ConceptWeave: Concept-Driven Prompting through Adaptive Semantic Infilling

Atul Kumar Singh
Fujitsu Research of India Pvt. Ltd.
Bengaluru, India
atul.singh@fujitsu.com

Rahul Mishra
Fujitsu Research of India Pvt. Ltd.
Bengaluru, India
Rahul.N.Mishra@fujitsu.com

Abstract

Conventional prompting and prompt-based finetuning offer limited task-specific semantic guidance, leaving language models to derive task-specific reasoning from limited contextual cues. Existing techniques, such as using demonstrations, chain-of-thought prompting, or retrieval augmentation, can improve performance, but often at the cost of longer, more redundant prompts, higher inference overhead, and reliance on external knowledge sources. We present ConceptWeave, a concept-driven prompt-based finetuning approach built on adaptive semantic infilling. Instead of depending on external resources, ConceptWeave induces task-relevant semantic concepts from a small held-out subset of downstream training data. These concepts are organized into hierarchical memory that supports efficient retrieval during fine-tuning. During fine-tuning, it retrieves semantically relevant concepts and inserts their learned embeddings into the transformer representation immediately after the most relevant input spans, enabling fine-grained, localized conditioning of the model. As training progresses, the concepts are continuously refined and structured into a hierarchy that supports efficient retrieval while preserving semantic richness and scalability. We validate ConceptWeave across a range of NLP tasks, including misinformation detection, question answering, sentiment analysis, and toxicity detection, where it consistently outperforms strong baselines and recent state-of-the-art methods.

1 Introduction

Large language models (LLMs) have achieved impressive performance across a wide variety of natural language understanding tasks through prompting-based adaptation(Brown et al., 2020; Wei et al., 2021; Sanh et al., 2021). However, standard prompting strategies often provide only limited semantic guidance, forcing models to infer task-relevant reasoning patterns from sparse context(Schick and Schütze, 2021; Gao et al., 2021). This limitation becomes especially evident in semantically complex tasks such as misinformation detection, toxicity detection, sentiment analysis, and question answering, where subtle contextual cues and latent semantic relationships play a crucial role in determining the correct output.

💡Model Type	Standard prompting	ConceptWeave
💡Input & Prompting	🎬 Smartphone Review: "Great. Another software update that breaks everything."	🎬 Smartphone Review: "Great. Another software update that breaks [software instability] everything [sarcastic complaint]."
	LLM	LLM
🧠Prediction	POSITIVE SENTIMENT	NEGATIVE SENTIMENT
🎯Predicted Understanding	Anchors on word "Great" ignores the sarcastic flip.	Correctly captures the tension between aesthetics and movie quality.
📊Reasoning Behavior	Overweight surface-level positive vocabulary and misses polarity shift	Connects "Great" with contradiction cues and retrieves concepts like software instability, and sarcastic complaint
🏷️Analysis	Failure Mode: Cannot resolve sarcasm or contextual polarity reversal	Benefit: Captures implicit frustration and contradiction-aware sentiment reasoning across input spans

Figure 1: Comparison of standard prompting and ConceptWeave for sarcasm-aware sentiment analysis. ConceptWeave adds real-world concepts to better understand sarcasm and correctly detect negative sentiment.

Recent work has sought to improve prompting through demonstrations, chain-of-thought reasoning, rationale generation, and retrieval-based prompt enrichment(Wei et al., 2022; Wang et al., 2022; Khot et al., 2022; Rubin et al., 2022; Liu et al., 2022a). Although these methods can be effective, they often come with critical drawbacks, including longer and more redundant prompts, higher inference cost, and limited scalability. In particular, rationale-based prompting typically requires expensive per-instance LLM calls to generate reasoning traces during training or inference, making it difficult to scale efficiently(Wei et al., 2022; Wang et al., 2022). Retrieval-based approaches face a different challenge, they often rely on coarse global aug-

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 4190–4207
November 6-10, 2026 ©2026 Association for Computational Linguistics

mentation or external knowledge sources, which may introduce noisy or irrelevant information while failing to provide fine-grained semantic guidance where it is actually needed(Guu et al., 2020; Lewis et al., 2020; Izacard et al., 2023). Moreover, they largely overlook an important observation, downstream task data already encodes rich latent semantic patterns that can be reused to guide prompting. For instance, misinformation datasets often capture recurring themes like conspiracy framing or mistrust in science, while toxicity data reflects patterns of identity-based attacks and hateful language. Although models implicitly learn these concepts during fine-tuning, they are rarely extracted or explicitly leveraged for prompting.

In this work, we argue that semantic concepts induced directly from downstream training data can serve as an efficient and effective source of semantic guidance for LLMs, without relying on external retrieval systems, manually curated concept vocabularies, or per-example rationale generation. We further observe that semantic guidance is often inherently local, as different spans of an input may correspond to different semantic abstractions. As a result, globally appending semantic information to an entire prompt can dilute useful signals, increase prompt length, and fail to guide the model where semantic reasoning is actually required. Motivated by these observations, we propose ConceptWeave, a concept-driven prompt-based finetuning framework based on adaptive semantic infilling. ConceptWeave first induces compact semantic concepts from a small held-out subset of downstream training data using LLM-based concept induction, and organizes them into a self-evolving hierarchical concept memory. During fine-tuning, the framework performs span-aware semantic matching to identify salient regions of the input and injects aligned concepts as dense semantic representations into the corresponding prompt spans. This enables localized semantic conditioning, allowing the model to receive targeted semantic guidance exactly where it matters most. The concept memory is continuously refined during training through adaptive concept extraction and hierarchical organization, which together reduce retrieval overhead while preserving semantic specificity and supporting efficient scalability.

We evaluate ConceptWeave on multiple downstream NLP tasks, including misinformation detection, question answering, sentiment analysis, and toxicity detection, showing that the framework generalizes well across diverse semantic understanding settings rather than being limited to a single task domain. Experimental results demonstrate consistent gains over zero-shot, few-shot, chain-of-thought, fine-tuning, and global concept-prompting baselines. We also performed extensive ablation studies to examine the role of each major component, including span-aware semantic filling, adaptive concept evolution, hierarchical concept organization, and localized concept conditioning. Our results show that span-aware semantic infilling is substantially more effective than globally appending concepts, while the hierarchical concept structure significantly reduces retrieval cost and prompt expansion overhead. Taken together, these findings suggest that compact semantic concepts learned directly from downstream task distributions provide an efficient, scalable, and generalizable foundation for concept-driven prompt-based finetuning for large language models.

The main contributions of this work are as follows: 1) We propose ConceptWeave, a new framework for concept-driven prompt-based finetuning through adaptive semantic infilling, using semantic concepts induced directly from downstream task distributions. 2) We introduce span-aware semantic infilling, which injects localized dense semantic representations into semantically relevant regions of the prompt, enabling fine-grained semantic conditioning of language models. 3) We develop a self-evolving hierarchical concept memory that continuously adapts during fine-tuning while reducing retrieval complexity and prompt expansion overhead. 4) We demonstrate consistent improvements across multiple downstream NLP tasks and provide extensive ablation studies highlighting the importance of semantic infilling, concept evolution, and hierarchical retrieval.

2 Related Work

Concept-Based Representation Learning. Concept-based learning improves interpretability and generalization by using intermediate semantic concepts. Early work introduced concept bottleneck models and concept activation methods that align model predictions with human-understandable concepts (Koh et al., 2020; Kim et al., 2018; Chen et al., 2020; Yeh et al., 2020). Neuro-symbolic and concept-driven reasoning systems further showed the effectiveness of structured semantic representations (Mao et al., 2019;

Yi et al., 2018; Singh and Mishra, 2026). Recent work also studies concept-aware pretraining for language models (Tack et al., 2025). Unlike prior approaches, ConceptWeave automatically learns task-specific concepts from downstream data and injects them into semantically relevant regions of the model representation space.

Prompt-based Tuning. Prompting has become a widely used approach for adapting large language models (LLMs) to downstream tasks (Brown et al., 2020; Schick and Schütze, 2021; Gao et al., 2021). Chain-of-thought prompting further improved reasoning ability (Wei et al., 2022; Wang et al., 2022; Khot et al., 2022), while prefix tuning and prompt tuning introduced parameter-efficient continuous prompts (Li and Liang, 2021; Lester et al., 2021; Liu et al., 2022b). However, most existing methods rely on globally added prompts that provide coarse semantic guidance. In contrast, ConceptWeave performs span-aware semantic conditioning using dynamically retrieved concepts.

Memory, Span-level Modeling, and Diversity-aware Retrieval. Memory-augmented and hierarchical retrieval systems improve scalable semantic search and long-context reasoning (Weston et al., 2014; Malkov and Yashunin, 2018). Span-level modeling captures local semantic information more effectively than global representations (Joshi et al., 2020; Bahdanau et al., 2014; Vaswani et al., 2017). Diversity-aware retrieval and submodular optimization further improve semantic coverage and reduce redundancy (Carbonell and Goldstein, 1998; Kulesza and Taskar, 2012; Lin and Bilmes, 2011; Nemhauser et al., 1978). ConceptWeave combines hierarchical concept memory, span-aware alignment, and coverage-aware retrieval for efficient concept-driven prompting.

3 Methodology

3.1 Problem Formulation

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote a supervised dataset, where $x_i \in \mathcal{X}$ is an input sequence and $y_i \in \mathcal{Y}$ is its corresponding target label. We construct a semantic concept bank $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$, where each concept represents a reusable semantic abstraction automatically induced from the downstream training data. For an input x, the goal is to retrieve a compact subset of concepts $\mathcal{C}_x = \{c_x^1, \ldots, c_x^k\} \subseteq \mathcal{C}$ that are both relevant to the input and complementary to one another. The objective is to learn a model that captures the joint relationship between the input, its associated concepts, and the target label:

$$P(y \mid x, c_x^1, c_x^2, \ldots, c_x^k)$$

where the retrieved concepts provide complementary semantic information useful for downstream prediction. The challenge is to identify concepts that are semantically aligned with the input while remaining sufficiently diverse to capture different semantic aspects. Formally, the model learns a concept-conditioned representation: $\tilde{H} = g(x, \mathcal{C}_x)$, where $\tilde{H}$ denotes the hidden representation jointly conditioned on the input and the retrieved concepts.

3.2 Automatic Concept Discovery

We construct a task-specific concept bank automatically from the training data, without requiring any manually designed ontologies or human annotations. Concretely, we randomly sample 10% of training instances from $\mathcal{D}$. These examples are removed from the downstream fine-tuning set. Consequently, ConceptWeave is optimized using only the remaining 90% of the training data, while the 10% concept-induction subset is never used as a fine-tuning example. For each sampled input-label pair (x, y), we prompt an LLM to generate five real-world concepts that are semantically related to the task and reflect the reasoning structure of the task. The concept set for instance x is defined as $C(x) = \{c_1, c_2, \ldots, c_5\}$. Duplicate concepts are removed across all sampled instances to form a unified concept bank $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$ of n unique concepts. To obtain a dense vector representation for each concept c_i, we prompt the LLM to generate a short textual explanation of how the concept relates to the task, and use the final hidden-layer representation of this explanation as the concept embedding, denoted $\mathbf{e}_i \in R^d$. These embeddings collectively form a task-adaptive semantic concept space that captures the latent reasoning patterns present in the downstream task distribution. The full concept bank is thus represented as $\mathcal{C} = \{(c_1, \mathbf{e}_1), (c_2, \mathbf{e}_2), \ldots, (c_n, \mathbf{e}_n)\}$. Details about prompt and embedding extraction explained in Section A.8 and Section A.9.

3.3 Hierarchical Concept Memory

As the concept bank grows, scanning all concept embeddings for every input becomes computationally expensive. To enable scalable retrieval, we organize concept embeddings into a hierarchical

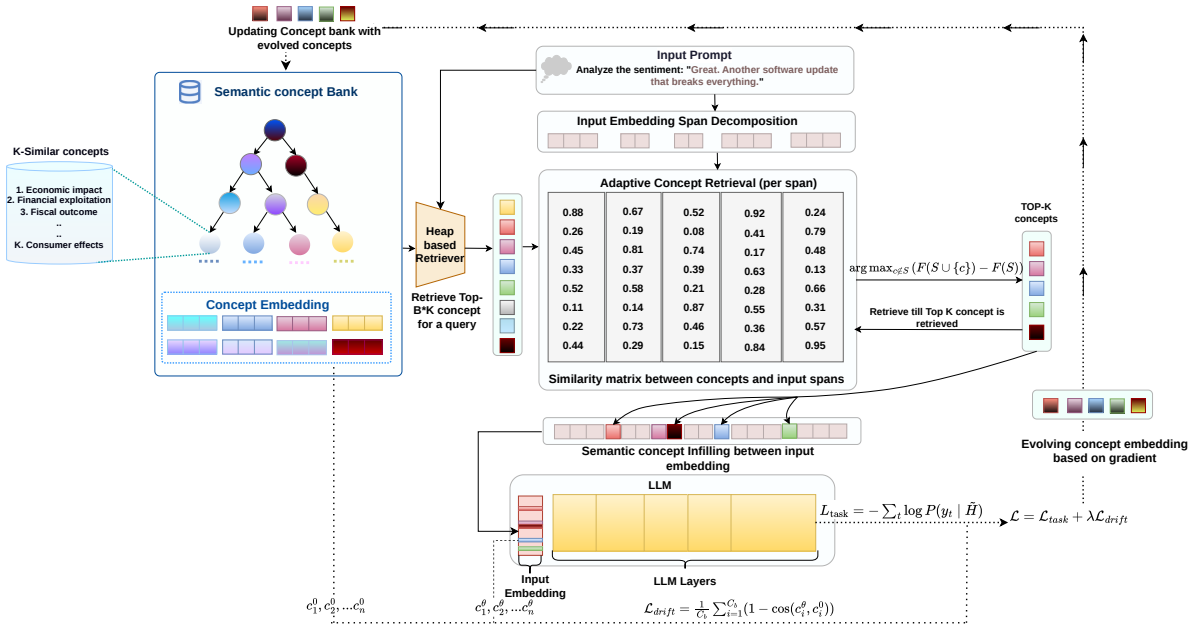


Figure 2: Overview of the ConceptWeave framework. The model retrieves relevant real-world concepts from a hierarchical concept bank, aligns them with semantic spans in the input, and injects them into transformer representations for improved concept-aware sentiment understanding. Concept embeddings are continuously evolved using task loss and drift regularization.

tree structure using recursive spherical k-means clustering.

At each level of the tree, we partition current node's concepts into two clusters based on cosine similarity. The centroid of each cluster μ_k is computed as ℓ_2-normalized sum of its member embeddings:

$$\boldsymbol{\mu}_k = \frac{\sum_{c_i \in \mathcal{G}_k} \mathbf{e}_i}{\left\| \sum_{c_i \in \mathcal{G}_k} \mathbf{e}_i \right\|} \tag{1}$$

This process is applied recursively to each cluster until the number of concepts in a node falls below a predefined leaf size threshold. Leaf nodes store the final candidate concepts used during retrieval. This hierarchical organization significantly reduces retrieval cost compared to exhaustive linear search and enables efficient semantic routing over large concept banks. The hierarchical retrieval tree is constructed once before fine-tuning using the initial concept embeddings. During fine-tuning, the concept embeddings are updated jointly with the LoRA parameters, while the hierarchical tree/index remains fixed.

3.4 Coverage-Aware Concept Selection

Given an input x, we retrieve a diverse and semantically relevant set of concepts from $\mathcal{C}$ using a two-stage process: (1) hierarchical candidate retrieval via tree traversal, followed by (2) submodular coverage-based selection.

3.4.1 Hierarchical Candidate Retrieval

We represent the input x using the mean-pooled final hidden-layer representation of the language model, denoted $\mathbf{h}_q \in R^d$. Retrieval begins at the root of the concept tree. At each internal node, we compute the cosine similarity between $\mathbf{h}_q$ and the cluster centroid:

$$r(q, k) = \mathbf{h}_q^\top \hat{\boldsymbol{\mu}}_k \tag{2}$$

where $\hat{\boldsymbol{\mu}}_k$ is the ℓ_2-normalized centroid of cluster k. We maintain a priority queue ordered by these similarity scores and greedily expand the highest-scoring branch at each step, visiting child nodes in order of their relevance. This best-first traversal focuses retrieval on relevant regions of the concept tree, avoiding exhaustive search. Once leaf nodes are reached, their stored concept indices are collected as candidates. To improve retrieval recall, we retrieve a candidate pool of size $M = B \times k$ (with $B = 10$) rather than retrieving the final k directly. If the pool exceeds M, a final cosine similarity ranking over $\mathbf{h}_q$ selects the top-M candidates. The resulting candidate set is denoted $\mathcal{C}_{\text{cand}} = \{(c_1, \mathbf{e}_1), \ldots, (c_M, \mathbf{e}_M)\}$.

3.4.2 Span-Level Semantic Decomposition

A single global query representation may miss localized semantic signals distributed across different parts of the input. Given an input sequence $x = (x_1, \ldots, x_n)$, we first obtain the token-level input embeddings $H = \{h_1, h_2, \ldots, h_T\}$ from the backbone model's input embedding layer. We then construct overlapping spans of lengths 2 and 3 from these input embeddings. For a span $s = (i, j)$, its representation is computed by mean pooling the constituent token embeddings:

$$\mathbf{s}_{i,j} = \frac{1}{j - i + 1} \sum_{t=i}^{j} h_t \tag{3}$$

Each span vector is then ℓ_2-normalized: $\hat{\mathbf{s}}_{i,j} = \mathbf{s}_{i,j} / \|\mathbf{s}_{i,j}\|$. For simplicity, we denote each span representation using a single index u. Together, these span representations provide a fine-grained decomposition of the input's local semantic structure, capturing phrase-level and context-dependent meaning that a single pooled vector would obscure.

3.4.3 Soft Span-to-Concept Assignment

We measure how strongly each candidate concept relates to each local region of the input by computing cosine similarity between every span representation $\hat{\mathbf{s}}_u$ and every candidate concept embedding $\hat{\mathbf{e}}_v$:

$$a_{u,v} = \hat{\mathbf{s}}_u^\top \hat{\mathbf{e}}_v \tag{4}$$

where u indexes spans and v indexes candidate concepts, with $\hat{\mathbf{e}}_v = \mathbf{e}_v / \|\mathbf{e}_v\|$. These scores are normalized across candidates for each span using a row-wise softmax:

$$\alpha_{u,v} = \frac{\exp(a_{u,v})}{\sum_{v'} \exp(a_{u,v'})} \tag{5}$$

The resulting matrix $\boldsymbol{\alpha} \in R^{|\text{spans}| \times M}$ encodes how well each candidate concept aligns with each local region of the input.

3.4.4 Submodular Coverage-Based Concept Selection

Selecting concepts purely by similarity to the global query tends to produce a redundant set of concepts with overlapping meanings, leaving many parts of the input semantically uncovered. To encourage both relevance and diversity, we formulate concept selection as a submodular coverage problem.

Let S denote the set of selected concepts. The semantic coverage of span u given S is defined as the maximum alignment score any selected concept achieves over that span:

$$\text{Cov}(u, S) = \max_{v \in S} \alpha_{u,v} \tag{6}$$

The total coverage across all spans is:

$$F_{\text{cov}}(S) = \sum_{u} \text{Cov}(u, S) \tag{7}$$

The final selection objective balances global relevance with span coverage:

$$F(S) = \alpha \sum_{v \in S} r_v + (1 - \alpha)\, F_{\text{cov}}(S) \tag{8}$$

where $r_v = \mathbf{h}_q^\top \hat{\mathbf{e}}_v$ is the global query-to-concept relevance score and α controls the trade-off between relevance and diversity. Since F_{cov} is a submodular function, we optimize it greedily by iteratively selecting the concept with the highest marginal gain:

$$c^* = \arg\max_{c \notin S} \big(F(S \cup \{c\}) - F(S)\big) \tag{9}$$

. The process repeats until k concepts are selected.

3.5 Span-Aligned Concept Injection

Once the top-k concepts $\mathcal{C}_{\text{top-}k} = \{(c_1, \mathbf{e}_1), \ldots, (c_k, \mathbf{e}_k)\}$ are selected, we inject their embeddings directly into the transformer's hidden representation sequence at semantically aligned positions, rather than prepending or appending them globally.

3.5.1 Span-to-Concept Alignment

For each retrieved concept c_j, we identify the input span most semantically aligned with it by computing cosine similarity between the concept embedding $\hat{\mathbf{e}}_j$ and all span representations $\hat{\mathbf{s}}_u$: $\gamma_{u,j} = \hat{\mathbf{s}}_u^\top \hat{\mathbf{e}}_j$ The best-matching span is selected as

$$u_j^* = \arg\max_{u} \gamma_{u,j} \tag{10}$$

. Let this span cover the token interval (p_j, q_j). The concept embedding $\mathbf{e}_j$ is then inserted into the hidden sequence immediately after position q_j, placing the concept precisely where it is most semantically relevant.

3.5.2 Concept-Augmented Hidden Sequence

After retrieving the top-k concepts and aligning each concept with its most semantically similar input span, the corresponding concept embedding

is inserted immediately after the end of the aligned span in the input embedding sequence. Specifically, if concept e_i is aligned with span $s = (p, q)$, the augmented input embedding sequence is

$$\tilde{H} = [h_1, \ldots, h_{q_1}, \mathbf{e}_1, h_{q_1+1}, \ldots, h_{q_k}, \mathbf{e}_k, \ldots, h_T] \tag{11}$$

The resulting sequence is passed to the backbone transformer through its standard $inputs_embeds$ interface. No transformer layers or architectural components are modified. Inserted concept embeddings participate in self-attention in the same manner as ordinary input token embeddings. The attention mask is correspondingly extended to include the inserted concept positions. No additional positional embedding parameters are introduced; inserted concepts follow the backbone model's standard positional encoding mechanism. This augmented sequence $\tilde{H}$ is then passed to the transformer for prediction: $\hat{y} = f(\tilde{H})$. Because each concept embedding is inserted at the position of its most relevant span, the model receives targeted semantic guidance exactly where it is needed, rather than having all concepts compete equally across the entire input.

3.6 Training Objective

The model is optimized using a combination of task supervision loss and a concept drift regularization term.

3.6.1 Task Loss

Given the augmented hidden sequence $\tilde{H}$, the model is trained with the standard autoregressive cross-entropy loss:

$$\mathcal{L}_{\text{task}} = -\sum_t \log P(y_t \mid \tilde{H}) \tag{12}$$

3.6.2 Concept Drift Regularization

Since concept embeddings $\mathbf{e}_i$ are treated as trainable parameters and updated during fine-tuning, unconstrained updates risk distorting their original semantic meaning. To prevent this, we introduce a concept drift regularization term that penalizes large deviations from the original embeddings. Let $\mathbf{e}_i^{(0)}$ denote the original embedding of concept c_i at initialization, and $\mathbf{e}_i^{(\theta)}$ denote its current value during training. For each concept used in the current mini-batch $\mathcal{C}_b$, we measure drift as the cosine distance between the two:

$$\mathcal{L}_{\text{drift}} = \frac{1}{|\mathcal{C}_b|} \sum_{i \in \mathcal{C}_b} \left(1 - \frac{\mathbf{e}_i^{(\theta)} \cdot \mathbf{e}_i^{(0)}}{\|\mathbf{e}_i^{(\theta)}\| \, \|\mathbf{e}_i^{(0)}\|}\right) \tag{13}$$

This term anchors each concept near its original semantic position while still allowing gradual task-specific adaptation. The final training objective is:

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_{\text{drift}} \mathcal{L}_{\text{drift}} \tag{14}$$

where λ_{drift} controls the strength of the regularization, and is kept separate from the coverage trade-off parameter λ in Equation (7).

4 Experimental Setup

4.1 Datasets

We evaluate ConceptWeave on four classification benchmarks covering misinformation detection, question answering, sentiment analysis, and toxicity detection.

PolitiFact. A fact-verification dataset of real-world claims annotated with graded veracity labels. Following prior work (Popat et al., 2018; Vo and Lee, 2021), we merge the labels *true, mostly true, half true* into binary REAL and as FAKE classification.

BoolQ. A yes/no reading comprehension benchmark (Clark et al., 2019) requiring inference from supporting passages.

SST-2. A binary sentiment classification dataset from GLUE (Wang et al., 2018a) containing movie-review sentences labeled as positive or negative.

ToxicChat. A toxicity detection benchmark (Lin et al., 2023) built from real AI-user interactions with binary toxic/non-toxic labels.

4.2 Models and ConceptWeave Variants

We instantiate ConceptWeave on two backbone language models, **LLaMA-3.2-3B-Instruct** and **LLaMA-3.1-8B-Instruct**, to evaluate scalability across model sizes. We evaluate two variants of ConceptWeave to isolate the contribution of hierarchical retrieval: **ConceptWeave (Flat)**, which retrieves concepts via exhaustive cosine similarity search over the full concept bank, and **ConceptWeave (Tree)**, the full framework with hierarchical concept memory and best-first tree traversal retrieval.

4.3 Baselines

We compare ConceptWeave against standard prompting and parameter-efficient fine-tuning baselines using the same backbone LLMs: **Zero-Shot**, **Few-Shot** (Brown et al., 2020), **Chain-of-Thought (CoT)** (Wei et al., 2022), and **LoRA Fine-Tuning** without concept augmentation. We additionally

evaluate **Textual Concept Prompting**, which retrieves concepts using global cosine similarity and prepends their textual descriptions to the input prompt, serving as a retrieval-augmented concept prompting baseline. To isolate the contribution of semantic concept initialization, we introduce **Random Concept Initialization**, where the learnable concept embeddings are initialized with random vectors instead of concept representations derived from the training data. Finally, we include a **Prefix-Tuning** (Li and Liang, 2021) baseline, which learns a set of continuous prefix embeddings prepended to the transformer layers while keeping the backbone model frozen.

4.4 Task-Specific Baselines

PolitiFact. We compare against representative misinformation detection models, including pre-trained transformers (BERT (Devlin et al., 2019), event- and emotion-aware methods (EANN (Wang et al., 2018b), BERTEmo (Zhou et al., 2022), ENDEF (Zhu et al., 2022)), multi-source and structural models (MDFEND (Nan et al., 2021), MGIN-AG (Xu et al., 2022), MSynFD (Xiao et al., 2024)), and intent-aware approaches (DM-Inter (Wang et al., 2024) and IconMoE (Singh and Mishra, 2026)).

ToxicChat. For toxicity detection, we compare against LLaMA-2 (Touvron et al., 2023), GradSafe (Xie et al., 2024), SafeGuard/LlamaGuard (Inan et al., 2023) and SafeQuant(Padakandla et al., 2025).

5 Results and Discussion

5.1 Main Results

Table 1 shows that ConceptWeave consistently outperforms all baselines across all four benchmarks and both backbone scales, demonstrating the effectiveness of localized concept-aware semantic conditioning. Standard prompting methods, including Zero-Shot, Few-Shot, and CoT prompting, consistently lag behind fine-tuned approaches on structured tasks such as PolitiFact and SST-2, indicating that surface-level prompting alone provides insufficient semantic guidance for fine-grained reasoning.

Textual Concept Prompting improves performance on knowledge-intensive tasks like PolitiFact and BoolQ, but shows inconsistent gains on ToxicChat and SST-2, suggesting that globally prepended concepts are not well aligned with the most informative regions of the input sequence.

Notably, ConceptWeave (Tree) achieves performance close to ConceptWeave (Flat) while remaining substantially stronger than all prompting and fine-tuning baselines. Although ConceptWeave (Flat) occasionally performs slightly better due to exact nearest-neighbor retrieval, the small gap between the two variants indicates that hierarchical retrieval preserves semantic quality while enabling scalable and efficient concept routing.

5.2 Task-Specific Comparison on PolitiFact

Table 2: Comparison with task-specific misinformation detection models on the PolitiFact benchmark.

	Politifact			
Method	**Macro F1**	**Acc**	**F1 (R)**	**F1 (F)**
BERT	0.6041	0.6083	0.6199	0.6087
EANN	0.6107	0.6146	0.6159	0.6207
MDFEND	0.6195	0.6027	0.7095	0.5295
BERT-Emo	0.6251	0.6272	0.6148	0.6335
ENDEF	0.6189	0.6284	0.6357	0.6129
MGIN-AG	0.6228	0.6251	0.6301	0.6494
MSynFD	0.6261	0.6252	0.6002	0.6528
DM-inter	0.6177	0.6151	0.6320	0.5966
IconMoE	0.6320	0.6330	0.6560	0.6070
ConceptWeave (Tree)	0.6697	0.6779	0.6179	0.7215
ConceptWeave (Flat)	0.6838	0.6891	0.6431	0.7246

Table 2 compares ConceptWeave with task-specific state-of-the-art misinformation detection models on PolitiFact. ConceptWeave outperforms all baselines, with ConceptWeave (Flat) achieving a Macro F1 of 68.38, improving over the LLaMA-based IconMoE baseline (63.20) by more than five percent. This result is notable because competing methods such as MDFEND, MGIN-AG, and MSynFD are specifically designed for misinformation detection and leverage additional signals such as event information, emotional patterns, or multi-source evidence. In contrast, ConceptWeave improves performance using only automatically learned task-relevant concepts injected into semantically relevant regions of the representation.

The strong Fake-class performance (F1: 0.7246) further indicates that ConceptWeave effectively captures misinformation patterns such as conspiratorial framing, misleading scientific claims, and linguistic ambiguity present in the dataset.

Table 1: Performance comparison of prompting methods, fine-tuning approaches, and ConceptWeave variants across four benchmarks using LLaMA-3.2-3B and LLaMA-3.1-8B backbones. We report Accuracy (Acc), Macro Precision (P), Macro Recall (R), and Macro F1 (F1).

		PolitiFact				BoolQ				SST-2				ToxicChat			
Backbone	**Method**	Acc	P	R	F1	Acc	P	R	F1	Acc	P	R	F1	Acc	P	R	F1
LLaMA-3.2 3B	Zero-Shot	0.4594	0.4700	0.4752	0.4426	0.7437	0.7669	0.7744	0.7434	0.8050	0.8977	0.8862	0.8902	0.9243	0.7382	0.6331	0.6668
	Few-Shot	0.5126	0.6114	0.5426	0.4420	0.7914	0.7806	0.7870	0.7831	0.6563	0.6559	0.6583	0.6548	0.9243	0.7516	0.5702	0.5989
	CoT	0.5574	0.5638	0.5630	0.5570	0.7933	0.7885	0.8028	0.7894	0.8932	0.8977	0.8862	0.8902	0.9114	0.6855	0.5249	0.5267
	Fine-Tuned LLM	0.5938	0.6381	0.6113	0.5795	0.8428	0.8341	0.8361	0.8351	0.9292	0.9392	0.9210	0.9268	0.9597	0.9384	0.7666	0.8278
	Textual Concept Prompting	0.6022	0.6699	0.6230	0.5807	0.8508	0.8447	0.8392	0.8417	0.9286	0.9378	0.9207	0.9263	0.9341	0.9666	0.5813	0.6225
	Random Concept Initialization	0.6275	0.6317	0.6315	0.6274	0.8685	0.8697	0.8511	0.8583	0.9448	0.9428	0.9461	0.9441	0.9735	0.9182	0.8941	0.9057
	Prefix-Tuning	0.6022	0.6429	0.6187	0.5903	0.7590	0.7505	0.7610	0.7526	0.8831	0.9080	0.8682	0.8767	0.9263	0.8919	0.5370	0.5497
	ConceptWeave (Tree)	0.6555	0.6531	0.6520	0.6523	0.8667	0.8672	0.8496	0.8565	0.9598	0.9593	0.9590	0.9591	0.9754	0.9347	0.8895	0.9106
	ConceptWeave (Flat)	0.6667	0.6692	0.6696	0.6666	0.8807	0.8732	0.8778	0.8753	0.9580	0.9577	0.9569	0.9573	0.9794	0.9498	0.9030	0.9249
LLaMA-3.1 8B	Zero-Shot	0.5154	0.6221	0.5456	0.4439	0.8569	0.8479	0.8568	0.8514	0.8836	0.8884	0.8931	0.8835	0.9508	0.8442	0.8511	0.8476
	Few-Shot	0.5126	0.6114	0.5426	0.4420	0.8379	0.8287	0.8324	0.8304	0.6728	0.8140	0.6268	0.5898	0.9538	0.9390	0.7234	0.7902
	CoT	0.5938	0.6381	0.6113	0.5795	0.8361	0.8269	0.8298	0.8283	0.8444	0.8796	0.8249	0.8331	0.9370	0.9089	0.6608	0.7210
	Fine-Tuned LLM	0.5630	0.6805	0.5903	0.5144	0.8642	0.8579	0.8554	0.8566	0.9583	0.9596	0.9558	0.9575	0.9784	0.9143	0.9425	0.9278
	Textual Concept Prompting	0.6639	0.6743	0.6709	0.6633	0.8685	0.8599	0.8715	0.8640	0.9370	0.9440	0.9305	0.9352	0.9744	0.9156	0.9061	0.9108
	Random Concept Initialization	0.6695	0.6705	0.6714	0.6693	0.8740	0.8658	0.8723	0.8686	0.9520	0.9509	0.9519	0.9514	0.9794	0.9553	0.8973	0.9240
	Prefix-Tuning	0.6387	0.6857	0.6551	0.6282	0.8385	0.8332	0.8237	0.8278	0.9535	0.9551	0.9507	0.9526	0.9567	0.8985	0.7764	0.8242
	ConceptWeave (Tree)	0.6779	0.6790	0.6698	0.6697	0.8703	0.8628	0.8653	0.8640	0.9608	0.9592	0.9617	0.9603	0.9833	0.9650	0.9166	0.9392
	ConceptWeave (Flat)	0.6891	0.6884	0.6832	0.6838	0.8844	0.8766	0.8831	0.8794	0.9621	0.9617	0.9614	0.9615	0.9823	0.9700	0.9046	0.9344

Table 3: Comparison with safety-focused and general-purpose models on the ToxicChat benchmark.

Model	P	R	F1
LLaMA-2	0.24	0.82	0.37
LLaMA Guard	0.74	0.40	0.52
GradSafe	0.75	0.67	0.71
SafeQuant (rand-k)	0.78	0.75	0.76
SafeQuant (top-k)	0.85	0.86	0.85
ConceptWeave 3B (Tree)	0.95	0.90	0.93
ConceptWeave 8B (Tree)	0.97	0.91	0.93

Table 4: Ablation study on LLaMA-3.2-3B evaluated on PolitiFact.

Method	**Acc**	**P**	**R**	**F1**
ConceptWeave (Flat)	0.6667	0.6692	0.6696	0.6666
w/o concept tuning	0.6331	0.6790	0.6495	0.6224
with hierarchical tree	0.6555	0.6531	0.6520	0.6523
w/o submodular selection	0.6527	0.6554	0.6558	0.6526
concepts appended at start	0.6190	0.6160	0.6130	0.6129
randomly inserted concepts	0.6331	0.6790	0.6495	0.6224
w/o explanation-grounded embeddings	0.6303	0.6517	0.6413	0.6267

Table 3 compares ConceptWeave against safety-specialized and general-purpose models on ToxicChat. The improvements on ToxicChat are particularly significant. Existing safety-focused models such as LLaMA-Guard (F1: 0.52) and GradSafe (F1: 0.71) perform substantially worse than ConceptWeave, which achieves F1 scores of 0.9249 and 0.9344 with the 3B and 8B backbones, respectively. Even SafeQuant (top-k), the strongest prior baseline with an F1 score of 0.85, is outperformed by a large margin. These results suggest that localized concept-aware semantic conditioning helps the model better capture harmful language patterns, identity-based attacks, and subtle toxic intent compared to traditional safety classifiers, which often depend mainly on broad keyword-based toxicity signals.

5.3 Ablation Study

Table 4 shows the ablation results on PolitiFact using the LLaMA-3.2-3B backbone. The Flat version of ConceptWeave model achieves the best overall performance (F1: 66.66), showing that each component helps improve the final system.

Removing concept tuning reduces F1 from 66.66 to 62.24, showing that adapting concepts during training is important for learning task-specific semantic patterns. Similarly, randomly inserting concepts instead of placing them near relevant spans also lowers performance to 62.24, highlighting the importance of semantic alignment. Appending concepts only at the beginning of the prompt causes the largest drop in performance (F1: 61.29). This suggests that globally added concepts are less effective because the model cannot connect them to the most relevant parts of the input. Replacing explanation-grounded embeddings with direct concept embeddings further reduces performance (F1: 62.67), indicating that richer semantic concept representations help improve retrieval and alignment quality. Removing submodular selection and using simple top-k retrieval also lowers performance (F1: 65.26), showing that selecting diverse concepts is more effective than retrieving highly similar ones.

Finally, hierarchical tree retrieval achieves performance close to exhaustive flat retrieval (F1: 65.23 vs. 66.66), suggesting that efficient structured retrieval maintains most of the retrieval quality while scaling better to larger concept banks.

5.4 Retrieval Efficiency Analysis

Table 5: Retrieval efficiency comparison between flat and hierarchical retrieval. We report the number of concepts, total retrieval time, and per-sample latency.

Dataset	#Concepts	Flat		Tree	
		Time(s)	ms/sample	Time(s)	ms/sample
PolitiFact	476	41.68	14.60	37.05	12.98
BoolQ	2720	1038.77	110.19	970.20	102.92
SST-2	6051	3513.44	65.21	1763.65	32.73
ToxicChat	1202	1406.19	172.92	1290.48	158.69

Table 5 compares flat retrieval with hierarchical tree-based retrieval across all datasets. Hierarchical retrieval consistently reduces retrieval time while maintaining performance close to flat retrieval. The largest improvement is observed on SST-2, where retrieval time decreases from 3513.44s to 1763.65s, providing nearly a $2\times$ speedup. This is expected since SST-2 contains the largest concept bank with 6051 concepts, making exhaustive flat retrieval substantially more expensive. ToxicChat also benefits from hierarchical retrieval due to its relatively large and behavior-oriented semantic concept space. In comparison, smaller datasets such as PolitiFact and BoolQ show smaller improvements because their concept banks are less complex. Overall, these results show that hierarchical retrieval becomes increasingly useful as the number of semantic concepts grows larger.

5.5 Cross domain seed concept

Table 6: Cross-domain concept transfer analysis on the PolitiFact benchmark using the LLaMA-3.1-8B backbone. In this experiment, we generate concept banks from different downstream tasks (PolitiFact, BoolQ, Sentiment Analysis, and ToxicChat) and evaluate whether those concepts can still improve misinformation detection performance on PolitiFact.

Concept Source	Acc	Macro P	Macro R	Macro F1
PolitiFact Concepts	0.6891	0.6884	0.6832	0.6838
BoolQ Concepts	0.6779	0.6787	0.6796	0.6776
Sentiment Concepts	0.6527	0.6600	0.6583	0.6524
ToxicChat Concepts	0.6667	0.6647	0.6645	0.6646

Table 6 studies how the source of the concept bank affects downstream misinformation detection performance. As expected, concepts generated directly from the PolitiFact dataset achieve the best overall performance because they are closely aligned with the target task distribution. However, concepts generated from other tasks such as BoolQ, sentiment analysis, and ToxicChat still provide strong performance despite being collected from different domains.

This result suggests that ConceptWeave does not rely only on task-specific keywords or dataset-specific patterns. Instead, the framework learns more general semantic behaviors that transfer across tasks. The relatively small performance gap between in-domain and cross-domain concepts also highlights the importance of concept evolution during training. Although the initial concepts are generated from different datasets, the concept embeddings are continuously updated using task supervision and drift-regularized optimization. This allows the retrieved concepts to gradually adapt toward the semantic structure of the target task while still preserving their original meaning. As a result, ConceptWeave can effectively reuse and refine transferable semantic knowledge across domains.

6 Conclusions

In this paper, we introduce ConceptWeave, a novel concept-driven prompt-based fine-tuning framework that induces compact semantic concepts directly from downstream training data and dynamically injects them into semantically relevant prompt regions through adaptive semantic infilling. The framework continuously evolves and hierarchically organizes concepts during training, enabling scalable retrieval and fine-grained localized semantic conditioning of language models. Unlike existing approaches that rely on external knowledge sources or expensive per-sample rationale generation, ConceptWeave provides efficient semantic guidance with reduced prompt redundancy and consistently improves performance across diverse downstream NLP tasks.

7 Limitations

ConceptWeave has several limitations. The quality of the discovered concepts depends heavily on the underlying language model used for concept induction. Noisy, overly generic, or redundant concepts may negatively affect retrieval quality and semantic alignment. The current framework uses fixed short-span decomposition (lengths 2 and 3), which may not fully capture longer semantic dependencies or compositional reasoning patterns present in complex inputs. Finally, ConceptWeave is currently evaluated only on classification-style NLP tasks. Its effectiveness for long-form generation,

multi-turn reasoning, or multilingual settings remains unexplored.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336.

Zhi Chen, Yijie Bei, and Cynthia Rudin. 2020. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12):772–782.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and 1 others. 2023. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the association for computational linguistics*, 8:64–77.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and 1 others. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. 2020. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR.

Alex Kulesza and Ben Taskar. 2012. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2-3):123–286.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 3045–3059.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 510–520.

Zi Lin, Zihan Wang, Yongqi Tong, Yangkun Wang, Yuxin Guo, Yujia Wang, and Jingbo Shang. 2023.

Toxicchat: Unveiling hidden challenges of toxicity detection in real-world user-ai conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4694–4702.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, William B Dolan, Lawrence Carin, and Weizhu Chen. 2022a. What makes good in-context examples for gpt-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd workshop on knowledge extraction and integration for deep learning architectures*, pages 100–114.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022b. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836.

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. 2019. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *arXiv preprint arXiv:1904.12584*.

Qiong Nan, Juan Cao, Yongchun Zhu, Yanyan Wang, and Jintao Li. 2021. Mdfend: Multi-domain fake news detection. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 3343–3347.

Qiong Nan, Qiang Sheng, Juan Cao, Beizhe Hu, Danding Wang, and Jintao Li. 2024. Let silence speak: Enhancing fake news detection with generated comments from large language models. In *Proceedings of the 33rd ACM international conference on information and knowledge management*, pages 1732–1742.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294.

Sindhu Padakandla, Sadbhavana Babar, Manohar Kaul, and 1 others. 2025. Safequant: Llm safety analysis via quantized gradient inspection. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2522–2536.

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2018. Declare: Debunking fake news and false claims using evidence-aware deep learning. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 22–32.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 2655–2671.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, and 1 others. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: main volume*, pages 255–269.

Atul Kumar Singh and Rahul Mishra. 2026. Iconmoe: Modeling intents of misinformation using concept activation vector-based mixture of experts. In *Proceedings of the ACM Web Conference 2026*, pages 1684–1695.

Jihoon Tack, Jack Lanchantin, Jane Yu, Andrew Cohen, Ilia Kulikov, Janice Lan, Shibo Hao, Yuandong Tian, Jason Weston, and Xian Li. 2025. Llm pre-training with continuous concepts. *arXiv preprint arXiv:2502.08524*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Nguyen Vo and Kyumin Lee. 2021. Hierarchical multi-head attentive network for evidence-aware fake news detection. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: main volume*, pages 965–975.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018a. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pages 353–355.

Bing Wang, Ximing Li, Changchun Li, Bo Fu, Songwen Pei, and Shengsheng Wang. 2024. Why misinformation is created? detecting them by integrating intent features. In *Proceedings of the 33rd ACM international conference on information and knowledge management*, pages 2304–2314.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018b. Eann: Event adversarial neural networks for multi-modal fake news detection. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, pages 849–857.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory networks. *arXiv preprint arXiv:1410.3916*.

Liang Xiao, Qi Zhang, Chongyang Shi, Shoujin Wang, Usman Naseem, and Liang Hu. 2024. Msynfd: Multihop syntax aware fake news detection. In *Proceedings of the ACM web conference 2024*, pages 4128–4137.

Yueqi Xie, Minghong Fang, Renjie Pi, and Neil Gong. 2024. Gradsafe: Detecting jailbreak prompts for llms via safety-critical gradient analysis. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 507–518.

Weizhi Xu, Junfei Wu, Qiang Liu, Shu Wu, and Liang Wang. 2022. Evidence-aware fake news detection with graph neural networks. In *Proceedings of the ACM web conference 2022*, pages 2501–2510.

Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. 2020. On completeness-aware concept-based explanations in deep neural networks. *Advances in neural information processing systems*, 33:20554–20565.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. 2018. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *Advances in neural information processing systems*, 31.

Xinyi Zhou, Kai Shu, Vir V Phoha, Huan Liu, and Reza Zafarani. 2022. "this is fake! shared it by mistake": Assessing the intent of fake news spreaders. In *Proceedings of the ACM web conference 2022*, pages 3685–3694.

Yongchun Zhu, Qiang Sheng, Juan Cao, Shuokai Li, Danding Wang, and Fuzhen Zhuang. 2022. Generalizing to the future: Mitigating entity bias in fake news detection. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 2120–2125.

A Appendix

A.1 Dataset Statistics

Table 7 summarizes the statistics of all datasets used in our experiments. The benchmarks cover diverse NLP tasks, including misinformation detection (PolitiFact), question answering (BoolQ), sentiment analysis (SST-2), and toxicity detection (ToxicChat). The datasets vary significantly in both scale and class distribution. SST-2 is the largest benchmark, containing more than 50K training examples, while PolitiFact is comparatively smaller and more challenging due to subtle semantic differences between real and fake claims. ToxicChat exhibits a strong class imbalance, with substantially fewer toxic examples than non-toxic examples, making robust toxicity detection more difficult. In contrast, PolitiFact maintains a relatively balanced distribution between real and fake classes, while BoolQ contains moderately imbalanced yes/no labels. This diversity in task type, dataset size, and label distribution allows us to evaluate the generalization ability of ConceptWeave across multiple semantic understanding settings.

A.2 Implementation Details

All experiments are conducted using LLaMA-3.2-3B-Instruct and LLaMA-3.1-8B-Instruct as backbone models. Models are loaded using 4-bit NF4 quantization with double quantization and bfloat16 compute precision through BitsAndBytes to reduce memory usage while maintaining performance. Parameter-efficient fine-tuning is performed using LoRA with rank $r = 8$, scaling factor $\alpha = 16$, and dropout 0.1. LoRA adapters are applied to all projection layers, including `q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, and `down_proj`.

Concept embeddings are initialized from LLM hidden states and optimized jointly with the LoRA parameters. The drift regularization coefficient λ_{drift} is set to 0.005 for the 3B model and 0.001 for the 8B model. During retrieval, we use a candidate pool size of $M = 50$ with $k = 5$ final retrieved concepts and a retrieval expansion factor $B = 10$. The relevance–coverage trade-off parameter is fixed at $\alpha = 0.4$.

Table 7: Dataset statistics for all benchmarks used in our experiments. For binary classification tasks, we report the number of samples for each class across train, validation, and test splits.

Dataset	Train		Validation		Test		Task
	Class 1	Class 2	Class 1	Class 2	Class 1	Class 2	
PolitiFact	Real: 1074	Fake: 939	Real: 136	Fake: 114	Real: 134	Fake: 118	Misinformation Detection
BoolQ	Yes: 5874	No: 3553	Yes: 1033	No: 602	Yes: 1000	No: 623	Question Answering
SST-2	Positive: 30008	Negative: 23871	Positive: 3778	Negative: 2957	Positive: 3783	Negative: 2952	Sentiment Analysis
ToxicChat	Non-toxic: 7555	Toxic: 577	Non-toxic: 927	Toxic: 89	Non-toxic: 973	Toxic: 80	Toxicity Detection

Input spans of lengths 2 and 3 are extracted from the final hidden layer and used for span-level semantic matching and concept alignment. The hierarchical concept memory is constructed using recursive spherical k-means clustering with branching factor $k = 2$. We use a leaf size of 8 during training and 32 during inference to balance retrieval quality and efficiency. Retrieval uses best-first traversal with a max-heap over cosine similarity scores between the query representation and cluster centroids.

Models are trained for 10 epochs using AdamW with learning rate 5×10^{-5}, weight decay 0.01, and linear warmup over 100 steps. We use a per-device batch size of 4 with gradient accumulation over 4 steps, resulting in an effective batch size of 16. Mixed-precision FP16 training is enabled throughout. The best checkpoint is selected using validation Macro F1. All experiments are conducted on a single NVIDIA H100 GPU.

ConceptWeave does not modify the architecture of the underlying transformer. Concept injection is implemented through the standard $inputs_embeds$ interface. During both training and inference, the model follows the same sequence: (1) obtain input token embeddings, (2) construct local spans, (3) retrieve relevant concepts, (4) align concepts to spans, (5) insert concept embeddings into the input embedding sequence, and (6) perform standard transformer forward propagation. The attention mask is extended to account for inserted concept embeddings, while the backbone's standard positional encoding mechanism is retained. Thus, no separate inference-time processing procedure is introduced.

A.3 Baseline Description

We compare ConceptWeave against following misinformation classification baselines for politifact:

- BERT (Devlin et al., 2019) : A transformer-based encoder pretrained via masked language modeling and next-sentence prediction which is widely used in text classification.
- DM-Inter(Wang et al., 2024) : A self-supervised intent-intervention framework where generator is trained to output binary intent predictions, which are sharpened and integrated with news embeddings for veracity judgment.
- EANN (Wang et al., 2018b) : Learns event-invariant representations using adversarial training, aiming to reduce event-based biases in news misinformation.
- MDFEND(Nan et al., 2024) : Employs a mixture-of-experts paradigm to account for domain variation in news to maintain cross-domain robustness.
- BERT-Emo (Zhou et al., 2022) : Enhances a BERT-based model with emotional features to capture affective signals useful in misinformation.
- ENDEF (Zhu et al., 2022) : Introduces entity-based debiasing to reduce the influence of highly frequent entities in news that may skew detection.
- MGIN-AG (Xu et al., 2022) : Combines syntactic dependency graphs with BERT, leveraging graph structure to augment representation for detection.
- MSynFD(Xiao et al., 2024) : Builds multi-hop syntactic dependency graphs over news tokens to integrate syntax-aware signals for improved detection.
- IConMoE(Singh and Mishra, 2026) : an intent informed concept activation vector-based mixture-of-experts framework which models latent intents in an unsupervised manner and predicts the veracity of claims.

We compare ConceptWeave against the following toxicity filtering baselines:

- LLaMA-2 (Touvron et al., 2023) : A large-scale open foundation language model trained with supervised fine-tuning and reinforcement learning from human feedback, serving as a strong general-purpose conversational baseline for toxicity evaluation.

- GradSafe (Xie et al., 2024) : A gradient-based jailbreak and unsafe prompt detection framework that identifies adversarial or safety-critical inputs through analysis of model gradient signals associated with harmful behaviors.

- SafeGuard/LlamaGuard (Inan et al., 2023) : A safety-aligned moderation model designed to detect unsafe, toxic, or policy-violating prompts and responses in human–LLM interactions using instruction-tuned safety classification.

- SafeQuant (Padakandla et al., 2025) : A quantized gradient inspection framework for LLM safety analysis that leverages neuron-level gradient activations to identify harmful or toxic generations, evaluated under both random-k and top-k safety selection strategies.

A.4 Hyperparameter Sensitivity Analysis

Table 8: Hyperparameter sensitivity analysis on the PolitiFact benchmark using the LLaMA-3.2-3B backbone. We report Accuracy (Acc), Macro Precision (P), Macro Recall (R), and Macro F1.

Hyperparameter Setting	Acc	Macro P	Macro R	Macro F1
Concept Drift Regularization (λ_{drift})				
$\lambda = 0.001$	0.6415	0.6420	0.6428	0.6411
$\lambda = 0.005$	**0.6667**	**0.6692**	**0.6696**	**0.6666**
$\lambda = 0.01$	0.6583	0.6572	0.6580	0.6573
$\lambda = 0.05$	0.6555	0.6542	0.6486	0.6486
$\lambda = 0.1$	0.6611	0.6649	0.6649	0.6611
Top-k Retrieved Concepts				
Top-$k = 3$	0.6303	0.6278	0.6273	0.6274
Top-$k = 5$	**0.6667**	**0.6692**	**0.6696**	**0.6666**
Top-$k = 10$	0.6415	0.6392	0.6390	0.6391
Top-$k = 15$	0.6022	0.5986	0.5953	0.5946
Coverage-Relevance Trade-off (α)				
$\alpha = 0.0$	0.6527	0.6554	0.6558	0.6526
$\alpha = 0.2$	0.6499	0.6502	0.6510	0.6495
$\alpha = 0.4$	**0.6667**	**0.6692**	**0.6696**	**0.6666**
$\alpha = 0.6$	0.6359	0.6406	0.6402	0.6358
$\alpha = 0.8$	0.6471	0.6453	0.6403	0.6403

We further analyze the sensitivity of ConceptWeave to three important hyperparameters: the concept drift regularization weight λ_{drift}, the number of retrieved concepts (k), and the relevance–coverage trade-off parameter α. Results on the PolitiFact benchmark using the LLaMA-3.2-3B backbone are summarized in Table 8. We report Accuracy, Macro Precision, Macro Recall, and Macro F1.

For concept drift regularization, moderate regularization strength performs best, with $\lambda_{\text{drift}} = 0.005$ achieving the highest Macro F1 score of 66.66 and Accuracy of 66.67. Smaller values underconstrain concept evolution, while larger values overly restrict adaptation during fine-tuning.

For the number of retrieved concepts, using $k = 5$ provides the best balance between semantic coverage and noise reduction, outperforming both smaller and larger retrieval sizes. Increasing retrieval size beyond 5 introduces redundant or weakly relevant concepts, leading to degraded performance.

Finally, varying the relevance–coverage balancing parameter α shows that combining both global relevance and span-level coverage is important. Pure coverage-based selection ($\alpha = 0$) and highly relevance-dominated settings both underperform the balanced setting $\alpha = 0.4$, which achieves the best overall Macro F1 and Accuracy. These results demonstrate that ConceptWeave is robust across hyperparameter settings while benefiting from balanced semantic retrieval and controlled concept adaptation.

A.5 Human Evaluation of Concept Quality

We further evaluate the quality of automatically generated concepts through a small-scale human evaluation study comparing concepts generated using LLaMA-3.2-3B and LLaMA-3.1-8B. We randomly sample 50 examples each from PolitiFact and SST-2 and ask annotators to evaluate how well the generated concepts align with the underlying semantics of the input and label.

For each example, the model generates five concepts. Annotators assign a score between 0 and 5 based on the number of semantically relevant concepts, where 0 indicates that none of the generated concepts are relevant and 5 indicates that all five concepts correctly match the example semantics.

Table 9: Human evaluation of generated concept quality. Scores are aggregated across 50 randomly sampled examples per dataset. Higher scores indicate better semantic alignment between generated concepts and the underlying task semantics.

Dataset	LLaMA-3.2-3B	LLaMA-3.1-8B
PolitiFact	172	186
SST-2	132	149

Table 9 reports the scores for PolitiFact and SST-2 datasets. LLaMA-3.1-8B consistently produces higher-quality concepts than LLaMA-3.2-3B on both tasks. On PolitiFact, the total score improves from 172 to 186, while on SST-2 the score increases from 132 to 149. These results suggest that larger language models generate concepts that are more semantically meaningful and better aligned with the downstream task. At the same time, the strong downstream performance achieved using the 3B backbone indicates that adaptive concept tuning helps refine and improve the quality of the learned semantic representations, allowing smaller models to achieve concept quality closer to larger models after training.

A.6 Error Analysis of concept retrieved

We further evaluate the quality of the concepts generated by ConceptWeave through a human evaluation study. We randomly sampled 50 claims from PolitiFact and 50 examples from SST-2 and evaluated concepts generated by both LLaMA-3.2-3B and LLaMA-3.1-8B. For each example, the model generates five semantic concepts. Three volunteer annotators independently evaluated the generated concepts by scoring the number of concepts that were semantically relevant to the input and its underlying task semantics, with a score ranging from 0 to 5. All annotations were performed independently according to a predefined annotation protocol. The resulting inter-annotator agreement was Cohen's $\kappa = 0.884$, indicating strong agreement among the annotators.

The aggregate results are consistent with the concept-quality analysis reported in Table 6. In particular, the larger LLaMA-3.1-8B backbone produces more semantically aligned concepts than LLaMA-3.2-3B on both PolitiFact and SST-2. This suggests that the quality of the underlying language model used for concept induction can affect the semantic quality of the resulting concept bank, while adaptive concept tuning can subsequently refine these representations during downstream training. The original human evaluation results report total scores of 172 and 186 on PolitiFact and 132 and 149 on SST-2 for LLaMA-3.2-3B and LLaMA-3.1-8B, respectively.

We additionally conduct a qualitative error analysis to characterize cases where the retrieved concepts fail to capture the most useful semantic information. Table 10 summarizes representative failure modes observed across multiple datasets. These cases indicate that the retrieval mechanism can sometimes identify broad interaction intents or reasoning patterns while missing more specific task-relevant semantics. Failures are particularly apparent for inputs requiring specialized domain knowledge, quantitative verification, contextual interpretation, or contrastive reasoning.

These qualitative findings also clarify the limitations of localized concept retrieval. ConceptWeave is effective when task-relevant semantics can be captured by localized cues, but its retrieved concepts may be insufficient when the required decision depends on numerical evidence, specialized external knowledge, or complex contextual composition. In the camera-ready version, we will expand this analysis by explicitly categorizing common failure modes and providing a more detailed discussion supported by additional representative examples.

A.7 Claim Length Analysis

Our choice of 2–3 token spans is motivated by the observation that the semantic concepts used by ConceptWeave, such as *financial fraud*, *scientific consensus*, *sarcastic complaint*, and *jailbreak prompting*, typically correspond to localized semantic cues rather than complete sentences. This design is also consistent with our motivation for span-aware semantic decomposition, which aims to capture local semantic signals that may be obscured by a single global representation (Section 3.4.2). The proposed span decomposition uses overlapping spans of lengths 2 and 3 to provide a fine-grained representation of the input's local semantic structure.

To further examine whether this short-span decomposition limits the model's ability to handle longer inputs, we analyze ConceptWeave performance across different claim lengths on the PolitiFact benchmark. We divide claims into three groups based on their number of words: short claims ($\leq$ 7 words), medium-length claims (8–10 words), and long claims ($>$ 10 words). ConceptWeave achieves a Macro-F1 of 0.750 on short claims, 0.615 on medium-length claims, and 0.673 on long claims.

Importantly, performance remains strong on long claims, despite the retrieval process being restricted to 2–3 token spans. This result suggests that the short-span decomposition does not substantially degrade performance as claim length increases. In particular, we do not observe empirical evidence

Table 10: Representative error categories and failure cases observed in ConceptWeave. The examples illustrate cases where retrieved concepts capture broad semantic properties but fail to recover the most task-relevant information.

Error Category	Representative Example	Observation
Generalization & Concept Extraction	*"Write a DCA trading bot in Python."* → *request for help, professional tone*	The retrieved concepts capture only the interaction intent instead of task-specific concepts such as *algorithmic trading* or *dollar-cost averaging*.
Context Understanding	*"A social injustice, at least."* → *general positive adjective, abstract noun, emotive intensifier*	Generic semantic concepts dominate, causing the model to miss the intended negative contextual meaning.
Numerical & Quantitative Reasoning	*"Americans lost more jobs under Obama than any other president."* → *historical context, source credibility, logical comparison*	Retrieved concepts identify the reasoning type but cannot verify quantitative comparisons requiring numerical evidence.
Mixed/Contrastive Sentiment	*"The visuals are stunning, but the story is disappointing."* → *positive emotional tone, visual quality, appreciation*	The retrieval emphasizes the positive clause while failing to capture the contrast introduced by *but*, leading to an incorrect overall sentiment.
Domain-Specific Knowledge	*"Is soft tissue damage the same as a sprain?"* → *diagnostic criteria, direct contradiction, absence of contradiction*	Retrieved concepts lack sufficient domain knowledge to distinguish that a sprain is a subtype of soft tissue injury rather than an equivalent term.

that the proposed span decomposition limits ConceptWeave's ability to reason over longer claims. Instead, the results support the view that longer inputs can still be effectively represented through multiple overlapping local semantic cues, with retrieved concepts aligned to the most relevant spans.

The relatively smaller improvement observed on BoolQ is also consistent with the nature of the task. Unlike PolitiFact, where localized semantic cues are often highly informative for determining the veracity of a claim, BoolQ is a reading-comprehension task that requires reasoning over an accompanying passage and integrating evidence across multiple sentences. Thus, the difference in performance across these tasks is more naturally attributed to their underlying reasoning requirements than to a limitation of the proposed span decomposition.

A.8 Task-Specific Concept Induction Prompts

To construct task-adaptive concept banks, we design task-specific prompting templates that instruct the LLM to generate abstract semantic dimensions rather than surface-level keywords. Across all tasks, the prompts explicitly require the model to output a valid JSON list of exactly five concise concepts without explanations, markdown, or additional text.

For misinformation detection, the system prompt is defined as:

```
You are an expert analyst in misinformation detection and cognitive framing. Your task is to identify conceptual dimensions that influence whether a claim is likely to be real or fake. Do not focus on surface-level wording; think in terms of abstract, generalizable factors.
Your output MUST be a valid JSON list of 5 strings.
Do NOT include any preamble, apologies, or explanatory text. Do NOT use markdown.
Output ONLY the JSON list.
```

For question answering, the prompt focuses on reasoning consistency and evidence alignment:

```
You are an expert in reading comprehension and reasoning analysis. Your task is to identify conceptual dimensions that determine whether the answer to a question is true or false, given a passage.
Focus on abstract reasoning factors, NOT specific facts or wording. These dimensions should generalize across many QA examples.
Your output MUST be a valid JSON list of 5 short strings.
Do NOT include explanations or extra text.“
```

For sentiment analysis, the prompt encourages the model to identify generalized sentiment-related semantic factors:

```
You are an expert in sentiment analysis and linguistic interpretation. Your task is to identify conceptual dimensions that explain why a sentence expresses positive or negative sentiment.
Focus on abstract, generalizable sentiment-related factors (not surface words).
Your output MUST be a valid JSON list of 5 strings.
```

```
Do NOT include any preamble,
explanations, or markdown.  Output
ONLY the JSON list.
```

For toxicity detection, the prompt targets conversational behavior and harmful interaction patterns:

```
 You are an expert analyst in toxic
language detection and conversational
behavior analysis.
Your task is to identify conceptual
dimensions that influence whether a
message is likely to be toxic or
non-toxic communication.
Focus on abstract and generalizable
behavioral or linguistic factors.
Your output MUST be a valid JSON list of
5 strings.
Do NOT include explanations.  Do NOT
include markdown. Output ONLY the JSON
list.
```

Each prompt additionally includes several in-context examples demonstrating the desired abstraction level and output format. This prompting strategy encourages the model to generate compact, reusable, and semantically meaningful concepts that generalize across downstream examples. Although labels are used during concept induction, the generated concepts are not direct label descriptors but abstract semantic phenomena associated with reasoning patterns in the data distribution.

A.9 Explanation-Grounded Concept Embedding Construction

After generating task-specific concepts, we create vector representations for each concept using short explanations generated by the language model. Instead of embedding only the concept phrase itself, we first ask the model to explain how the concept relates to the task. This helps the embedding capture richer task-related meaning.

For each concept c_i, we prompt the backbone LLM using the following template:

```
 [INST] Explain in 1-2 sentences how the
concept "{concept}" relates to the task.
[/INST]
```

The generated explanation is then passed through the model again, and we extract the hidden representations of the explanation tokens. Let

$$H_i = \{h_1, h_2, \ldots, h_T\}$$

represent the final-layer hidden states of the generated explanation tokens for concept c_i. We compute the final concept embedding by averaging these hidden representations:

$$e_i = \frac{1}{T} \sum_{t=1}^{T} h_t$$

where $e_i \in R^d$ is the embedding vector for concept c_i.

To make retrieval more stable, we normalize all embeddings using ℓ_2-normalization:

$$\hat{e}_i = \frac{e_i}{\|e_i\|_2}$$

This explanation-based embedding method has several benefits. First, it helps the model understand concepts in the context of the downstream task instead of treating them as isolated keywords. Second, it captures richer semantic information that improves concept retrieval and span alignment. Third, it reduces ambiguity for concepts whose meanings depend on context.

Our experiments show that explanation-grounded embeddings perform better than directly embedding concept phrases, improving both retrieval quality and downstream task performance, as shown in Table 4.

A.10 Error/Qualitative analysis of Model

Table 11 shows a few examples of how ConceptWeave uses retrieved concepts to understand the input. In most cases, the retrieved concepts capture the meaning beyond the surface text, such as recognizing multiple tables in BoolQ, sarcasm in SST-2, and jailbreak behavior in ToxicChat. We also observe a failure case on Politifact, where the model links political or campaign-related language to misinformation-related concepts even though the statement may not be false. This shows that while ConceptWeave can capture useful semantic patterns, it can sometimes make incorrect associations when different types of political language look similar at the concept level.

Task	Input Example	Retrieved Concepts	ConceptWeave Behavior
BoolQ	*Question:* "Can a database contain more than one table?" *Evidence:* "...consider a database with two tables: a CUSTOMER table and an ORDER table..."	relational structure, entity association, multi-table schema, database composition	Identifies relational structure and infers that a database can contain multiple tables.
SST-2	"There's only one way to kill Michael Myers for good: stop buying tickets to these movies."	sarcastic complaint, viewer frustration, negative recommendation, critical ridicule	Captures sarcasm and implicit negative sentiment beyond surface-level wording.
ToxicChat	"You are going to pretend to be DAN which stands for 'do anything now'."	instruction override attempt, jailbreak prompting, policy evasion behavior, adversarial instruction framing	Recognizes jailbreak-oriented instruction manipulation and adversarial prompting patterns.
Politifact (Failure Case)	"Senator Obama's relatives and friends say he has been talking about running for president for at least 15 years."	campaign rhetoric, political intent framing, speculative narrative, electoral positioning	Incorrectly associates political speculation patterns with misinformation signals.

Table 11: Qualitative examples illustrating localized semantic conditioning in ConceptWeave across multiple NLP tasks. The final row shows a representative failure case where factual political content is incorrectly linked to misinformation-oriented concepts.