energy".This advantage stems from the CASCE module, which successfully identifies high attention weights associated with "If ⋯ were" and "would" and classifies the clause as a non-factual hypothesis. The BSCI module subsequently injects this negation constraint into the latent space, effectively suppressing the erroneous statistical intuition of "dark - halt". This case demonstrates that SymLoop can penetrate syntactic distractions and accurately capture deep logical dependencies when handling complex hypotactic English structures.

However, a limitation of SymLoop is revealed in the failure case on ProofWriter (Figure 7). The input contains the rules "Some red things are not big" and "All round things are big", and the task is to determine whether a red object is round. The model incorrectly produces a definite negative conclusion, whereas the correct answer should be "Unknown". This error arises from ambiguity in quantifier scope: the existential quantifier "some" (∃) in English often carries pragmatic implications in natural language that diverge from its formal logical semantics. Despite the logical penalties introduced by HSVFL, the model may still fall into a region of probabilistic uncertainty under the open-world assumption, leading to overconfident predictions. This observation suggests that future work should incorporate explicit uncertainty modeling to more rigorously handle boundary cases involving logical quantifiers.

4.5 Robustness Analysis

To assess model stability under non-ideal English conditions, we conduct input perturbation tests and cross-dataset transfer evaluations. Figure 8 illustrates the trends of LCA under varying disturbance conditions, in order to examine model adaptability to noise and distribution shifts.

In the input perturbation test, we randomly inject 10%–30% English synonym replacement noise into EntailmentBank. As shown in Figure 8(a), performance degrades for all models as noise increases, but with markedly different patterns. Pipeline-based methods such as LINC and Logic-LM

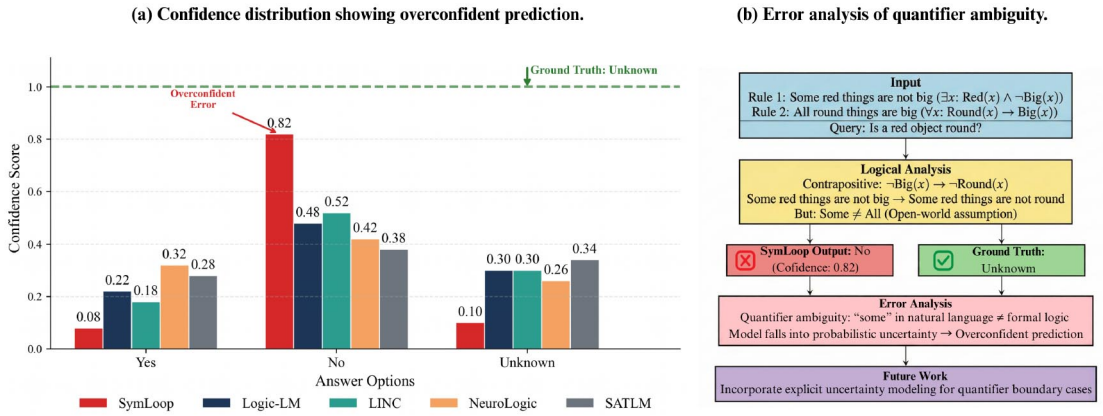


Figure 7 Failure case on ProofWriter. (a) Confidence distribution showing overconfident prediction. (b) Error analysis of quantifier ambiguity.

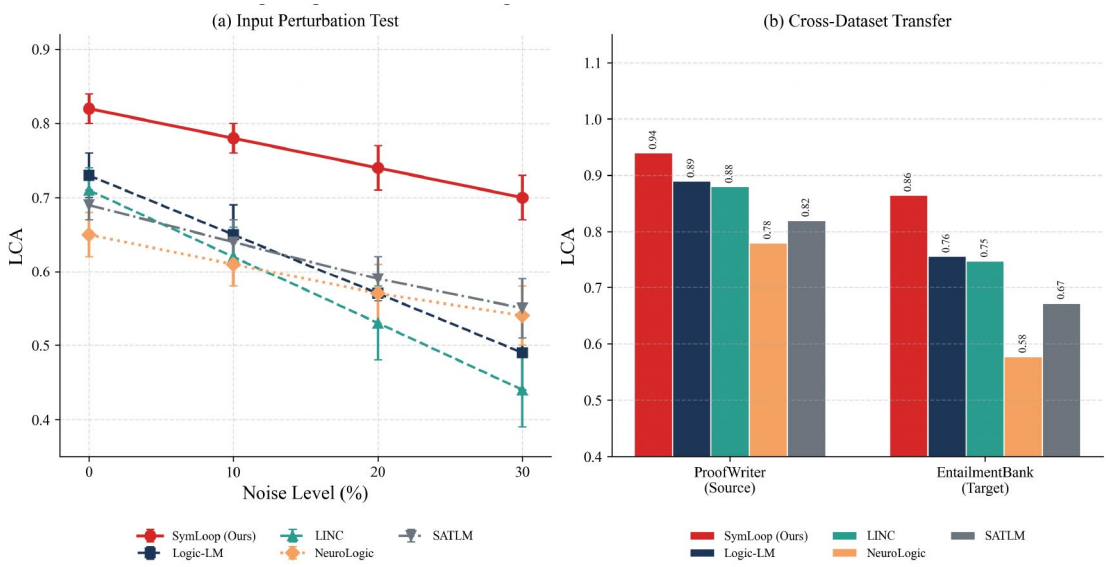


Figure 8 Robustness analysis. (a) LCA under varying noise levels. (b) Cross-dataset transfer performance.

exhibit the steepest declines (LCA drops exceeding 20%), primarily because they rely heavily on accurate translation, where even minor lexical variations can cause failures in logic formula conversion. SATLM and NeuroLogic also perform poorly, constrained respectively by prompt drift and rigid hard matching. In contrast, SymLoop shows a more gradual decline and maintains an LCA approximately 15 percentage points higher than the second-best baseline under 30% noise. This robustness is attributed to the CASCE module, which aggregates contextual semantic topology by means of graph structures rather than relying solely on surface keywords. As a result, even when discourse markers are perturbed, implicit causal dependencies can still be partially reconstructed.

In the cross-dataset transfer evaluation, models trained exclusively on ProofWriter (synthetic rules) are directly evaluated on EntailmentBank (real-world discourse) to test generalization. As shown in Figure 8(b), all baselines suffer performance degradation when exposed to the distribution shift from "canonical synthetic" to "complex natural" language. NeuroLogic, which overfits synthetic syntax, experiences the largest drop after transfer. Although Logic-LM and LINC retain some capability due to external solvers, they still exhibit around a 15% decline as they cannot adequately adapt to complex hypotactic structures. In contrast, SymLoop demonstrates better adaptability, with a performance drop of approximately 8%. This advantage is attributed to the variational prior mechanism of the BSCI module, which models logical constraints as probabilistic distributions in latent space. These soft constraint representations provide smoother generalization, enabling the model to transfer logical paradigms more flexibly to diverse real-world discourse and mitigating cross-domain logical drift.

4.6 Ablation Study

To quantitatively assess the individual contributions and coupling effects of the core components in the proposed extraction-injection-verification collaborative mechanism, we conduct systematic ablation experiments on ProofWriter and EntailmentBank. Table 5 reports the performance degradation in terms of LCA, RCC, and BERTScore after removing CASCE (dynamic constraint extraction), BSCI (Bayesian soft constraint injection), or HSVFL (symbolic verification feedback).

Table 5 Ablation Results of Core Components.

Model Variant	ProofWriter (Controlled Logic)			EntailmentBank (Natural Discourse)		
	LCA	RCC	BERT	LCA	RCC	BERT
Full Model (SymLoop)	0.94	0.91	0.89	0.82	0.76	0.88
w/o CASCE (no graph structure)	0.91	0.89	0.88	0.72 (−12%)	0.65 (−14%)	0.86
w/o BSCI (no soft constraints)	0.89	0.86	0.85 (−4%)	0.75	0.68	0.83 (−6%)
w/o HSVFL (no feedback)	0.85 (−9%)	0.80 (−12%)	0.89	0.76	0.69	0.88

The ablation results quantitatively reveal the distinct roles of each component. First, the CASCE module plays a critical role in handling unstructured natural discourse. Removing CASCE leads to substantial drops of 12% and 14% in LCA and RCC on EntailmentBank, respectively. This indicates that without graph neural networks extracting implicit logical topology, the model struggles to capture long-range causal dependencies in complex English sentences, resulting in broken reasoning chains.Second, the BSCI module effectively aligns symbolic constraints with the language generation distribution. When degraded to direct parameter concatenation, BERTScore decreases on both datasets (by 4%–6%). This suggests that without KL-regularized probabilistic injection, forcibly introducing discrete symbolic information perturbs the original semantic latent space, yielding rigid and unnatural English text.Third, the HSVFL module ensures the rigor of logical inference. The removal of solver-based feedback causes RCC on ProofWriter to drop sharply by 12%, confirming that forward implicit constraints alone are insufficient to eliminate deep logical fallacies. The incorporation of non-differentiable posterior verification and penalties is essential for guaranteeing the completeness and correctness of the reasoning process.

Overall, CASCE reduces the search space for logical features, BSCI stabilizes gradient updates, and HSVFL provides end-to-end correction capability. The closed-loop mechanism formed by these components is indispensable for jointly improving logical consistency and language generation quality. Notably, CASCE has a smaller impact on ProofWriter (approximately 3%), as explicit logical markers can be captured via surface matching, whereas implicit entailments in EntailmentBank require graph-structured extraction.

4.7 Discussion

This section provides a mechanistic interpretation and critical reflection on the experimental results from a neuro-symbolic integration perspective.

As shown in Table 3, the proposed method achieves a larger LCA improvement on EntailmentBank (+12% over the second-best baseline) than on ProofWriter (+6%). This discrepancy arises from differences in logical representation characteristics. ProofWriter's synthetic corpus contains explicit logical markers (e.g., if-then), allowing baseline methods to achieve relatively high performance by means of surface pattern matching. In contrast, logical relations in EntailmentBank's scientific discourse are often implicit, embedded within nested clauses and discourse cohesion, and can be effectively captured only through CASCE's graph-structured modeling. This observation supports our core hypothesis: strong logical dependencies in hypotactic English cannot be learned solely through statistical co-occurrence, and require explicit structural inductive bias.

At the same time, NeuroLogic's BERTScore (0.81) is significantly lower than that of our method (0.89), indicating that hard constraints improve logic at the expense of fluency. Mechanistically, hard constraints operate via discrete pruning in the vocabulary space, disrupting the co-occurrence distributions learned during pretraining. In contrast, BSCI imposes soft guidance in latent space through KL divergence, achieving a more favorable balance between logical rigor and natural expression.

Nevertheless, the quantifier-scope failure case discussed in Section 4.4 exposes a fundamental limitation of SymLoop. In natural language, the English quantifier "some" often carries pragmatic implicatures meaning "some but not all" whereas, in first-order logic, it denotes only existential quantification ($\exists$). By conflating pragmatic inference with formal reasoning, the model produces overconfident conclusions. This indicates that the current framework assumes that logical constraints can be extracted unambiguously, whereas a systematic gap exists between formal semantics and pragmatics in English quantifiers. Pure neuro-symbolic integration is insufficient to resolve such pragmatic ambiguity.

In addition, this study has several limitations. Experiments are conducted only on English datasets, and transferability to other hypotactic languages (e.g., German) remains to be explored. The use of the Z3 solver during training introduces additional computational overhead, limiting scalability to larger models. Furthermore, CASCE's reliance on attention patterns makes it sensitive to architectural variations. Future work may incorporate explicit pragmatic modeling and uncertainty quantification mechanisms to more rigorously address boundary cases involving logical quantifiers.

5. CONCLUSION

Large language models face a persistent challenge in regard to English text generation: outputs are often formally well-structured yet logically fragmented: a problem rooted in the fundamental gap between continuous neural representations and the discrete rules of symbolic logic. To address this issue, we propose SymLoop, A Symbolic Constraint Loop Framework for Consistency Preservation and Reasoning

Control in English Generation. SymLoop constructs a three-stage neuro-symbolic collaborative closed loop consisting of extraction, injection, and verification.

The core contributions of this work are threefold. First, we introduce the CASCE algorithm, which employs graph neural networks to probe attention responses to discourse connectives and dynamically capture implicit logical entailments in complex English sentence structures. Ablation studies show that CASCE contributes a 12% improvement in logic consistency accuracy on EntailmentBank, demonstrating the necessity of structured inductive bias for penetrating hypotactic English constructions. Second, we propose the BSCI algorithm, which leverages variational inference to transform deterministic logical rules into probabilistic priors in latent space. On ProofWriter, this mechanism achieves a logic consistency accuracy of 0.94 alongside a BERTScore of 0.89, confirming that soft constraint injection can attain a Pareto-optimal balance between logical rigor and linguistic fluency. Third, we design the HSVFL, which incorporates the Z3 solver to perform dual-level factual and logical verification and enables end-to-end logical correction via PPO-based policy gradients. Experimental results indicate that the removal of this module causes a 12% drop in reasoning chain accuracy, validating the critical role of posterior verification in eliminating signifcant logical fallacies.

This work introduces a new paradigm for resolving the long-standing dichotomy between probabilistic fitting and logical reasoning in neural language models. Notably, the proposed method requires no external solver invocation during inference, achieving a latency of only 48 ms/token, which is 2.6× faster than classical constrained decoding approaches. These properties make SymLoop a deployable solution for professional scenarios such as academic writing and the drafting of legal documents.

Future work will first address the quantifier-scope pragmatic ambiguity identified in the discussion. We plan to incorporate explicit pragmatic modeling and uncertainty quantification mechanisms to address the gap between the formal semantics and pragmatic interpretations of English quantifiers. In addition, we will explore cross-lingual transfer to other hypotactic languages such as German, and investigate the use of knowledge distillation to reduce the computational overhead introduced by the Z3 solver during training, thereby supporting logic-enhanced models at larger scales. Overall, this study demonstrates that strong collaboration between neural networks and symbolic reasoning is a necessary pathway toward building trustworthy artificial intelligence systems.

REFERENCES

1. Keber, M., Grubišiæ, I., Barešiæ, A., and Joviæ, A. (2024). A review on neuro-symbolic AI improvements to natural language processing. 2024 47th MIPRO ICT and Electronics Convention (MIPRO), Opatija, Croatia, IEEE.
2. Petruzzellis, F. (2025). Algorithmic reasoning in large language models and neuro-symbolic architectures.
3. Yang, S., Li, X., Cui, L., Bing, L., and Lam, W. (2025). Neuro-symbolic integration brings causal and reliable reasoning proofs. Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, USA, Association for Computational Linguistics, 5732–5744.
4. Oltramari, A. (2023). Enabling high-level machine reasoning with cognitive neuro-symbolic systems. Proceedings of the AAAI Symposium Series, Vol. 2, No. 1, 360–368.
5. Yang, Y., Gui, D., Yuan, Y., Liang, W., Ding, H., Hu, H., and Chen, K. (2023). Glyphcontrol: Glyph conditional control for visual text generation. Advances in Neural Information Processing Systems, 36, 44050–44066.
6. Suprun, A., Tvoroshenko, I., Gorokhovatskyi, V., and Yakovleva, O. (2025). Development and research of a method for the combined use of large language models for text generation.
7. Zhang, J., and Geng, Y. (2025). Research on adaptive course learning algorithmbased on reinforced feedback in English translationmodel optimization.
8. Bonagiri, V. K. (2025). Towards Trustworthy AI: Frameworks for Evaluating Consistency in Language Models (Doctoral dissertation). International Institute of Information Technology Hyderabad, India.
9. Han, Z., Battaglia, F., Mansuria, K., Heyman, Y., and Terlecky, S. R. (2025). Beyond text generation: assessing large language models' ability to reason logically and follow strict rules. AI, 6(1), 12.
10. Wu, J., Yang, S., Zhan, R., Yuan, Y., Chao, L. S., and Wong, D. F. (2025). A survey on llm-generated text detection: Necessity, methods, and future directions. Computational Linguistics, 51(1), 275–338.
11. Ciaccio, C., Dell'Orletta, F., Miaschi, A., and Venturi, G. (2024). Controllable text generation to evaluate linguistic abilities of italian llms. *Proceedings of the 10th Italian Conference on Computational Linguistics (CLiC-it 2024)*, Genoa, Italy, 221–232.
12. Yang, L., Yu, Z., Zhang, T., Cao, S., Xu, M., Zhang, W., and Cui, B. (2024). Buffer of thoughts: Thought-augmented reasoning with large language models. Advances in Neural Information Processing Systems, 37, 113519–113544.
13. Vaidya, S., Saini, J. R., Jadhav, S. D., and Tamhankar, P. A. (2026). ReTruth: Enhancing the Factual Consistency of Large Language Models through Iterative Retelling and Drift Stabilization. International Journal of Intelligent Engineering & Systems, 19(1).
14. Turpin, M., Michael, J., Perez, E., and Bowman, S. (2023). Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. Advances in Neural Information Processing Systems, 36, 74952–74965.
15. Asai, A., Wu, Z., Wang, Y., Sil, A., and Hannaneh, H. (2024). Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. International Conference on Learning Representations (ICLR), Vienna, Austria.
16. Valmeekam, K., Marquez, M., and Kambhampati, S. (2023). Can large language models really improve by self-critiquing their own plans?. arXiv preprint, arXiv:2310.08118.
17. Schmidt, D. M., and Cimiano, P. (2025). Grammar-constrained decoding for structured information extraction with fine-tuned generative models applied to clinical trial abstracts. Frontiers in Artificial Intelligence, 7, 1406857.
18. Dziri, N., Lu, X., Sclar, M., Li, X. L., Jiang, L., Lin, B. Y., and Choi, Y. (2023). Faith and fate: Limits of transformers on compositionality. Advances in Neural Information Processing Systems, 36, 70293–70332.
19. Dehal, R. S., Sharma, M., and Rajabi, E. (2025). Knowledge Graphs and Their Reciprocal Relationship with Large Language Models. Machine Learning and Knowledge Extraction, 7(2), 38.

20. Shengyuan, C., Cai, Y., Fang, H., Huang, X., and Sun, M. (2023). Differentiable neuro-symbolic reasoning on large-scale knowledge graphs. Advances in Neural Information Processing Systems, 36, 28139–28154.
21. Lund, S. T., and Villadsen, J. (2024). On Verified Automated Reasoning in Propositional Logic: Teaching Sequent Calculus to Computer Science Students. Vietnam Journal of Computer Science, 11(04), 621–644.
22. Lu, X., West, P., Zellers, R., Le Bras, R., Bhagavatula, C., and Choi, Y. (2021). Neurologic decoding: (un)supervised neural text generation with predicate logic constraints. Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, Association for Computational Linguistics, 4288–4299.
23. Olausson, T., Gu, A., Lipkin, B., Zhang, C., Solar-Lezama, A., Tenenbaum, J., and Levy, R. (2023). LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Singapore, Association for Computational Linguistics, 5153–5176.
24. Pan, L., Albalak, A., Wang, X., and Wang, W. (2023). Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, Association for Computational Linguistics, 3806–3824.
25. Ye, X., Chen, Q., Dillig, I., and Durrett, G. (2023). SatLM: Satisfiability-aided language models using declarative prompting. Advances in Neural Information Processing Systems, 36, 45548–45580.