LLM-Guided Semantic Bootstrapping for Interpretable Text Classification with Tsetlin Machines

Jiechao Gao*[1], Rohan Kumar Yadav[2], Yuangang Li[3], Yuandong Pan[1], Jie Wang[1], Ying Liu[4], and Michael Lepech[1]

[1]Stanford University
[2]Independent Researcher
[3]University of California, Irvine
[4]University of the Chinese Academy of Sciences
[1]{jiechao, ydpan, jiewang, mlepech}@stanford.edu
[2]errohanydv@gmail.com, [3]yuanganl@uci.edu, [4]liuy@ucas.ac.cn

Abstract

Pretrained language models (PLMs) like BERT provide strong semantic representations but are costly and opaque, while symbolic models such as the Tsetlin Machine (TM) offer transparency but lack semantic generalization. We propose a semantic bootstrapping framework that transfers LLM knowledge into symbolic form, combining interpretability with semantic capacity. Given a class label, an LLM generates sub-intents that guide synthetic data creation through a three-stage semantic bootstrapping curriculum (seed, core, enriched), expanding semantic diversity. A Non-Negated TM (NTM) learns from these examples to extract high-confidence literals as interpretable semantic cues. Injecting these cues into real data enables a TM to align clause logic with LLM-inferred semantics. Our method requires no embeddings or runtime LLM calls, yet equips symbolic models with pretrained semantic priors. Across multiple text classification tasks, it improves interpretability and accuracy over vanilla TM, achieving performance comparable to BERT while remaining fully symbolic and efficient.

1 Introduction

The Tsetlin Machine (TM) has recently gained traction in artificial intelligence (AI) due to its transparent learning process, interpretable structure, and fully explainable outputs (Granmo, 2018). It has shown promising results in various natural language processing (NLP) tasks, including document classification (Abeyrathna et al., 2023), sentiment analysis (Yadav et al., 2021b), topic classification (Yadav et al., 2021a), and fake news detection (Bhattarai et al., 2022). Thanks to its clause-based symbolic logic, the TM is particularly suitable for high-stakes domains such as legal and medical document analysis, where interpretability and decision traceability are critical (Saha and Jyhne, 2022; Gao et al., 2025; Berge et al., 2018).

Despite these advantages, the TM's symbolic nature imposes limitations. Operating on Boolean bag-of-words (BoW) representations, it struggles to generalize across semantically related terms unless explicitly observed in training. In contrast, pretrained language models (PLMs) such as BERT provide strong semantic representations by capturing contextual relationships (Devlin et al., 2019), but they are costly and opaque. Prior attempts to close this gap—for example, enriching TM inputs with word vectors from embedding models like Word2Vec or GloVe (Yadav et al., 2021a)—have achieved only limited semantic alignment.

To address this, we propose an **LLM-guided semantic bootstrapping framework** that integrates high-level semantic knowledge from large language models (LLMs) into the TM pipeline through symbolic augmentation rather than embeddings. The framework prompts an LLM to decompose class labels into interpretable sub-intents (e.g., *positive_due_to_plot*) and generate synthetic training samples via a structured curriculum. These examples guide clause formation in a Non-Negated TM (NTM), whose extracted literals enrich real-world data with interpretable semantic features. A standard TM fine-tuned on this hybrid representation benefits from LLM-inferred semantics while preserving symbolic transparency and efficiency. Our approach thus bridges the gap between symbolic reasoning and pretrained language understanding without requiring embeddings or LLM inference at runtime.

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2026, pages 30210–30222

2 Related Work

Semantic Modeling and Symbolic Interpretability: Pretrained transformer-based language models such as BERT, T5, and GPT have achieved state-of-the-art performance across a wide range of NLP tasks (Devlin et al., 2019; Raffel et al., 2019; Brown et al., 2020; Liu, 2019). However, their high computational cost and lack of interpretability limit their use in high-stakes domains such as healthcare or law (Rudin, 2018; Jain and Wallace, 2019; Wu et al., 2023).

Symbolic models offer an interpretable alternative. The Tsetlin Machine (TM), in particular, provides clause-level transparency through propositional logic (Granmo, 2018), and has been applied to sentiment analysis, topic classification, and explainable fact-checking (Yadav et al., 2021b,a; Bhattarai et al., 2022). However, its reliance on literal bag-of-words representations limits its ability to generalize across semantically similar inputs unless explicitly encoded. Augmenting TM inputs with static embeddings like GloVe (Yadav et al., 2021a) offers limited semantic alignment and fails to capture context-dependent meaning.

LLM-Based Supervision for Interpretable Systems: Recent approaches have explored leveraging large language models (LLMs) to generate weak supervision signals, synthetic data, or structural priors through prompt-based interaction (Ding et al., 2024; Hsieh et al., 2025; Yeh et al., 2025). These strategies aim to incorporate the semantic capabilities of LLMs without depending on them at inference time. Concurrently, symbolic distillation methods attempt to extract rules or interpretable logic from LLM outputs (McDonald and Emami, 2024; Xie et al., 2025; Xu et al., 2024), though they typically target decision tree structures or linear rules.

Our work builds on these ideas by enabling clause-based symbolic models to inherit semantic priors through structured sub-intent supervision. Unlike prior methods, we maintain full symbolic transparency and efficiency, without embedding layers or runtime LLM calls.

3 Proposed Model

We propose a LLM-guided semantic bootstrapping framework for interpretable text classification with Tsetlin Machines, without relying on embeddings or LLM inference during deployment. Our method operates in three key stages: (1) LLM-guided discovery of interpretable sub-intents and structured data generation, (2) pretraining a Non-Negated Tsetlin Machine (NTM) to extract semantic clause patterns, and (3) semantic enrichment of real data using these patterns, followed by fine-tuning a standard TM. The resulting pipeline remains fully interpretable, data-efficient, and domain-adaptable.

3.1 LLM-Guided Sub-intent Discovery and Semantic Bootstrapping

Large language models (LLMs) have demonstrated strong capabilities in extracting structured semantic information through prompting, instruction tuning, and zero-shot reasoning (Wang et al., 2023; Tripathi et al., 2025). We leverage these capabilities not for direct classification, but for generating symbolic scaffolds that guide clause discovery in symbolic models such as Tsetlin Machines (TMs). Rather than predicting labels, we prompt the LLM to identify *fine-grained sub-intents*—interpretable drivers that explain why a sample belongs to a particular class.

To support generalization across domains, we design a generic prompt template that operates over any labeled dataset. The prompt includes placeholders for dataset name, domain description, and class labels, and instructs the model to generate sub-intents in the desired format. These sub-intents serve as symbolic anchors for interpretable clause learning and help inject domain knowledge into downstream models. For all synthetic data generation, we used `gpt-4o` (Azure OpenAI, May 2024 release). We adopted nucleus sampling with $p = 0.9$, temperature = 0.7.

To simulate sub-intent distributions and build symbolic associations, we design a three-stage synthetic data generation pipeline rather than relying on a single prompt. This multi-step generation process is crucial for clause-based models like the TM, which require stability in lexical patterns to learn interpretable clauses, as well as exposure to syntactic and semantic diversity to avoid overfitting to surface form (Yadav et al., 2022). **Seed Stage.** The LLM is prompted with real domain-specific examples and asked to generate short, semantically faithful samples (approximately 15–20 words) for a given sub-intent. These serve as canonical expressions and provide clause-level anchors for the TM to begin learning consistent patterns. **Core Stage.** Using the seed examples as anchors, the LLM generates structurally varied yet lexically stable samples. This variation ensures that the TM

Table 1: The Type I and Type II Feedback.

Input	Type I Feedback				Type II Feedback			
	Clause	1		0	Clause	1		0
	Literal	1	0	1 0	Literal	1	0	1 0
Include Literal	P(Reward)	$\frac{s-1}{s}$	NA	0 0	P(Reward)	0	NA	0 0
	P(Inaction)	$\frac{1}{s}$	NA	$\frac{s-1}{s}$ $\frac{s-1}{s}$	P(Inaction)	1.0	NA	1.0 1.0
	P(Penalty)	0	NA	$\frac{1}{s}$ $\frac{1}{s}$	P(Penalty)	0	NA	0 0
Exclude Literal	P(Reward)	0	$\frac{1}{s}$	$\frac{1}{s}$ $\frac{1}{s}$	P(Reward)	0	0	0 0
	P(Inaction)	$\frac{1}{s}$	$\frac{s-1}{s}$	$\frac{s-1}{s}$ $\frac{s-1}{s}$	P(Inaction)	1.0	0	1.0 1.0
	P(Penalty)	$\frac{s-1}{s}$	0	0 0	P(Penalty)	0	1.0	0 0

learns invariant features across syntax, which is essential for symbolic models that cannot rely on latent context to smooth over form variation (Li et al., 2023). **Enriched Stage.** Finally, the LLM is prompted to introduce novel but semantically aligned expressions through modifiers, synonyms, and compositional phrasing. This stage expands the lexical space while maintaining intent consistency, helping TM clauses generalize beyond exact string matches.

This multi-stage strategy adopts a generation curriculum, reflecting the principles of curriculum learning (Bengio et al., 2009), where simpler, canonical cases are learned first before introducing complexity. Prior work has shown that LLMs, when prompted with complex generation tasks in a single step, tend to collapse into high-probability patterns or overly generic phrasing (Yun et al., 2025). By explicitly structuring generation into progressive stages, we ensure coverage, lexical variation, and semantic fidelity—each critical for effective clause formation in Boolean-symbolic models like the TM.

3.2 Pretraining the Non-Negated Tsetlin Machine (NTM)

To better understand our framework, we give an overview of the basic TM architecture to understand how the vanilla TM functions in general.

3.2.1 Tsetlin Machine

The Tsetlin Machine (TM) learns correlations between features and labels using propositional logic (Granmo, 2018). A propositional logic formula in the TM, known as a clause, is a conjunction of negated and non-negated forms of the input features. These features are referred to as literals and are controlled by a set of Tsetlin Automata (TAs). Each input feature corresponds to two TAs: one (*TA*) controls the original (non-negated) form of the literal, while the other (*TA'*) controls its negation. Each TA decides whether to include or exclude the literal and operates with two actions (Include/Exclude) across $2N$ states. When a TA moves from state 1 to N, it performs the Exclude action; when it moves from state $N+1$ to $2N$, it performs the Include action. Each move of a TA is triggered by feedback in the form of Reward, Penalty, or Inaction (Granmo, 2018).

The training of the TM, as detailed in (Granmo, 2018), involves learning the optimal actions through feedback mechanisms, specifically Type I and Type II feedback, as shown in Table 1. Consider a training example $(\boldsymbol{X}, y)$, where the input vector $\boldsymbol{X}$ is a bag-of-words (BoW) representation and y is the ground-truth label. The predicted label is denoted by $\hat{y}$. Multiple teams of TAs are responsible for TM learning, with each clause being assigned one TA per literal. Each TA operates in an environment defined by the training samples and the updating rule.

In the TM, the feedback mechanism is crucial for learning. Type I feedback occurs during true positive cases, when the clause output and the target agree. This feedback reinforces the inclusion of literals that correctly identify the positive class by rewarding the TAs that contributed to the correct classification—moving them away from the center states (toward the endpoints) and solidifying their actions. Conversely, Type II feedback occurs during false positive cases, when the clause output is positive but the target is negative. This feedback penalizes the inclusion of literals that incorrectly identify the class by moving the TAs toward the center states, weakening their actions and potentially switching their decisions.

The most important component of TM is the clause, which represents a certain sub-pattern among a particular set of patterns. This sub-pattern is in propositional AND-form, making it highly interpretable and suitable for logical understanding of the task. To illustrate, consider a bag-of-words input $X = [x_1, \cdots, x_n]$, where $x_k \in \{0, 1\}$ indicates the presence or absence of a word in a sentence, and n represents the vocabulary size. Assuming γ classes, with each class requiring α clauses, the model comprises $\gamma \times \alpha$ clauses C_ι^κ, defined as:

$$C_\iota^\kappa = \left(\bigwedge_{k \in I_\iota^\kappa} x_k \right) \wedge \left(\bigwedge_{k \in \bar{I}_\iota^\kappa} \neg x_k \right), \qquad (1)$$

where I_ι^κ and $\bar{I}_\iota^\kappa$ are non-overlapping subsets of the input variable indices, $I_\kappa^\iota, \bar{I}_\kappa^\iota \subseteq \{1, \cdots, n\}, I_\kappa^\iota \cap \bar{I}_\kappa^\iota = \emptyset$. The set I_ι^κ includes indices of features that TAs include in their original

form, while $\bar{I}_\iota^\kappa$ includes indices of features in their negated form.

In each class, clauses with odd indexes are assigned positive polarity (+), voting in favor of the target class, while those with even indexes are assigned negative polarity (-), voting against it. The overall output for a class is obtained by subtracting the number of negative votes from the number of positive votes, as follows:

$$f^\kappa(X) = \sum_{\iota=1,3,\ldots}^{\alpha-1} C_\iota^\kappa(X) - \sum_{\iota=2,4,\ldots}^{\alpha} C_\iota^\kappa(X). \quad (2)$$

For γ classes, the final classification output $\hat{y}$ is determined by the argmax operator, which selects the class with the highest net sum of votes:

$$\hat{y} = \text{argmax}_\kappa \left(f^\kappa(X) \right). \quad (3)$$

3.3 Non-Negated Tsetlin Machine (NTM)

To inject LLM-derived semantic knowledge into clause logic while preserving transparency, we introduce the Non-Negated Tsetlin Machine (NTM)—a variant of the standard TM tailored for symbolic semantic supervision. The NTM modifies the learning process in two key ways: (1) it eliminates negated literals from clause construction to improve monotonic interpretability, and (2) it applies *boosted Type I feedback* to stabilize high-confidence literal selection.

Clause Structure. Unlike the standard TM which uses both positive and negated literals (Equation 1), the NTM restricts clause formation to only original (positive) literals. Each clause becomes a purely monotonic conjunction:

$$C_\iota^\kappa = \bigwedge_{k \in I_\iota^\kappa} x_k, \quad (4)$$

where $I_\iota^\kappa \subseteq \{1, \ldots, n\}$ denotes the index set of included features. This modification reduces clause complexity and ensures all learned rules reflect affirmatively correlated lexical patterns, which aligns with the goal of identifying constructive sub-intent semantics.

Feedback Adjustment. To further enhance stability and interpretability, the NTM modifies the standard probabilistic Type I update by applying a boosted scheme. Specifically, for TAs associated with literals contributing to a correct prediction, the reward probability is increased from the standard $\frac{s-1}{s}$ to 1.0, and penalties are disabled (set to 0). Formally:

- Type I Reward: $P_{\text{reward}} = 1.0$
- Type I Penalty: $P_{\text{penalty}} = 0.0$

This aggressive reinforcement encourages TAs to quickly converge on a high-confidence set of literals per clause that robustly define the sub-intent semantics from the synthetic dataset. Importantly, *Type II feedback remains active* in the NTM, as in the standard TM, to suppress false positives and maintain class-level discrimination. Since NTM excludes negated literals by design, all feedback (including penalties) applies only to non-negated literals, ensuring semantic consistency and interpretability.

Literal Extraction. After pretraining the NTM on the synthetic samples generated for each sub-intent, we analyze the final states of the TAs associated with each clause. Literals with the deepest TA states (i.e., those furthest from the Exclude threshold) are selected as the most confident semantic indicators of that sub-intent. These literals serve as interpretable symbolic features and are retained for enriching real-world samples in the fine-tuning stage.

This simplification of clause structure and adjustment of feedback dynamics makes the NTM an ideal intermediate module for transferring LLM-inferred symbolic knowledge into clause-based learners without introducing inference-time complexity.

3.4 Semantic Feature Injection and TM Fine-tuning

The final stage of the framework injects the extracted semantic knowledge into real labeled data. Each sample in the original dataset is passed through the NTM, which outputs predicted sub-intents and their activated clauses. From these, we collect the corresponding high-confidence literals and append binary indicators of their presence to the sample's original bag-of-words (BoW) representation.

This enriched Boolean input combines direct lexical evidence with symbolic semantic abstractions derived from LLM-guided pretraining. A standard TM is then fine-tuned on this augmented dataset using both Type I and Type II feedback. During training, the TM learns interpretable clause logic over both raw and semantically inferred inputs.

Importantly, the entire pipeline introduces no new components at inference: the final model remains purely symbolic and efficient. All semantic

augmentation occurs offline during training, and the prediction process uses clause voting based on Boolean inputs alone. This design preserves the transparency, auditability, and deployability of the TM while leveraging LLMs to guide symbolic generalization during learning.

3.5 Interpretable Learning in NTM: Sub-Intent Clause Structuring

The Non-Negated Tsetlin Machine (NTM) produces inherently interpretable models by learning symbolic clauses that capture coherent semantic patterns. Interpretability arises from two constraints: (i) exclusion of negated literals, yielding monotonic conjunctions, and (ii) boosted Type I feedback, which deterministically rewards literals that co-occur with correct predictions. These design choices ensure clauses are both human-readable and semantically meaningful.

Unlike prior work where clauses are trained on coarse class labels, our framework uses fine-grained *sub-intents*—interpretable, LLM-derived factors such as `politics_due_to_election`. Each sub-intent is assigned a dedicated pool of clauses (150 in our setup), allowing extraction of class-specific symbolic features without polarity assignments.

Monotonic Clause Semantics Each clause is a conjunction of included features:

$$C_j^{\kappa}(X) = \prod_{x_k \in I_j^{\kappa}} x_k, \tag{5}$$

where $I_j^{\kappa} \subseteq \{1, \ldots, n\}$ indexes features selected by Tsetlin Automata (TAs). Clauses activate only in the presence of affirmatively correlated features, ensuring transparent behavior.

Automaton-Guided Literal Selection Each literal x_k is governed by a TA with state $\phi_{jk} \in \{1, \ldots, 2N\}$, determining whether it is included ($\phi_{jk} > N$) or excluded ($\phi_{jk} \leq N$). We define confidence as

$$\text{Conf}_{jk} = \max(0, \phi_{jk} - N), \tag{6}$$

and select literals with Conf_{jk} exceeding a stability threshold as semantic cues. We fixed the stability threshold at $\delta = 5$ across all datasets, following standard TM heuristics.

Boosted Feedback Dynamics NTM modifies standard Type I feedback by setting $P_{\text{reward}} = 1.0$ and $P_{\text{penalty}} = 0.0$, ensuring stable inclusion of semantically important literals. Type II feedback remains unchanged to suppress false positives.

Clause Evolution Example On synthetic sub-intent samples, NTM clauses converge to compact symbolic rules. For instance:

- $C^{\texttt{politics\_due\_to\_election}} = \texttt{parliament} \wedge \texttt{election}$
- $C^{\texttt{sports\_due\_to\_weather}} = \texttt{championship} \wedge \texttt{rain}$

These clauses evolve into sub-intent–specific literal clusters (e.g., {`parliament`, `election`, `results`}), which form symbolic embeddings used for downstream fine-tuning. Additional examples are included in Appendix B.

4 Interpretable Feature Groups via Sub-Intents

To assess the semantic coherence and interpretability of symbolic learning guided by LLM-derived sub-intents, we extract representative feature groups from trained Non-Negated Tsetlin Machines (NTMs). Each sub-intent is associated with a dedicated set of clauses, and within these, we identify high-confidence literals based on Tsetlin Automaton (TA) states.

These literals—interpreted as semantic keywords—form a symbolic signature of the sub-intent. Concretely, for each clause $\mathcal{C}_j^{\kappa}$ belonging to sub-intent κ, we select the top-ranked literals $\{x_k\}$ whose automaton states $\phi_{jk} > N + \delta$ exceed a confidence threshold. The union of such literals across clauses yields the final feature group for sub-intent κ.

This process yields symbolic representations that are:

- statistically grounded, as they are reinforced by consistent Type I feedback across samples.
- semantically aligned, as sub-intents were derived from LLMs trained on rich world knowledge.
- interpretable, as all literals correspond to raw input vocabulary—no embeddings or opaque transformations are involved.

Below, we present qualitative examples of symbolic feature groups for selected sub-intents across multiple domains and datasets. These reflect what the model has learned to associate with specific semantic drivers.

Hallmarks of Cancer (HoC)

Sub-intent: *activating invasion and metastasis_due_to: Activation of Notch1 signaling pathway under hypoxia*
Feature group: cell, activation, invasion, signaling, carcinoma, metastasis, hypoxia, pathway

Sub-intent: *deregulating cellular energetics_due_to: ALA inhibiting Warburg effect and inducing cancer cell death*
Feature group: ALA, Warburg, glycolysis, cell, apoptosis, cancer, metabolism, death

Sub-intent: *sustaining proliferative signaling_due_to: Overexpression of growth factor receptors*
Feature group: receptor, proliferation, EGFR, growth, tumor, overexpression, activation, cancer

AG News (Topic Classification)

Sub-intent: *topic_acquisition_negotiations*
Feature group: negotiation, merger, agreement, discussion, deal, acquisition, conflict, proposal

Sub-intent: *topic_agriculture_subsidies*
Feature group: agriculture, farmer, subsidy, grant, aid, crop, finance, support

IMDb (Sentiment)

Sub-intent: *positive_due_to_plot*
Feature group: story, plot, engaging, intriguing, development, twist, script, writing

Sub-intent: *positive_due_to_acting*
Feature group: actor, performance, cast, believable, expressive, delivery, role, talent

Sub-intent: *negative_due_to_plot*
Feature group: predictable, slow, boring, weak, nonsense, messy, confusing, dull

Sub-intent: *negative_due_to_acting*
Feature group: overacting, flat, awkward, unconvincing, bland, wooden, poor, amateur

5 Experiments and Results

5.1 Datasets

To evaluate the robustness and generalizability of our approach, we selected benchmark datasets across three domains: topic classification, sentiment analysis, and biomedical classification. **AG News** (Zhang et al., 2015): A news categorization dataset with four classes—World, Sports, Business, and Sci/Tech. **R8 & R52**: Subsets of the Reuters 21578 corpus with 8 and 52 topic labels respectively, commonly used for symbolic text classification. **IMDb**: A binary sentiment dataset of long-form movie reviews, capturing nuanced opinionated text. **SST-2**: A sentence-level sentiment classification dataset of short movie phrases with binary labels. **HoC** (Baker et al., 2017): The Hallmarks of Cancer dataset with biomedical abstracts labeled for cancer-related biological processes.

5.2 Experimental Setup

We pretrained the Non-Negated Tsetlin Machine (NTM) using synthetic data generated via LLM-guided semantic bootstrapping. For each sub-intent, we generated **50 seed samples**, **50 core samples**, and **100 enriched samples**, totaling **200 samples per sub-intent**. This semantic bootstrapping curriculum promotes lexical diversity while maintaining semantic consistency. For all synthetic data generation, we used gpt-4o accessed via the Azure OpenAI API. Prompts for sub-intent discovery and Seed/Core/Enriched generation are included in Appendix A. We used nucleus sampling with $p = 0.9$, temperature = 0.7.

For NTM pretraining, we used $C = 150$ clauses per sub-intent to capture diverse monotone patterns. Unlike the standard TM, the NTM employs only positive literals, so polarity assignments (positive/negative clause pairs) are not used in this stage. The threshold was set to $T = 5000$ to ensure sufficient evidence accumulation, and specificity to $s = 5$ to balance inclusion probability and avoid overfitting. During fine-tuning, we enriched real labeled data with high-confidence literals extracted from NTM clauses and trained a standard TM with both positive and negated literals, again with $s = 5$ for clause inclusivity.

To increase sensitivity to clause strength, we adopted the integer-weighted TM variant of (Abeyrathna et al., 2020). While our equations in Section 3 describe the unweighted scoring function for clarity, in practice clause votes were weighted according to this scheme. This detail ensures consistency with prior work.

We note that hyperparameter choices (C, T, s) were based on values commonly used in TM literature (Gorji et al., 2019) and found empirically stable. However, we did not conduct full ablations (e.g., varying s, number of synthetic samples, or weighting scheme), which limits causal attribution

of the observed gains. We leave a systematic ablation study to future work.

5.3 Performance Comparison

Benchmark Selection. The baseline models reported in Table 2 are drawn from a curated set of prior works. Our selection reflects the evolution of text representation techniques for classification—ranging from traditional methods such as bag-of-words (BoW) and TF-IDF, to neural architectures like CNNs and RNNs, and embedding-based models such as fastText and BERT. We conclude with the Tsetlin Machine (TM) and its GloVe-enhanced variant, which mark a shift toward symbolic and interpretable learning. Results for comparison are obtained from the following benchmarks: **BoW, TF-IDF** (Jones, 2021; Baker et al., 2016; Ali et al., 2021; Liu et al., 2018), **CNN, RNN/LSTM, RNN/LSTM (GloVe)** (Zhang et al., 2015; Hochreiter and Schmidhuber, 1997; Dong and de Melo, 2018), **fastText** (Joulin et al., 2017; Munikar et al., 2019), **BERT-base** (Peng et al., 2019; Sun et al., 2019; Chen and Miyake, 2021), **TM, TM (GloVe)** (Yadav et al., 2021a; Kadhim et al., 2024; Bhattarai et al., 2024).

5.4 Performance Analysis

Our proposed **LLM-Guided TM** demonstrates consistent performance improvements across all datasets when compared to the vanilla Tsetlin Machine (TM) and even TM variants enriched with GloVe embeddings. Notably, our model achieves significant gains on low-context or sentiment-heavy datasets like **SST2** (+9.63% over TM and +8.86% over TM (GloVe)) and **IMDB** (+1.48% over TM), where clause-level understanding benefits greatly from semantically enriched features.On topic classification benchmarks, our model surpasses or matches neural models in accuracy. For instance, on **AG-News**, it outperforms TM by +4.76%, TM (GloVe) by approximately +3%, and narrows the gap to BERT to just 1.65%. On **R8**, it slightly outperforms TM (GloVe) and approaches BERT (97.88% vs. 97.49%). Similarly, on **R52**, our model surpasses TM by nearly 10% and TM (GloVe) by over 5%, while performing comparably to BERT (94.45% vs. 94.26%). In the biomedical domain (**HoC**), we report micro-F1 for HoC which often contains noisy or sparsely annotated samples, our model achieves 81.90%, reducing the performance gap with BERT (82.90%) while outperforming all TM variants and neural baselines such as RNN and char-CNN.

These results collectively illustrate that our LLM-Guided TM delivers competitive performance on par with transformer-based models—while preserving the full interpretability and symbolic reasoning capabilities of clause-based learning. Crucially, the improvement stems from our LLM-guided NTM's ability to learn semantically coherent and class-specific feature sets through interpretable clauses. These symbolic features, once transferred to the original data, provide enriched input representations that enable the downstream TM to fine-tune on more informative, context-aware signals—resulting in both accuracy gains and traceable decision logic.

6 Empirical Analysis: Symbolic Enrichment Enables Generalization

To examine how symbolic enrichment improves model performance, we analyze cases *misclassified by vanilla TM* but *correctly classified after enrichment*. Vanilla TM, operating on bag-of-words (BoW) inputs, discards rare tokens and cannot capture synonyms or compositional structure, since clauses activate only on exact conjunctions of observed literals. These limitations hinder generalization to unseen or lexically varied expressions. Our pipeline mitigates this by prompting an LLM to generate sub-intents and diverse synthetic examples across Seed, Core, and Enriched stages. A Non-Negated TM (NTM) trained on this corpus extracts high-confidence literals spanning a broader vocabulary. Injecting these literals into real data equips the final TM with semantically informed features that are otherwise inaccessible.

Importantly, enrichment features are injected deterministically from NTM clause activations, ensuring that no LLMs are queried at inference. This preserves reproducibility and makes the final pipeline fully symbolic. Injected features are binary indicators derived deterministically from NTM clause activations, meaning that rare or decomposed forms can be recovered (e.g., `immune`, `suppression` from `immunosuppression`) while preserving auditability.

We illustrate this process with three representative case studies.

Case Study 1: IMDb Sentiment Misclassification. *Original sample:* "A crazy ride with a twisted ending." *Issue:* Rare words `crazy`,

Table 2: Performance of the proposed LLM-Guided TM on selected classification datasets. The bold entries indicate the highest performance, while bold and underlined indicate the second highest. A dash (–) indicates missing values not reported in literature. LLM-Guided TM results are averaged over 5 runs, and the includes standard deviation. Baseline results are drawn from literature and do not report variances, so significance testing could not be applied.

Model/Datasets	**AG-News**	**R8**	**R52**	**IMDB**	**SST2**	**HoC**
BoW TFIDF	89.64	93.74	86.95	85.30	75.50	73.4
char-CNN	87.20	94.02	85.37	80.35	80.78	76.60
fastText, h = 10	91.50	**98.10**	**94.74**	–	80.1	–
RNN/LSTM (GloVe)	92.10	96.09	90.48	80.72	80.35	76.26
BERT-base	**94.75**	97.49	94.26	**93.46**	**94.00**	**82.90**
TM	88.34	96.16	84.62	90.62	75.61	77.42
TM (GloVe)	90.12	97.50	89.14	90.88	76.38	78.78
LLM-Guided TM	**93.10** $\pm$ 0.96	**97.88** $\pm$ 0.29	**94.45** $\pm$ 0.33	**92.10** $\pm$ 0.68	**85.24** $\pm$ 1.12	**81.90** $\pm$ 1.40

`twisted` are dropped in preprocessing; remaining terms (`ride`, `ending`) are too weak for clause activation, leading to misclassification. *LLM Sub-intent:* `positive_due_to_plot` *NTM Feature Group:* {`plot`, `twist`, `engaging`, `storyline`, `writing`} *Enriched input:* ride, ending, plot, twist, engaging, storyline, writing *Interpretation:* Enrichment reintroduces semantically linked cues (`plot`, `twist`), activating clauses such as `plot` $\wedge$ `twist` $\wedge$ `ending`, and yielding the correct positive label with clause-level justification.

Case Study 2: AG News – Topic Category Recovery. *Original sample:* "Talks between the two nations collapsed over unresolved objections." *Issue:* Only diffuse tokens (`talks`, `collapsed`, `nations`) remain, leading to misclassification as `World`. *LLM Sub-intent:* `topic_acquisition_negotiations` *NTM Feature Group:* {`negotiation`, `agreement`, `deal`, `proposal`, `conflict`} *Enriched input:* talks, collapsed, nations, negotiation, agreement, deal, proposal, conflict *Interpretation:* Negotiation-related features trigger business-specific clauses such as `proposal` $\wedge$ `conflict`, correcting the prediction and providing a transparent rationale.

Case Study 3: HoC Biomedical Tagging. *Original sample:* "There was no evidence of immunosuppression." *Issue:* The compound `immunosuppression` is opaque to BoW, which lacks frequent base forms like `immune` and `suppression`. *LLM Sub-intent:* `immune_evasion_due_to_suppression` *NTM Feature Group:* {`immune`, `suppression`, `evasion`, `resistance`, `modulation`} *Enriched input:* immunosuppression, immune, suppression, evasion, resistance, modulation *Interpretation:* Decomposition introduces transparent triggers (`immune`, `suppression`, `evasion`), enabling clauses such as `immune` $\wedge$ `suppression` to fire, linking the sample to immune evasion. The result is a symbolically justified correction.

These analyses highlight the core contribution of symbolic enrichment: it recovers semantically aligned cues absent in surface tokens, enabling clauses to generalize beyond strict lexical overlap. Gains are strongest in domains with compositional or sparse vocabulary (e.g., biomedical text), while sentiment tasks remain more brittle, consistent with prior findings on the limits of synthetic augmentation (Li et al., 2023).

7 Conclusion

We presented a novel framework that bridges the gap between large language models (LLMs) and interpretable clause-based learners by using LLMs to semantically bootstrap the Tsetlin Machine (TM). Our method introduces a three-stage generation process—seed, core, and enriched—that produces high-quality, semantically structured training data grounded in fine-grained sub-intents. A Non-Negated TM (NTM) trained on this data extracts symbolic, human-interpretable feature sets, which are then used to enrich real-world samples for fine-tuning a standard TM. Through extensive experiments across sentiment analysis, topic classification, and biomedical datasets, we demonstrate that our LLM-Guided TM consistently outperforms traditional TM variants and approaches the performance of state-of-the-art transformer models—all while maintaining full transparency and clause-level interpretability. Importantly, our approach requires no LLMs or embeddings at inference time, making it lightweight, transparent, and ideal for deployment in real-time and resource-constrained environments—offering a compelling solution where

explainability, efficiency, and performance must coexist.

8 Limitation

Despite its promising results, our framework has several limitations. It depends on LLM-generated data, which may introduce subtle biases or hallucinated patterns into symbolic learning. The assumption that LLM-derived sub-intents accurately represent latent class semantics may not hold in complex or overlapping domains. In addition, removing negated literals in the Non-Negated TM improves interpretability but reduces expressive power, limiting the model's ability to capture negation or contrastive logic. Finally, our evaluation focuses mainly on accuracy and qualitative interpretability, leaving robustness, clause stability, and hyperparameter sensitivity for future exploration.

9 Acknowledgements

This research is sponsored by funding from Stanford's Center for Sustainable Development and Global Competitiveness (SDGC) and the Yonghua Foundation.

References

K. Darshana Abeyrathna, Ahmed A. O. Abouzeid, Bimal Bhattarai, Charul Giri, Sondre Glimsdal, Ole-Christoffer Granmo, Lei Jiao, Rupsa Saha, Jivitesh Sharma, Svein A. Tunheim, and Xuan Zhang. 2023. Building concise logical patterns by constraining tsetlin machine clause size. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, IJCAI '23.

Kuruge Darshana Abeyrathna, Ole-Christoffer Granmo, and Morten Goodwin. 2020. Extending the tsetlin machine with integer-weighted clauses for increased interpretability. *IEEE Access*, 9:8233–8248.

Noha Ali, Ahmed H. AbuEl-Atta, and Hala H. Zayed. 2021. Enhancing the performance of cancer text classification model based on cancer hallmarks. *IAES International Journal of Artificial Intelligence (IJ-AI)*.

Simon Baker, Imran Ali, Ilona Silins, Sampo Pyysalo, Yufan Guo, Johan Högberg, Ulla Stenius, and Anna Korhonen. 2017. Cancer hallmarks analytics tool (chat): a text mining approach to organize and evaluate scientific literature on cancer. *Bioinformatics*, 33(24):3973–3981.

Simon Baker, Anna Korhonen, and Sampo Pyysalo. 2016. Cancer hallmark text classification using convolutional neural networks. In *Proceedings of the Fifth Workshop on Building and Evaluating Resources for Biomedical Text Mining (BioTxtM2016)*, pages 1–9, Osaka, Japan. The COLING 2016 Organizing Committee.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *International Conference on Machine Learning*.

Geir Thore Berge, Ole-Christoffer Granmo, Tor Oddbjørn Tveit, Morten Goodwin, Lei Jiao, and Bernt Viggo Matheussen. 2018. Using the tsetlin machine to learn human-interpretable rules for high-accuracy text categorization with medical applications. *IEEE Access*, 7:115134–115146.

Bimal Bhattarai, Ole-Christoffer Granmo, and Lei Jiao. 2022. Explainable tsetlin machine framework for fake news detection with credibility score assessment. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4894–4903, Marseille, France. European Language Resources Association.

Bimal Bhattarai, Ole-Christoffer Granmo, Lei Jiao, Rohan Yadav, and Jivitesh Sharma. 2024. Tsetlin machine embedding: Representing words using logical expressions. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1512–1522, St. Julian's, Malta. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Lei Chen and Hirokazu Miyake. 2021. Label-guided learning for item categorization in e-commerce. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, pages 296–303, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*.

Bosheng Ding, Chengwei Qin, Ruochen Zhao, Tianze Luo, Xinze Li, Guizhen Chen, Wenhan Xia, Junjie Hu, Anh Tuan Luu, and Shafiq R. Joty. 2024. Data augmentation using llms: Data perspectives, learning paradigms and challenges. In *Annual Meeting of the Association for Computational Linguistics*.

Xin Dong and Gerard de Melo. 2018. A helping hand: Transfer learning for deep sentiment analysis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2524–2534, Melbourne, Australia. Association for Computational Linguistics.

Jiechao Gao, Rohan Kumar Yadav, Xinyuan Huang, and Jie Wang. 2025. Enhancing interpretability in self-training with tsetlin machines for mitigating noisy pseudo-labels. In *2025 IEEE International Conference on Big Data (BigData)*, pages 3960–3968. IEEE.

Saeed Rahimi Gorji, Ole-Christoffer Granmo, Adrian Phoulady, and Morten Goodwin Olsen. 2019. A tsetlin machine with multigranular clauses. *ArXiv*, abs/1909.07310.

Ole-Christoffer Granmo. 2018. The tsetlin machine - a game theoretic bandit driven approach to optimal pattern recognition with propositional logic. *ArXiv*, abs/1804.01508.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9:1735–1780.

Chih-Jou Hsieh, Catarina Moreira, Isabel Blanco Nobre, Sandra Costa Sousa, Chun Ouyang, Margot Brereton, Joaquim A. Jorge, and Jacinto C. Nascimento. 2025. Dall-m: Context-aware clinical data augmentation with large language models. *Computers in biology and medicine*, 190:110022.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Karen Spärck Jones. 2021. A statistical interpretation of term specificity and its application in retrieval. *J. Documentation*, 60:493–502.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Ahmed K. Kadhim, Ole-Christoffer Granmo, Lei Jiao, and Rishad A. Shafik. 2024. Exploring state space and reasoning by elimination in tsetlin machines. *2024 International Symposium on the Tsetlin Machine (ISTM)*, pages 1–8.

Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. 2023. Synthetic data generation with large language models for text classification: Potential and limitations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10443–10461, Singapore. Association for Computational Linguistics.

Qian Liu, Heyan Huang, Yang Gao, Xiaochi Wei, Yuxin Tian, and Luyang Liu. 2018. Task-oriented word embedding for text classification. In *International Conference on Computational Linguistics*.

Yang Liu. 2019. Fine-tune bert for extractive summarization. *ArXiv*, abs/1903.10318.

Tyler McDonald and Ali Emami. 2024. Trace-of-thought prompting: Investigating prompt-based knowledge distillation through question decomposition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 293–306, Bangkok, Thailand. Association for Computational Linguistics.

Manish Munikar, Sushil Shakya, and Aakash Shrestha. 2019. Fine-grained sentiment classification using bert. *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, 1:1–5.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, Florence, Italy. Association for Computational Linguistics.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Cynthia Rudin. 2018. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1:206 – 215.

Rupsa Saha and Sander Jyhne. 2022. Interpretable text classification in legal contract documents using tsetlin machines. *2022 International Symposium on the Tsetlin Machine (ISTM)*, pages 7–12.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*.

Sahil Tripathi, Md Tabrez Nafis, Imran Hussain, and Jiechao Gao. 2025. The confidence paradox: Can llm know when it's wrong? In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 2078–2087.

Zhiqiang Wang, Yiran Pang, and Yanbin Lin. 2023. Large language models are zero-shot text classifiers. *ArXiv*, abs/2312.01044.

Zhaofeng Wu, William Merrill, Hao Peng, Iz Beltagy, and Noah A. Smith. 2023. Transparency helps reveal when language models learn meaning. *Transactions of the Association for Computational Linguistics*, 11:617–634.

Yiran Xie, Debin Xiao, Ping Wang, and Shuming Liu. 2025. A simple yet efficient prompt compression method for text classification data annotation using LLM. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pages 511–521, Abu Dhabi, UAE. Association for Computational Linguistics.

Fangzhi Xu, Zhiyong Wu, Qiushi Sun, Siyu Ren, Fei Yuan, Shuai Yuan, Qika Lin, Yu Qiao, and Jun Liu. 2024. Symbol-LLM: Towards foundational symbol-centric interface for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13091–13116, Bangkok, Thailand. Association for Computational Linguistics.

Rohan Kumar Yadav, Lei Jiao, Ole-Christoffer Granmo, and Morten Goodwin. 2021a. Enhancing interpretable clauses semantically using pretrained word representation. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 265–274, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rohan Kumar Yadav, Lei Jiao, Ole-Christoffer Granmo, and Morten Goodwin. 2022. Robust interpretable text classification against spurious correlations using and-rules with negation. In *International Joint Conference on Artificial Intelligence*.

Rohan Kumar Yadav, Lei Jiao, Ole-Christoffer Granmo, and Morten Goodwin Olsen. 2021b. Human-level interpretable learning for aspect-based sentiment analysis. In *AAAI Conference on Artificial Intelligence*.

Catherine Yeh, Donghao Ren, Yannick Assogba, Dominik Moritz, and Fred Hohman. 2025. Exploring empty spaces: Human-in-the-loop data augmentation. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, New York, NY, USA. Association for Computing Machinery.

Longfei Yun, Chenyang An, Zilong Wang, Letian Peng, and Jingbo Shang. 2025. The price of format: Diversity collapse in llms. *ArXiv*, abs/2505.18949.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Neural Information Processing Systems*.

10 Appendix A: Prompt Design for Sub-Intent Discovery and Sample Generation

Our framework leverages structured prompt engineering to extract fine-grained sub-intents and generate domain-aligned training data using large language models (LLMs). This appendix describes the exact prompt formulations used for (1) sub-intent discovery, and (2) three-stage sample generation: seed, core, and enriched.

10.1 A.1 Sub-Intent Discovery Prompt

To identify fine-grained semantic sub-intents, we use a domain-aware zero-shot prompt that incorporates class metadata and instructs the LLM to infer concrete reasons for class membership.

Prompt Template:

> You are an AI assistant tasked with analyzing biomedical texts from the [{DATASET_NAME}] dataset. This dataset consists of medical research documents related to [{DOMAIN_DESCRIPTION}], and each document is labeled with one or more of the following categories: [{CLASS_LABELS}].
>
> Your task is to identify fine-grained sub-intents for each class label. A sub-intent is a specific, grounded reason or topic that helps explain why a document belongs to a given category. Each sub-intent should be expressed using the format: classlabelname_due_to: explanation
>
> Guidelines: - Sub-intents must be specific, non-generic, and directly related to the context in the samples. - Use terminology appropriate to the domain. - Avoid duplicates and near-duplicates; merge similar expressions into a single, unified sub-intent. - Do not include commentary or bullet points. Just return a list of sub-intents, one per line.
>
> Output format: classlabel_due_to: explanation
> classlabel_due_to: explanation
> ...

Example Output:

> oncogenesis_due_to: gene expression linked to tumor suppressors
> inflammation_due_to: elevated cytokine response following infection

10.2 A.2 Synthetic Sample Generation Prompts

Sub-intents guide controlled sentence generation in three stages: seed, core, and enriched. Each prompt adapts to the input category and example set.

A.2.1 Seed Prompt

> Category: {CATEGORY_LABEL}
>
> Below are real example sentences: - {Example 1} - {Example 2}
>
> Now generate {N} new sentences that reflect this category.
>
> Your new sentences should: - Be **at least 15–20 words long** - Introduce **new vocabulary** that is contextually consistent with the examples - Use relevant terminology, synonyms, or modifiers that naturally fit the theme - Preserve the logic and context of the category, while **extending its vocabulary coverage**
>
> Output one sentence per line. Do not include comments or labels.

A.2.2 Core Prompt

> Category: {CATEGORY_LABEL}
>
> Here are example sentences for this category: - {Seed Example 1} - {Seed Example 2}
>
> Generate {N} new sentences expressing the same theme.
>
> Your sentences must: - Be **at least 15–20 words long** - Maintain the same context and meaning as the examples - **Vary the vocabulary** by introducing new relevant terms, phrase structures, or expressions - Expand the conceptual boundaries of the category while staying grounded in its core idea
>
> Use formal and precise language. Output one sentence per line.

A.2.3 Enriched Prompt

> Category: {CATEGORY_LABEL}
>
> Here are several sentences that reflect this category using consistent language: - {Core Sample 1} - {Core Sample 2}
>
> Now generate {N} new sentences that convey the **same theme**, but **expand the vocabulary** beyond what has already been used.
>
> Your output should: - Be **at least 15–20 words long** - Use **semantically consistent but new terms** compared to the original examples - Introduce **novel modifiers, synonyms, or phrasing** that enhance lexical diversity - Avoid simple paraphrasing; instead, aim to **generalize or deepen** the expression through new vocabulary and structure
>
> Output one sentence per line. No labels or commentary.

10.3 A.3 Prompt Design Objectives

Across all stages, prompt design follows these core principles:

- **Semantic precision**: Prompts elicit topic-specific, non-generic outputs.
- **Length and coverage**: All generated sentences meet minimum clause-level complexity and vocabulary extension.
- **Noise-free structure**: Output is clean, line-separated, and free of metadata or formatting noise.

These carefully controlled prompts enable LLMs to bootstrap symbolic models with rich, interpretable representations.

11 Appendix B: Extended Clause Evolution Examples

Here we expand the Clause Evolution Example, showing the full set of sub-intents and literal clusters learned by the Non-Negated Tsetlin Machine (NTM).

Politics Sub-Intents

Examples:

- politics_due_to_election: "The prime minister addressed the parliament after the election results."
- politics_due_to_debate: "A heated debate on foreign policy took place during the senate session."

Learned Clauses:

- $C_1^{\texttt{politics\_due\_to\_election}} = \texttt{parliament} \wedge \texttt{election}$
- $C_2^{\texttt{politics\_due\_to\_debate}} = \texttt{debate} \wedge \texttt{senate}$

Semantic Clusters:

- politics_due_to_election: {parliament, election, minister, results}
- politics_due_to_debate: {debate, senate, policy, session}

Sports Sub-Intents

Examples:

- sports_due_to_performance: "Ronaldo scored twice but the rest of the match was uneventful."
- sports_due_to_weather: "The championship final was interrupted due to heavy rain."

Learned Clauses:

- $C_3^{\texttt{sports\_due\_to\_performance}} = \texttt{match} \wedge \texttt{scored}$
- $C_4^{\texttt{sports\_due\_to\_weather}} = \texttt{championship} \wedge \texttt{rain}$

Semantic Clusters:

- sports_due_to_performance: {scored, goal, match, Ronaldo}
- sports_due_to_weather: {rain, stadium, interrupted, weather}

These extended examples show how NTM clauses evolve from simple conjunctions into richer semantic clusters that align with interpretable sub-intents. In the main text we present only a subset (one politics, one sports) for brevity; here, we provide the full set to support reproducibility and interpretability analysis.