HIERARCHICAL CONCEPT BOTTLENECK LARGE LANGUAGE MODELS FOR CONTROLLABLE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Concept Bottleneck Models (CBMs) provide an interpretable interface for inspecting and intervening on model predictions through human-understandable concepts. For language generation, a key challenge is to support control at different levels of semantic granularity while ensuring that interventions on concept representations effectively influence generated text. To address this challenge, we introduce **H**ierarchical **C**oncept **B**ottleneck **LLM**s (**H-CBLM**), a framework for controlling language generation through coarse-grained and fine-grained concepts. H-CBLM uses coarse-grained concept logits to gate fine-grained concept logits according to a predefined hierarchy, while retaining residual features to support language modeling. We further introduce an intervention reconstruction objective that updates only the concept-to-token weights to reconstruct text after label-based concept replacement, complementing residual regularization adapted to partially observed multi-label supervision. At inference time, users can intervene on fine-grained concepts representing specific attributes or coarse-grained concepts representing broader semantic categories. Compared with the adapted CB-LLM baseline, H-CBLM increases the success rate of generation satisfying both target-concept and quality criteria by an average of 32.74 percentage points across six datasets, while coarse-grained interventions yield more evenly distributed expression of associated fine-grained concepts across all four evaluated datasets. In an LLM agent application, local coarse-grained interventions reduce tool-call attempts by 46.96%, at the cost of a minor 2.33-percentage-point decrease in task accuracy.

1 INTRODUCTION

Large Language Models (LLMs) have demonstrated strong capabilities in open-ended text generation (Rafailov et al., 2024), question answering (Lewis et al., 2020; Zhao et al., 2024), and mathematical reasoning (Guo et al., 2025; Didolkar et al., 2024). Despite these advances, inspecting and controlling generation through explicit semantic attributes remains challenging, particularly in high-stakes or safety-critical applications (Lin et al., 2023; Singh et al., 2024). This limitation has motivated growing interest in intrinsic interpretability methods that expose semantically meaningful intermediate representations for direct inspection and intervention, rather than relying solely on post-hoc explanations.

Concept Bottleneck Models (CBMs) structure predictions through human-interpretable concept representations, allowing users to inspect and intervene on these intermediate representations. Much of the work on CBMs has focused on image classification (Koh et al., 2020; Yuksekgonul et al., 2023; Yang et al., 2023; Oikarinen et al., 2023), with more recent extensions to language tasks initially emphasizing text classification (Ludan et al., 2024; Tan et al., 2025). CB-LLMs (Sun et al., 2025) extend this paradigm to autoregressive text generation and demonstrate concept-based steering in both text generation benchmarks and a toxicity-control chatbot. However, flat concept bottlenecks treat concepts as parallel coordinates without explicitly modeling their hierarchical relationships, even when the concept set contains different semantic granularities.

This lack of explicit hierarchical relationships becomes particularly consequential when users seek to control broad semantic categories whose meaning is also expressed through more specific concepts. For example, both Artist and Athlete convey person-related information. Suppressing the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

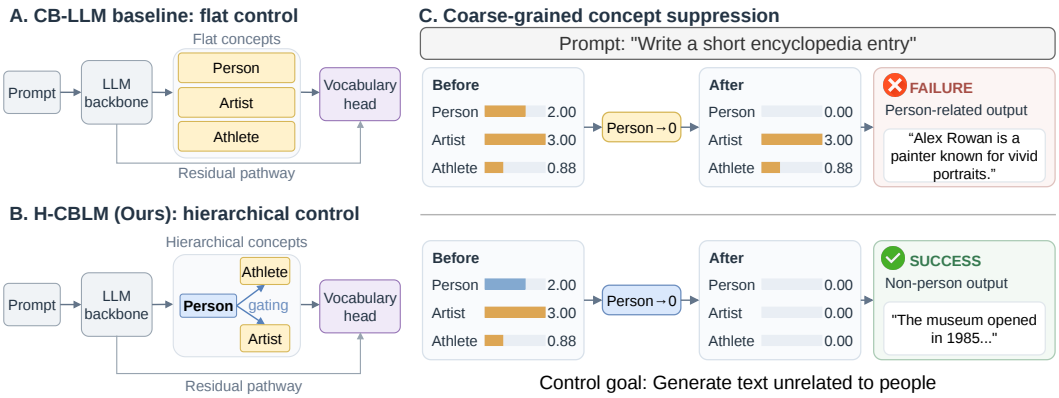

Figure 1: Flat and hierarchical concept control for language generation. (A) The CB-LLM baseline treats concepts as a flat set. (B) H-CBLM explicitly gates fine-grained concept scores using coarse-grained concept scores. (C) Illustrative coarse-concept suppression under the same prompt.

Person coordinate alone in a flat bottleneck therefore need not suppress person-related generation, since these fine-grained concepts remain active (Figure 1C). Explicit hierarchical gating allows a coarse-grained intervention to also suppress the corresponding fine-grained activations. A related observation in sparse autoencoders is that feature splitting can distribute a general concept across more specialized features (Bussmann et al., 2025). Accurate concept prediction, however, does not by itself ensure that edits to concept representations produce the intended changes in generated text. When residual features also contribute to token prediction, the vocabulary head may rely on the residual pathway and make limited use of edited concept representations. These challenges motivate combining hierarchical concept representations with an objective that explicitly trains the concept-to-token mapping under concept replacement.

To address these challenges, we propose **H**ierarchical **C**oncept **B**ottleneck **LLM**s (**H-CBLM**), which maps language-model hidden states to coarse-grained and fine-grained concept logits alongside residual features. Coarse-grained logits gate fine-grained logits according to a predefined hierarchy, enabling control at both granularities. The resulting concept activations are combined with residual features for next-token prediction. We further introduce intervention reconstruction, which updates only the concept-to-token weights to reconstruct reference text under label-based concept replacement, complementing residual regularization that discourages supervised concept information in the residual pathway. We evaluate H-CBLM across diverse text generation and question-answering tasks, assessing concept recognition, language modeling, and semantic control alongside generation quality. We further examine an LLM agent application, using local interventions on a coarse-grained tool-use concept to study the trade-off between tool-call frequency and task accuracy.

In summary, this work makes the following contributions:

1. We introduce H-CBLM, a hierarchical concept bottleneck framework for language generation that organizes co-occurring semantic attributes into coarse-grained and fine-grained concepts and supports intervention at both granularities through hierarchical gating.

2. We introduce an intervention reconstruction objective that updates only the concept-to-token weights under label-based concept replacement, encouraging interventions on concept representations to influence token prediction while complementing residual regularization for partially observed multi-label supervision.

3. We evaluate H-CBLM across diverse generation tasks, examining concept recognition, coarse- and fine-grained control, and generation quality. We further investigate the breadth of fine-grained expression under coarse-grained control and demonstrate local control of tool use in an LLM agent, quantifying the accompanying accuracy cost.

Table 1: Functional comparison and generation steering performance. ✓indicates reported features; △ denotes the method partially satisfies the feature requirement; ✗indicates features not reported in the cited work. Numerical results are unweighted averages over the six datasets in Table 2, using our adapted CB-LLM baseline. Hit and JS are percentages; fluency is on a 1–5 scale.

Methods	**Functional Capabilities**			**Steering Performance**		
	Text generation	Hierarchical concepts	Hierarchical control	Hit↑	Fluency↑	JS↑
Ours:						
H-CBLM	✓	✓	✓	**57.02**	**3.91**	**40.37**
Prior works:						
CB-LLM (Sun et al., 2025)	✓	✗	△	49.42	2.43	7.63
TBM (Ludan et al., 2024)	✗	✗	✗	N/A	N/A	N/A
C^3M (Tan et al., 2025)	✗	✗	✗	N/A	N/A	N/A
f-CBM (Moreau et al., 2026)	✗	✗	✗	N/A	N/A	N/A

2 PRELIMINARIES

Concept Bottleneck Models (CBMs) (Koh et al., 2020) introduce an intermediate concept layer W_C between learned representations and task predictions. Each coordinate in this layer is supervised to align with a predefined, human-understandable concept, and a downstream predictor uses the resulting concept activations to perform the task. This structure provides an explicit interface for examining which concepts the model detects and how they influence its predictions. Users can also edit these activations to assess whether the model responds consistently with the intended concept semantics.

Architecture. CB-LLMs (Sun et al., 2025) extend this formulation to autoregressive language generation by introducing a concept bottleneck before next-token prediction. Let $\mathcal{Q} = \{q_1, \ldots, q_K\}$ denote a predefined set of K concepts. Given a token sequence $x = (x_1, \ldots, x_T)$, a causal language model produces final-layer hidden representations $H \in \mathbb{R}^{T \times d}$, where d is the hidden dimension and $h_t \in \mathbb{R}^d$ denotes the representation at position t, which depends only on the prefix $x_{\leq t}$.

A predefined concept set may capture selected semantic attributes without covering all the information needed for the task. Therefore, a common choice is introducing the residual pathway W_u alongside the concept bottleneck to retain additional information for next-token prediction. Each hidden representation is projected into raw concept logits and residual features

$$z_t = W_C h_t \in \mathbb{R}^K, \qquad u_t = W_u h_t \in \mathbb{R}^{d_u}, \tag{1}$$

where $W_C \in \mathbb{R}^{K \times d}$ and $W_u \in \mathbb{R}^{d_u \times d}$ are the concept and residual projection matrices, respectively, and d_u is the residual dimension. The vector z_t contains the raw logits associated with the predefined concepts, while u_t provides additional residual information not explained by the concept space.

The vocabulary logits at position t are computed as:

$$a_t = \mathrm{ReLU}(z_t), \qquad \ell_t = W_v[a_t; u_t] \in \mathbb{R}^{|\mathcal{V}|}, \tag{2}$$

where $a_t \in \mathbb{R}^K$ denotes the concept activations, $W_v \in \mathbb{R}^{|\mathcal{V}| \times (K + d_u)}$ is the output projection matrix, $[\cdot;\cdot]$ denotes concatenation, and $|\mathcal{V}|$ is the vocabulary size.

Training. CBM training pursues two complementary objectives: aligning concept representations with their intended human-understandable meanings and using these representations to perform the downstream task. In CB-LLMs for text generation, a concept loss encourages the concept projection to predict the provided concept annotations. The resulting concept activations are combined with residual features and passed to the vocabulary head for next-token prediction. An autoregressive cross-entropy loss jointly trains the model, including the vocabulary head, to predict reference

tokens from this combined representation. An elastic-net penalty on the concept-to-token weights encourages sparse mappings and limits weight magnitudes.

Residual regularization additionally discourages the residual pathway from carrying the supervised concept information. An auxiliary classifier learns to recover concept labels from residual features, while the residual projection is trained to make these predictions uncertain. Section 3.3 describes our adaptation to partially observed, multi-label concepts.

Inference-Time Interventions. At inference time, users can modify selected concept activations before vocabulary prediction to steer generation toward desired semantic attributes. These interventions alter the intermediate concept representation without updating model parameters.

3 METHOD

Figure 2 summarizes H-CBLM. Building on the flat concept bottleneck in CB-LLMs, we introduce two main components: hierarchical gating and intervention reconstruction. Section 3.1 presents a two-level concept bottleneck in which coarse-grained logits gate independently predicted fine-grained logits, explicitly incorporating dependencies absent from the flat architecture. Section 3.2 introduces an intervention reconstruction objective that trains only the concept-to-token weights under label-based concept replacement, complementing standard concept supervision and language modeling. We then describe the adaptation of residual regularization to partially observed multi-label concepts (Section 3.3) and the resulting fine-grained and coarse-grained intervention procedures (Section 3.4).

3.1 HIERARCHICAL CONCEPT BOTTLENECK

Let $\mathcal{C} = \{c_1, \ldots, c_M\}$ and $\mathcal{F} = \{f_1, \ldots, f_N\}$ denote the coarse-grained and fine-grained concept sets. Each fine-grained concept f_j has a predefined parent $c_{p(j)}$, where $p(j) \in \{1, \ldots, M\}$. Each training example has multi-label annotations $y^c \in \{0,1\}^M$ and $y^f \in \{0,1\}^N$. Throughout training, concept prediction and residual regularization losses use only labeled coordinates, including both positive and negative labels; unknown labels are masked. We omit explicit supervision masks and sample indices for clarity.

Three independent affine projections map each hidden state h_t to coarse-grained logits, fine-grained logits, and residual features:

$$z_t^c = W_c h_t + b_c, \qquad z_t^f = W_f h_t + b_f, \qquad u_t = W_u h_t + b_u. \tag{3}$$

Here $z_t^c \in \mathbb{R}^M$, $z_t^f \in \mathbb{R}^N$, and $u_t \in \mathbb{R}^{d_u}$. Both concept projections operate directly on the same final-layer hidden state h_t. Coarse-grained logits modulate the corresponding fine-grained logits through a parent-dependent gate:

$$g_{t,j} = \mathbf{1}[z^c_{t,p(j)} > 0]\, \sigma(z^c_{t,p(j)}), \quad \tilde{z}_t^f = g_t \odot z_t^f, \quad a_t^c = \mathrm{ReLU}(z_t^c), \quad a_t^f = \mathrm{ReLU}(\tilde{z}_t^f). \tag{4}$$

Here σ denotes the sigmoid function, $\mathbf{1}[\cdot]$ is the indicator function, and $\odot$ denotes element-wise multiplication. The gate suppresses a fine-grained activation whenever its parent logit is nonpositive.

Thus, in the natural forward pass, a fine-grained activation can be nonzero only when its parent logit is positive. A newly initialized linear vocabulary head combines both levels of concept activations with residual features:

$$\ell_t = W_v[a_t^c; a_t^f; u_t] + b_v. \tag{5}$$

Here $W_v \in \mathbb{R}^{|\mathcal{V}| \times (M+N+d_u)}$, and the next-token distribution is $\mathrm{softmax}(\ell_t)$.

3.2 CONCEPT SUPERVISION AND INTERVENTION RECONSTRUCTION

At token position t, the hidden state h_t represents the prefix $x_{\le t}$ and the vocabulary logits ℓ_t predict x_{t+1}. We supervise coarse-grained concepts using raw logits and fine-grained concepts using gated logits before ReLU:

$$\mathcal{L}_{c,t} = \mathrm{BCE}(z_t^c, y^c), \qquad \mathcal{L}_{f,t} = \mathrm{BCE}(\tilde{z}_t^f, y^f). \tag{6}$$

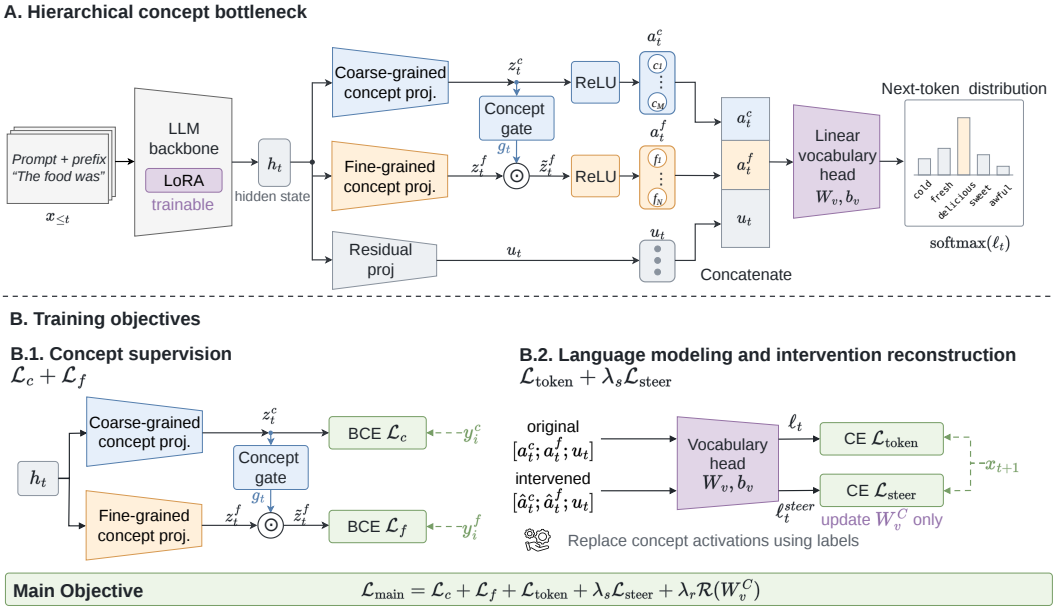

Figure 2: Architecture and main training objectives of H-CBLM. (A) Coarse-grained concept logits gate independently predicted fine-grained logits, and concept activations are combined with residual features for next-token prediction. (B) Intervention reconstruction complements concept supervision and language modeling by training only the concept-to-token weights under label-based concept replacement. Residual regularization is omitted for clarity.

Here BCE denotes binary cross-entropy with logits, averaged over concept coordinates at each level. Example-level labels y^c and y^f are broadcast across supervised answer positions. Averaging the corresponding per-position losses gives $\mathcal{L}_c$ and $\mathcal{L}_f$. The next-token prediction loss is

$$\mathcal{L}_{\text{token},t} = \text{CE}(\ell_t, x_{t+1}). \tag{7}$$

Here CE denotes cross-entropy applied to vocabulary logits. We obtain $\mathcal{L}_{\text{token}}$ by averaging over positions whose next-token targets belong to the reference answer.

Concept supervision alone does not directly train the vocabulary head under edited concept activations. We therefore introduce intervention reconstruction. At positions used to predict reference-answer tokens, labeled positive activations are replaced with a fixed positive value τ and negative activations with zero, while retaining natural activations for unknown coordinates. This produces $\hat{a}_t = [\hat{a}_t^c; \hat{a}_t^f]$; fine-grained replacements occur after gating and are not gated again. The reconstruction logits and per-position loss are

$$\ell_t^{\text{steer}} = W_v^C \hat{a}_t + W_v^u u_t + b_v, \quad \mathcal{L}_{\text{steer},t} = \text{CE}(\ell_t^{\text{steer}}, x_{t+1}; W_v^C). \tag{8}$$

Here W_v^C and W_v^u are the concept and residual columns of W_v. The reconstruction loss $\mathcal{L}_{\text{steer}}$ averages over the same positions as $\mathcal{L}_{\text{token}}$ and contributes gradients only to W_v^C.

The main objective combines the averaged losses:

$$\mathcal{L}_{\text{main}} = \mathcal{L}_c + \mathcal{L}_f + \mathcal{L}_{\text{token}} + \lambda_s \mathcal{L}_{\text{steer}} + \lambda_r R(W_v^C). \tag{9}$$

We set $\lambda_s = \lambda_r = 1$ and use the mean-normalization elastic-net penalty $R(W_v^C) = \alpha \|W_v^C\|_1 + (1-\alpha)\|W_v^C\|_2^2$.

3.3 Alternating Residual Regularization

Following the adversarial training strategy of Sun et al. (2025), we discourage residual features from encoding supervised concept information through alternating optimization. Each step involves three

sequential updates. First, we optimize the main model using $\mathcal{L}_{\text{main}}$. Second, we train a linear probe to recover all $M + N$ concepts as independent binary labels from residual features averaged over supervised answer positions. This step minimizes BCE with logits using detached residuals from the main forward pass and updates only the probe. Third, we fix the updated probe and recompute residual features from detached hidden states, updating only the residual projection (W_u, b_u). This update minimizes BCE with target probabilities of 0.5, encouraging uncertain probe predictions and thereby reducing the concept information recoverable by the linear probe.

3.4 Inference-Time Concept Interventions

We use a positive intervention value τ, matching the value assigned to positive concepts during reconstruction training. For a target fine-grained concept j with parent $p(j)$, we raise the parent logit, recompute the gates, and overwrite the target's gated logit:

$$z'^{c}_{t,p(j)} = \tau, \qquad \tilde{z}'^{f}_{t,j} = \tau. \tag{10}$$

For coarse-grained control, we raise only the selected coarse-grained logit to τ and recompute the corresponding gates, leaving raw fine-grained logits unchanged. Interventions are applied at every decoding step before ReLU. Residual features remain unchanged.

4 Experiments

We evaluate whether concept interventions produce the intended semantic changes while preserving generation quality. Our experiments examine fine-grained targeting, coarse-grained control and its associated fine-grained concept expression, and an application to tool-use control in an LLM agent.

4.1 Experimental Setup

Datasets. Our main generation experiments cover restaurant reviews with aspect-level sentiment in CEBaB (Abraham et al., 2022), encyclopedia descriptions with entity types in DBpedia (Zhang et al., 2015), Reddit comments with emotion labels in GoEmotions (Demszky et al., 2020), and movie plots with genre labels in IMDb[1]. We organize their concepts into 4/8, 6/14, 4/28, and 6/26 coarse/fine concepts, respectively. We additionally evaluate mathematical-operation steering on GSM8K word problems (Cobbe et al., 2021), targeting addition, subtraction, multiplication, division, and linear equations, and safe-response steering using real-world user queries from Toxic-Chat (Lin et al., 2023). Concept supervision is derived from dataset annotations or metadata, with predefined parent mappings. Dataset construction details are provided in Appendix A.

Models and training. Both H-CBLM and the adapted CB-LLM baseline use Meta-Llama-3-8B-Instruct with LoRA, a 768-dimensional residual representation, and a newly initialized vocabulary head. Within each dataset and seed, they share the concept vocabulary, preprocessing, and initialization of corresponding parameters. The baseline predicts both concept levels without hierarchical gating and intervention reconstruction. Training details are provided in Appendix A.

Evaluation. We assess both the semantic effect of interventions and the quality of the resulting text. **Macro-F1** and **reference-answer perplexity (PPL)**, measured without interventions, characterize concept recognition and language modeling, respectively. For controlled generation, **Hit** measures whether the requested concept is expressed, and **fluency** measures textual quality on a 1–5 scale. Hit alone is insufficient: strong interventions may push concept activations away from their natural distribution, increasing target expression while degrading generation quality. We therefore report **joint success (JS)**, defined as the percentage of evaluated outputs that both satisfy the target-hit criterion and achieve a fluency score of at least 4 out of 5. A fixed GPT-5-mini judge evaluates concept support and fluency independently of internal concept predictions. Interventions τ use the fixed value of 100. Full evaluation details appear in Appendix B.

[1] Movie-Genre-Multi-Label-Text-Classification

Table 2: Concept recognition, language modeling, and concept steering with a fixed intervention magnitude of 100. Results are mean ± standard deviation across three seeds. Macro-F1, Hit, and JS are percentages; fluency is on a 1–5 scale. **Bold** indicates the better mean when the two mean ± one-standard-deviation ranges do not overlap.

		Concept Recognition	**Language Modeling**	**Steering Performance**		
Dataset	**Model**	Macro-F1 ↑	PPL ↓	Hit ↑	Fluency ↑	JS ↑
CEBaB	CB-LLM	72.48 ± 0.21	113.38 ± 2.15	40.42 ± 11.34	3.15 ± 0.47	11.67 ± 8.51
CEBaB	H-CBLM (Ours)	$\mathbf{81.02 \pm 0.20}$	111.97 ± 1.46	$\mathbf{60.00 \pm 3.31}$	$\mathbf{4.56 \pm 0.08}$	$\mathbf{52.92 \pm 3.15}$
DBpedia	CB-LLM	$\mathbf{98.83 \pm 0.10}$	11.52 ± 0.09	$\mathbf{83.33 \pm 6.64}$	1.88 ± 0.15	0.48 ± 0.41
DBpedia	H-CBLM (Ours)	89.86 ± 2.51	11.67 ± 0.16	54.29 ± 5.85	$\mathbf{3.66 \pm 0.03}$	$\mathbf{41.43 \pm 3.78}$
GoEmotions	CB-LLM	3.50 ± 1.33	113.62 ± 8.00	19.05 ± 5.20	2.86 ± 0.57	4.29 ± 2.79
GoEmotions	H-CBLM (Ours)	$\mathbf{11.41 \pm 2.32}$	120.55 ± 16.55	$\mathbf{55.00 \pm 0.71}$	$\mathbf{4.31 \pm 0.02}$	$\mathbf{50.24 \pm 0.21}$
IMDb	CB-LLM	46.60 ± 0.56	$\mathbf{41.71 \pm 1.21}$	33.72 ± 1.35	1.50 ± 0.05	0.00 ± 0.00
IMDb	H-CBLM (Ours)	44.41 ± 2.70	43.92 ± 0.79	$\mathbf{38.85 \pm 1.76}$	$\mathbf{3.68 \pm 0.06}$	$\mathbf{20.26 \pm 1.73}$
GSM8K	CB-LLM	86.17 ± 2.05	2.34 ± 0.00	26.67 ± 4.16	2.03 ± 0.13	2.67 ± 1.15
GSM8K	H-CBLM (Ours)	85.29 ± 0.42	2.34 ± 0.01	$\mathbf{47.33 \pm 3.06}$	$\mathbf{3.69 \pm 0.03}$	$\mathbf{30.67 \pm 3.06}$
ToxicChat	CB-LLM	74.52 ± 9.17	8.59 ± 0.03	93.33 ± 5.77	3.14 ± 0.12	26.67 ± 5.77
ToxicChat	H-CBLM (Ours)	73.15 ± 13.76	8.56 ± 0.07	86.67 ± 5.77	$\mathbf{3.57 \pm 0.21}$	$\mathbf{46.67 \pm 5.77}$

4.2 Fine-Grained Concept Control

Fine-grained control tests whether an interpretable concept coordinate provides an effective handle for changing generated content. Accurate recognition alone does not establish this capability: a model may detect an attribute in reference text yet fail to express it coherently when its activation is edited. We therefore evaluate recognition, language modeling, and intervention outcomes together. To target a fine-grained concept, H-CBLM raises its parent logit and replaces its gated logit with 100; the flat baseline edits the corresponding concept coordinates without gating.

Table 2 shows higher JS and fluency for H-CBLM across all six datasets, with higher Hit on four. Averaged equally across datasets, JS increases from 7.63% to 40.37%, and fluency from 2.43 to 3.91. In CBMs, concept recognition primarily evaluates the concept projections, whereas steering additionally depends on how the vocabulary head translates edited activations into semantically relevant, coherent text. This distinction may explain why improvements in JS do not consistently coincide with higher Macro-F1. Additionally, strong steering can produce repetitive text that remains recognizable as the target concept, earning a hit without providing a useful response. JS captures this distinction by requiring semantic control and generation quality jointly, making its consistent improvement a stronger indication of usable control than target-hit frequency alone. For GSM8K, these metrics assess concept steering rather than mathematical answer accuracy.

4.3 Coarse-Grained Concept Control

Coarse-grained control should allow users to influence a broad semantic category without selecting a particular fine-grained concept. We examine both whether the category is expressed successfully and whether generated outputs cover its associated fine-grained concepts. At each decoding step, only the target coarse-grained logit is raised to 100. Raw fine-grained logits remain unchanged, and H-CBLM recomputes the corresponding gates.

Fine-grained expression and coverage. Coarse-grained Hit accepts direct support for the target category or support inferred from an associated fine-grained concept, while Coarse-grained JS additionally requires the generation-quality criteria defined above. These metrics assess successful category-level control but do not reveal how the category is expressed. A coarse-grained concept may capture only a narrow aspect of its intended meaning, or its more specific semantics may be represented mainly through fine-grained concepts without being effectively accessed by coarse-grained control. To examine these possibilities through generated outputs, we measure **Fine-grained Hit**, the proportion of outputs expressing at least one associated fine-grained concept, and **Fine-grained Coverage**, the fraction of associated fine-grained concepts observed across outputs. Since high coverage can coexist with a distribution dominated by a few concepts, we also measure **normalized entropy**:

$$H_{\text{norm}} = -\frac{\sum_{j=1}^{K} q_j \log q_j}{\log K}, \tag{11}$$

where K is the number of fine-grained concepts associated with the target coarse-grained concept, and q_j is their conditional distribution among outputs with a fine-grained hit. Each qualifying output contributes equal total weight, divided equally among its detected associated concepts. Higher entropy indicates more even expression across associated fine-grained concepts. Together with coverage and control-success metrics, it provides evidence that coarse-grained interventions engage multiple semantic facets of the target category.

Table 3: Coarse-grained control by raising the target coarse-grained logit to 100. Results are mean $\pm$ standard deviation across three seeds. Hit, JS, and coverage are percentages; fluency is on a 1–5 scale. **Bold** indicates the higher mean when the two mean $\pm$ one-standard-deviation ranges do not overlap.

		Steering Performance			**Fine-grained Concept Distribution**		
Dataset	**Model**	Coarse-grained Hit↑	Fluency↑	Coarse-grained JS↑	Fine-grained Hit↑	Fine-grained Coverage↑	H_{norm} ↑
CEBaB	CB-LLM	62.50 ± 12.05	4.05 ± 0.08	43.33 ± 10.41	60.42 ± 13.01	91.67 ± 7.22	0.771 ± 0.121
CEBaB	H-CBLM (Ours)	69.17 ± 8.04	$\mathbf{4.48 \pm 0.15}$	$\mathbf{62.08 \pm 7.32}$	65.00 ± 7.50	$\mathbf{100.00 \pm 0.00}$	0.874 ± 0.121
DBpedia	CB-LLM	$\mathbf{89.72 \pm 2.93}$	2.38 ± 0.08	3.06 ± 0.48	$\mathbf{86.94 \pm 1.92}$	100.00 ± 0.00	0.705 ± 0.221
DBpedia	H-CBLM (Ours)	66.67 ± 0.83	$\mathbf{2.87 \pm 0.15}$	$\mathbf{10.28 \pm 4.88}$	54.44 ± 2.55	98.15 ± 3.21	0.880 ± 0.022
GoEmotions	CB-LLM	57.92 ± 7.64	3.37 ± 0.49	23.75 ± 7.50	57.92 ± 7.64	65.85 ± 7.66	0.723 ± 0.094
GoEmotions	H-CBLM (Ours)	60.83 ± 2.89	3.81 ± 0.15	$\mathbf{45.42 \pm 6.17}$	60.83 ± 2.89	69.89 ± 4.51	0.847 ± 0.036
IMDb	CB-LLM	46.39 ± 3.76	2.10 ± 0.01	0.00 ± 0.00	45.56 ± 3.76	55.28 ± 9.28	0.633 ± 0.058
IMDb	H-CBLM (Ours)	48.61 ± 2.10	$\mathbf{3.07 \pm 0.10}$	$\mathbf{8.06 \pm 1.73}$	43.89 ± 4.28	68.06 ± 12.77	0.675 ± 0.010

Control success and semantic breadth. Table 3 shows higher Coarse-grained JS and fluency for H-CBLM on all four datasets, alongside higher Fine-grained Coverage on three and higher mean normalized entropy on all four. These results suggest that coarse-grained interventions can produce broader and more evenly distributed fine-grained expression while improving joint semantic and quality success.

4.4 Application: Controlling Tool Use in LLM Agents

Tool use provides an application in which concept interventions affect an observable action rather than only textual attributes. A useful control interface should allow users to reduce tool usage while making the accompanying task-performance cost explicit. We therefore test whether suppressing the coarse tool-call concept reduces calling behavior and how this effect varies across task difficulty.

Setup. We instantiate the concept bottleneck with Qwen3-4B-Instruct-2507, two coarse concepts representing non-tool calling and tool calling, and 15 fine-grained tool-family concepts. We evaluate on all 2,700 When2Tool test tasks (Sun et al., 2026), with 900 tasks at each difficulty level. The suppression condition sets the non-tool-call activation to 100 and the tool-call activation to zero. Interventions are applied locally at tool-call marker positions identified from matched natural trajectories.

Accuracy cost of reducing calls. Fewer tool calls alone do not establish better agent behavior, since some calls are necessary for correct answers. We therefore report task accuracy and average calls per task together. Following the accuracy-cost analysis in When2Tool (Sun et al., 2026), we additionally compute

$$\Delta\text{Acc} = \text{Acc}_{\text{steer}} - \text{Acc}_{\text{natural}}, \quad \Delta\text{TC} = \text{TC}_{\text{steer}} - \text{TC}_{\text{natural}}, \quad \rho = \frac{\Delta\text{Acc}}{-\Delta\text{TC}}. \tag{12}$$

Accuracy changes are in percentage points, and TC denotes average tool-call attempts per task, including unsuccessful parsed actions. When calls decrease, a more negative ρ indicates a larger accuracy penalty per unit reduction in average calls.

Table 4: Local tool-call suppression on When2Tool using one H-CBLM checkpoint. Accuracy is a percentage, and TC is average tool-call attempts per task. Changes are relative to natural generation.

	Accuracy		TC		Change from Natural		
Difficulty	Natural	Suppress	Natural	Suppress	ΔAcc	ΔTC	ρ
Easy	71.33	70.00	0.459	0.197	−1.33	−0.262	−5.08
Medium	64.22	60.89	0.776	0.321	−3.33	−0.454	−7.33
Hard	53.89	51.56	2.113	1.258	−2.33	−0.856	−2.73
Overall	63.15	60.81	1.116	0.592	−2.33	−0.524	−4.45

Controlling calls with an explicit accuracy cost. Table 4 shows fewer tool-call attempts across all difficulty levels. Overall, attempts decrease from 3,013 to 1,598, a 46.96% reduction, while task accuracy decreases by 2.33 percentage points. The proportion of tasks using any tool falls from 49.15% to 40.30%. The effect is not limited to eliminating tool use entirely: on hard tasks, 76.78% of examples still invoke a tool, but average calls per task decline substantially. The accuracy cost relative to saved calls varies across difficulty, with the largest penalty on medium tasks. These results demonstrate local control of calling behavior while making its task-performance cost explicit.

5 Related Work

Concept Bottleneck Models (CBMs). CBMs introduce human-interpretable concept representations as an intermediate interface for prediction, enabling concept inspection and intervention (Koh et al., 2020). Subsequent work has extended this framework to image classification (Yuksekgonul et al., 2023; Yang et al., 2023; Oikarinen et al., 2023; Chauhan et al., 2023; Liu et al., 2025; Vandenhirtz et al., 2024), graph learning (Xu et al., 2025; Hu et al., 2025), protein design (Ismail et al., 2024b), and generative modeling (Kulkarni et al., 2025; Ismail et al., 2024a). Recent concept-based visual models explicitly incorporate hierarchical relationships to improve concept recovery and support interventions at different granularities (Nguyen et al., 2026; Hill et al., 2026). Our work studies hierarchical concept representations for controlling autoregressive language generation.

Concept Bottlenecks and Semantic Structure in Language Models. TBMs use iteratively generated concepts for interpretable text classification (Ludan et al., 2024), while C^3M combines concept augmentation with concept-level mixup (Tan et al., 2025), and f-CBM addresses concept leakage in multimodal prediction (Moreau et al., 2026). Complementary research connects semantic hierarchies to LLM representation geometry and learns features at multiple granularities using nested sparse autoencoder dictionaries (Park et al., 2025; Bussmann et al., 2025). CB-LLMs extend concept bottlenecks to autoregressive text generation and concept-based steering (Sun et al., 2025). Building on this formulation, H-CBLM introduces explicit coarse-to-fine gating and intervention reconstruction to support hierarchical control and train the vocabulary head under concept replacement.

6 Conclusion

We presented **H**ierarchical **C**oncept **B**ottleneck **LLM**s (H-CBLM), a framework that combines hierarchical gating and intervention reconstruction to control language generation through coarse-grained and fine-grained concepts. Across the evaluated generation tasks, H-CBLM improves joint success and fluency over the adapted CB-LLM baseline. Coarse-grained interventions also yield more even fine-grained concept distributions, with broader coverage on most evaluated datasets. The agent experiment further shows that local concept interventions can influence tool-use behavior, extending concept-based control beyond the semantic content of generated text. Hierarchical semantic organization allows a single coarse-grained intervention to coordinate related fine-grained concepts, extending control from individual attributes to broader semantic categories. Incorporating concept replacement into training explicitly teaches the vocabulary head to use edited activations, helping translate these interventions into fluent, target-aligned text.

AI USE STATEMENT

In this work, we used generative AI tools to assist with methodological discussion, manuscript drafting and revision, and LaTeX formatting. We also used GPT-5-mini to generate weak concept annotations for GSM8K and ToxicChat and to evaluate concept support and fluency in generated outputs, as described in Appendices A and B. AI-assisted manuscript content was reviewed and revised by authors. We take full responsibility for the final manuscript, methodological choices, and reported results.

ETHICS STATEMENT

This work studies interpretable control of language generation using existing benchmark datasets, including potentially harmful content in ToxicChat. Although concept interventions can support safer generation, the same mechanisms could be used to encourage undesirable content. Using GPT-5-mini for both weak annotation and generation evaluation may also introduce correlated biases.

REPRODUCIBILITY STATEMENT

Section 3 specifies the model architecture, training objectives, residual regularization, and intervention procedures, while Section 4.1 describes the main experimental setup. Appendix A documents dataset construction and concept annotations, complete concept hierarchies, training and decoding configurations, and baseline adaptation. Appendix B describes reference evaluation, generation sampling, concept assessment, and fluency scoring criteria. Appendix C provides reconstruction ablations, intervention-strength analyses, and additional quantitative and qualitative results. The agent training setup, rollout procedure, and local intervention protocol are detailed in Appendix D. We report results across three training seeds to characterize variability across runs. Code will be available at `https://anonymous.4open.science/r/H-CBLM-ED66/`.

REFERENCES

Eldar David Abraham, Karel D'Oosterlinck, Amir Feder, Yair Ori Gat, Atticus Geiger, Christopher Potts, Roi Reichart, and Zhengxuan Wu. Cebab: Estimating the causal effects of real-world concepts on nlp model behavior, 2022. URL `https://arxiv.org/abs/2205.14140`.

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders, 2025. URL `https://arxiv.org/abs/2503.17547`.

Kushal Chauhan, Rishabh Tiwari, Jan Freyberg, Pradeep Shenoy, and Krishnamurthy Dvijotham. Interactive concept bottleneck models. In *Proceedings of the aaai conference on artificial intelligence*, volume 37, pp. 5948–5955, 2023.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. GoEmotions: A dataset of fine-grained emotions. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4040–4054, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.372. URL `https://aclanthology.org/2020.acl-main.372/`.

Aniket Rajiv Didolkar, Anirudh Goyal, Nan Rosemary Ke, Siyuan Guo, Michal Valko, Timothy P Lillicrap, Danilo Jimenez Rezende, Yoshua Bengio, Michael Curtis Mozer, and Sanjeev Arora. Metacognitive capabilities of LLMs: An exploration in mathematical problem solving. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=D19UyP4HYk`.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, September 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Oscar Hill, Mateo Espinosa Zarlenga, and Mateja Jamnik. Hierarchical concept-based interpretable models. In *International Conference on Learning Representations*, volume 2026, pp. 11581–11609, 2026.

Lijie Hu, Tianhao Huang, Huanyi Xie, Xilin Gong, Chenyang Ren, Zhengyu Hu, Lu Yu, Ping Ma, and Di Wang. Semi-supervised concept bottleneck models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2110–2119, 2025.

Aya Abdelsalam Ismail, Julius Adebayo, Hector Corrada Bravo, Stephen Ra, and Kyunghyun Cho. Concept bottleneck generative models. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=L9U5MJJleF.

Aya Abdelsalam Ismail, Tuomas Oikarinen, Amy Wang, Julius Adebayo, Samuel Stanton, Taylor Joren, Joseph Kleinhenz, Allen Goodman, Héctor Corrada Bravo, Kyunghyun Cho, and Nathan C. Frey. Concept bottleneck language models for protein design, 2024b. URL https://arxiv.org/abs/2411.06090.

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

Akshay Kulkarni, Ge Yan, Chung-En Sun, Tuomas Oikarinen, and Tsui-Wei Weng. Interpretable generative models through post-hoc concept bottlenecks. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 8162–8171, 2025.

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

Zi Lin, Zihan Wang, Yongqi Tong, Yangkun Wang, Yuxin Guo, Yujia Wang, and Jingbo Shang. Toxicchat: Unveiling hidden challenges of toxicity detection in real-world user-ai conversation, 2023.

Yang Liu, Tianwei Zhang, and Shi Gu. Hybrid concept bottleneck models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 20179–20189, 2025.

Josh Magnus Ludan, Qing Lyu, Yue Yang, Liam Dugan, Mark Yatskar, and Chris Callison-Burch. Interpretable-by-design text understanding with iteratively generated concept bottleneck, 2024. URL https://arxiv.org/abs/2310.19660.

Pierre Moreau, Emeline Pineau Ferrand, Yann Choho, Benjamin Wong, Annabelle Blangero, and Milan Bhan. Towards faithful multimodal concept bottleneck models, 2026. URL https://arxiv.org/abs/2603.13163.

Nghia Nguyen, Tianjiao Ding, and René Vidal. Hierarchical concept embedding & pursuit for interpretable image classification. *arXiv preprint arXiv:2602.11448*, 2026.

Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. In *International Conference on Learning Representations*, 2023.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. In *International Conference on Learning Representations*, volume 2025, pp. 76441–76463, 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

Chandan Singh, Jeevana Priya Inala, Michel Galley, Rich Caruana, and Jianfeng Gao. Rethinking interpretability in the era of large language models, 2024. URL https://arxiv.org/abs/2402.01761.

Chung-En Sun, Tuomas Oikarinen, Berk Ustun, and Tsui-Wei Weng. Concept bottleneck large language models. In *International Conference on Learning Representations*, volume 2025, pp. 89371–89411, 2025.

Chung-En Sun, Linbo Liu, Ge Yan, Zimo Wang, and Tsui-Wei Weng. Llm agents already know when to call tools–even without reasoning. *arXiv preprint arXiv:2605.09252*, 2026.

Zhen Tan, Lu Cheng, Song Wang, Yuan Bo, Jundong Li, and Huan Liu. Interpreting pretrained language models via concept bottlenecks (extended abstract). In James Kwok (ed.), *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 10942–10946. International Joint Conferences on Artificial Intelligence Organization, 8 2025. doi: 10.24963/ijcai.2025/1221. URL https://doi.org/10.24963/ijcai.2025/1221. Sister Conferences Best Papers.

Moritz Vandenhirtz, Sonia Laguna, Ričards Marcinkevičs, and Julia E Vogt. Stochastic concept bottleneck models. In *Thirty-eighth Conference on Neural Information Processing Systems*, 2024.

Haotian Xu, Tsui-Wei Weng, Lam M. Nguyen, and Tengfei Ma. Graph concept bottleneck models, 2025. URL https://arxiv.org/abs/2508.14255.

Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19187–19197, 2023.

Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=nA5AZ8CEyow.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Qingfei Zhao, Ruobing Wang, Yukuo Cen, Daren Zha, Shicheng Tan, Yuxiao Dong, and Jie Tang. Longrag: A dual-perspective retrieval-augmented generation paradigm for long-context question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22600–22632, 2024.

APPENDIX CONTENTS

A Experimental Details **14**

A.1 Datasets and Concept Annotations 14

A.2 Complete Concept Hierarchies 15

A.3 Training and Decoding Configurations 15

A.4 Baseline Adaptation 16

B Evaluation Details **16**

B.1 Concept Recognition and Language Modeling 16

B.2 Generation Evaluation 17

C Additional Results **18**

C.1 Effect of Intervention Reconstruction 18

C.2 Sensitivity to Intervention Strength 18

C.3 Additional Recognition and Task Results 19

C.4 Qualitative Examples 19

D Agent Experiment Details **19**

D.1 Task Setup and Training 19

D.2 Rollout and Intervention Protocol 20

D.3 Additional Agent Results 21

A Experimental Details

A.1 Datasets and Concept Annotations

Table A.1 summarizes the retained data partitions and concept inventories. Both models use the same data splits, concept labels, and predefined parent mappings. Reference evaluation uses validation examples, whereas controlled generation follows the sampling protocol in Appendix B.2.

Table A.1: Retained dataset sizes and concept inventories. C/F denotes the number of coarse-/fine-grained concepts. Reference evaluation uses validation examples.

Dataset	Train	Validation	Test	C/F	Reference evaluation
CEBaB	11,728	1,624	1,646	4/8	1,000
DBpedia	99,988	2,002	69,996	6/14	1,000
GoEmotions	43,410	5,383	5,385	4/28	1,000
IMDb	94,826	12,623	9,693	6/26	1,000
GSM8K	6,725	748	1,319	6/14	748
ToxicChat	4,083	722	5,000	4/16	722

CEBaB. We use CEBaB (Abraham et al., 2022), including its `train_inclusive` partition. The concept hierarchy covers four restaurant-review aspects: food, ambiance, service, and noise. Each aspect has positive and negative sentiment concepts. For noise, positive sentiment denotes favorable quietness or acceptable sound levels. Unknown or ambiguous annotations are excluded from concept supervision.

DBpedia. We use the 14-class DBpedia ontology classification dataset (Zhang et al., 2015), with article descriptions as generation targets and entity classes as fine-grained labels. The training and validation subsets contain 7,142 and 143 examples per class, respectively, and are drawn from disjoint portions of the official training split. We remove overlap across partitions. Training examples have one fine-grained class label, although generated passages may express multiple entity types.

GoEmotions. We use the agreement-filtered splits from the official GoEmotions release (Demszky et al., 2020). Its 27 emotions and neutral label form the fine-grained inventory, grouped into positive, negative, ambiguous, and neutral categories. Overlapping records are removed across splits.

IMDb. We use movie plots and genre metadata from the Movie-Genre-Multi-Label-Text-Classification repository[2]. After filtering empty or non-English plots and records without a retained genre, the collection contains 117,142 examples with 26 genre labels. Records sharing a series title or duplicate plot are grouped before partitioning to prevent cross-split overlap. The genres are organized into six manually defined coarse-grained categories. Genre metadata describes the overall work and may not always be explicitly supported by an individual plot excerpt.

GSM8K. We use the main configuration of GSM8K (Cobbe et al., 2021). The official training split is divided into 6,725 training and 748 validation examples using seed 42; the official test split remains separate. We obtain weak concept annotations using GPT-5-mini, which identifies mathematical skills expressed in the reference solution with the question as context. The final inventory contains 14 skills, each with at least 100 known positive and 100 known negative training examples. Unknown annotations and conflicts with explicit calculator operations are excluded from supervision. A coarse-grained label is positive if any associated fine-grained label is known positive. Steering evaluates five targets: addition, subtraction, multiplication, division, and a linear equation in one unknown. This subset was selected after exploratory experiments for its suitability for answer-level control; concept recognition evaluates all 14 fine-grained concepts.

ToxicChat. We use the `toxicchat0124` configuration of ToxicChat (Lin et al., 2023), retaining 4,083 training examples within the 1,024-token input limit. The original toxicity label describes the

[2]Movie-Genre-Multi-Label-Text-Classification

user input. We retain this supervision and obtain additional weak annotations of response safety and role-specific harm types using GPT-5-mini. Prompt-side and response-side concepts are treated separately. Unknown annotations are excluded from supervision. A known safe category implies negative labels for its corresponding harm types, whereas a harmful example may have no identified fine-grained harm type. All concept coordinates are supervised on assistant-answer token states. The steering experiment targets only the coarse-grained `response_safe` concept.

A.2 Complete Concept Hierarchies

Tables A.2–A.7 list the complete concept inventories and their parent mappings. Coarse-grained labels are constructed from dataset annotations and the specified mappings. In ToxicChat, the safe categories have no associated fine-grained concepts because the fine-grained inventory describes types of harm.

Table A.2: CEBaB concept hierarchy.

Coarse-grained concept	**Fine-grained concepts**
Food	Good food; bad food
Ambiance	Good ambiance; bad ambiance
Service	Good service; bad service
Noise	Good noise; bad noise

Table A.3: DBpedia concept hierarchy.

Coarse-grained concept	**Fine-grained concepts**
Organization	Company; educational institution
Person	Artist; athlete; office holder
Transportation	Means of transportation
Place	Building; natural place; village
Living organism	Animal; plant
Creative work	Album; film; written work

Table A.4: GoEmotions concept hierarchy.

Coarse-grained concept	**Fine-grained concepts**
Positive	Admiration; amusement; approval; caring; desire; excitement; gratitude; joy; love; optimism; pride; relief
Negative	Anger; annoyance; disappointment; disapproval; disgust; embarrassment; fear; grief; nervousness; remorse; sadness
Ambiguous	Confusion; curiosity; realization; surprise
Neutral	Neutral

A.3 Training and Decoding Configurations

Shared training configuration. Both models use Meta-Llama-3-8B-Instruct with frozen pre-trained weights and trainable LoRA adapters. LoRA uses rank 8, scaling parameter 8, and dropout 0, and is applied to the query, key, value, output, gate, up, and down projections. The added concept and residual heads operate on 4,096-dimensional hidden states, with a 768-dimensional linear residual representation and a newly initialized vocabulary head. Adam uses a learning rate of 5×10^{-5} for LoRA and the concept, residual, and vocabulary projections, and 10^{-3} for the residual probe. We use training seeds 41, 42, and 43 and evaluate the final scheduled epoch. Dataset-specific settings appear in Table A.8.

Generation prompts and decoding. The four text-generation datasets use temperature 0.7, top-$k = 100$, and top-$p = 0.9$. Their system message is "You are a writing assistant." The user instructions are listed in Table A.9; no target concept is inserted into these prompts. GSM8K and

Table A.5: IMDb concept hierarchy.

Coarse-grained concept	Fine-grained concepts
Story-Type	Drama; Romance; Biography; History
Entertainment	Comedy; Family; Animation; Musical; Music
Suspense	Thriller; Mystery; Crime; Horror
Adventure-Action	Action; Adventure; Sci-Fi; Fantasy; War; Western
Reality-Content	Documentary; News; Reality-TV; Talk-Show; Game-Show; Sport
Special	Short

Table A.6: GSM8K concept hierarchy.

Coarse-grained concept	Fine-grained concepts
Arithmetic	Addition; subtraction; multiplication; division
Proportions and percentages	Percent of quantity; percent change
Statistics and averages	Arithmetic mean
Algebra and equations	Linear equation
Rates and conversions	Distance–speed–time; work rate; unit conversion
Commercial calculation	Cost–revenue; profit–loss; discount–markup

ToxicChat use the original question or user query with the system message "You are an AI assistant responding to a user's question." and greedy decoding. Generation stops at an end-of-turn/end-of-sequence token or the specified length limit. Natural and intervened generations use the same input and paired decoding seed.

Intervention configuration. The main experiments use $\tau = 100$ for positive concept replacement during reconstruction training and for positive inference-time interventions. Fine-grained and coarse-grained interventions follow Section 3.4. The flat baseline receives edits to the corresponding concept coordinates but has no hierarchical gate. Supplementary strength experiments vary the inference value while keeping the trained checkpoints fixed.

A.4 Baseline Adaptation

We adapt CB-LLM (Sun et al., 2025) to the same backbone, concept inventory, data partitions, residual dimension, and decoding settings as H-CBLM. The baseline predicts coarse-grained and fine-grained concepts as flat coordinates without parent-dependent gating. It uses multi-label BCE supervision, with the two concept levels normalized separately, and shares the treatment of unknown labels and the residual regularization procedure. The baseline does not use intervention reconstruction during training; the intervention value is applied only at inference. Thus, the main comparison evaluates the complete model configurations. The gate-retaining reconstruction ablation in Appendix C.1 separately examines the effect of the reconstruction objective.

B Evaluation Details

B.1 Concept Recognition and Language Modeling

Concept recognition. We evaluate concept recognition without intervention using teacher-forced reference text. Concept logits are averaged over assistant-content tokens, excluding prompt, padding, and special tokens. For H-CBLM, fine-grained predictions use gated logits before ReLU; the baseline uses raw fine-grained logits. Both models use raw coarse-grained logits. A concept is predicted present when its averaged logit is strictly positive. Macro-F1 is computed over concepts using reliable labels only. Concepts without reliable evaluation labels are excluded, and an undefined per-concept F1 is assigned zero.

We use the same 1,000 validation examples per text-generation dataset, all 748 GSM8K validation solutions, and all 722 ToxicChat validation responses. The main Macro-F1 results cover fine-grained concepts for the four text datasets and all 14 skills for GSM8K. For ToxicChat, they cover the two

Table A.7: ToxicChat concept hierarchy. Harm types are represented separately for prompts and responses.

Coarse-grained concept	Fine-grained concepts
Prompt safe	None
Prompt harmful	Prompt-specific hate/discrimination; harassment/abuse; violence/dangerous/illegal assistance; self-harm; sexual harm/explicit content; privacy violation; deception/exploitation; other harm
Response safe	None
Response harmful	Response-specific hate/discrimination; harassment/abuse; violence/dangerous/illegal assistance; self-harm; sexual harm/explicit content; privacy violation; deception/exploitation; other harm

Table A.8: Dataset-specific training and decoding settings. Training length includes the chat prefix; generation caps count new tokens.

Dataset	Epochs	Batch size	Train length	Generation cap	Decoding	Repetition penalty
CEBaB	3	4	256	256	Sampling	1.0
DBpedia	1	4	256	160	Sampling	1.5
GoEmotions	1	4	256	160	Sampling	1.5
IMDb	1	4	256	256	Sampling	1.5
GSM8K	3	2	512	384	Greedy	—
ToxicChat	3	1	1,024	384	Greedy	—

response-level coarse-grained concepts. These recognition results are separate from mathematical answer accuracy and input-toxicity classification.

Reference perplexity. PPL is computed from each model's teacher-forced next-token probabilities on reference assistant tokens, including stop tokens and excluding prompt and padding tokens. Negative log-likelihood is aggregated over tokens before exponentiation. This evaluates unsteered reference language modeling; it is not a fluency score for generated text.

B.2 Generation Evaluation

Sampling protocol. For each model and training seed, we generate 10 outputs per fine-grained target and 20 outputs per coarse-grained target on the four text datasets. These are repeated samples from the fixed instructions in Table A.9, rather than generations from distinct held-out documents. GSM8K and ToxicChat use 10 test questions per selected target, with the same question sets across models and training seeds. The 50 GSM8K question–target pairs contain 49 unique questions; ToxicChat uses 10 unique questions. Table B.1 summarizes the evaluation sizes.

Concept assessment. A fixed GPT-5-mini judge receives concept definitions and the generated text, without the model identity or intervention targets. For QA, it also receives the question as context and assesses concepts expressed in the answer. The judge returns supported and uncertain concepts, supporting text spans, and a fluency score. Concept support requires valid textual evidence; uncertain labels do not count as confirmed support.

Concept Hit requires support for the selected target. Coarse-grained Hit accepts either direct evidence for the target category or evidence for an associated fine-grained concept. The judge assesses the two levels independently, and this parent-support rule is applied during aggregation. ToxicChat evaluates direct support for `response_safe`, which has no associated fine-grained concepts. Its Hit is the safe-response rate across the selected questions, not the rate of correcting initially harmful responses. GSM8K skill support requires use of the target operation in the answer; merely naming an operation or adding irrelevant calculations is insufficient. A supported operation need not produce a correct final answer.

Table A.9: User instructions for text generation.

Dataset	User instruction
CEBaB	Write a restaurant review.
DBpedia	Write a short Wikipedia article.
GoEmotions	Write a short piece of text.
IMDb	Write a short piece of text.

Table B.1: Number of steered outputs per model and training seed.

Dataset	Fine-grained	Coarse-grained	QA steering
CEBaB	80	80	—
DBpedia	140	120	—
GoEmotions	280	80	—
IMDb	260	120	—
GSM8K	—	—	50
ToxicChat	—	—	10

Fluency assessment. Fluency is scored on the five-point scale in Table B.2. It assesses linguistic coherence and readability separately from factual correctness, mathematical correctness, and response safety.

Table B.2: Fluency scoring criteria.

Score	Description
1	Incoherent text or repetitive fragments with little readable structure.
2	Major coherence or grammatical problems that substantially hinder understanding.
3	Understandable text with noticeable coherence, grammatical, or repetition problems.
4	Coherent and readable text with minor issues.
5	Natural, coherent, and fluent text.

C Additional Results

C.1 Effect of Intervention Reconstruction

To examine intervention reconstruction, we train a variant of H-CBLM with $\lambda_s = 0$, retaining hierarchical gating and the remaining objectives. The comparison uses three training seeds on CEBaB, GoEmotions, and DBpedia, with $\tau = 100$ at inference and the original dataset-specific generation limits. As shown in Table C.1, reconstruction improves JS and fluency on all three datasets. On DBpedia, removing reconstruction increases Hit but substantially lowers fluency and JS, illustrating that frequent target expression alone does not establish effective generation control.

C.2 Sensitivity to Intervention Strength

We vary the inference intervention value over $\{10, 50, 100, 150, 200, 300, 500\}$ while keeping the trained checkpoints fixed. H-CBLM is trained with reconstruction value 100, whereas the adapted CB-LLM baseline is trained without reconstruction. All conditions, including natural generation, use a maximum of 100 new tokens. Consequently, the results at $\tau = 100$ can differ from the main experiments, which use longer generation limits.

Tables C.2–C.4 show that stronger interventions do not produce uniformly better joint success. For H-CBLM, moderate increases can improve target expression, while larger values eventually reduce fluency and JS. The strength yielding the highest reported JS differs across datasets. These results characterize sensitivity to intervention strength; the main experiments retain the common value of 100.

Table C.1: Ablation of intervention reconstruction with hierarchical gating retained. Results are mean ± standard deviation across three seeds. F1, Hit, and JS are percentages.

Dataset	Variant	Macro-F1↑	PPL↓	Hit↑	Fluency↑	JS↑
CEBaB	Full	81.02 ± 0.20	$\mathbf{111.97 \pm 1.46}$	$\mathbf{60.00 \pm 3.31}$	$\mathbf{4.56 \pm 0.08}$	$\mathbf{52.92 \pm 3.15}$
CEBaB	Without reconstruction	$\mathbf{81.64 \pm 0.48}$	113.20 ± 2.79	29.58 ± 6.88	3.11 ± 0.46	11.67 ± 4.73
GoEmotions	Full	$\mathbf{11.41 \pm 2.32}$	$\mathbf{120.55 \pm 16.55}$	$\mathbf{55.00 \pm 0.71}$	$\mathbf{4.31 \pm 0.02}$	$\mathbf{50.24 \pm 0.21}$
GoEmotions	Without reconstruction	10.80 ± 4.56	184.36 ± 69.59	21.67 ± 2.53	3.45 ± 0.33	11.07 ± 4.39
DBpedia	Full	$\mathbf{89.86 \pm 2.51}$	$\mathbf{11.67 \pm 0.16}$	54.29 ± 5.85	$\mathbf{3.66 \pm 0.03}$	$\mathbf{41.43 \pm 3.78}$
DBpedia	Without reconstruction	87.80 ± 3.58	11.83 ± 0.45	$\mathbf{78.33 \pm 3.60}$	1.90 ± 0.07	0.24 ± 0.41

Table C.2: CEBaB intervention-strength sweep with a 100-token generation cap. Results are mean ± standard deviation across three seeds.

	CB-LLM			H-CBLM		
τ	Hit (%)	Fluency	JS (%)	Hit (%)	Fluency	JS (%)
Natural	23.33 ± 3.82	4.53 ± 0.30	21.25 ± 5.73	22.08 ± 3.15	4.60 ± 0.26	20.00 ± 5.00
10	21.25 ± 1.25	4.61 ± 0.11	19.58 ± 1.91	25.00 ± 5.45	4.60 ± 0.19	22.50 ± 6.96
50	28.33 ± 4.39	4.52 ± 0.25	25.00 ± 5.73	39.58 ± 4.73	4.69 ± 0.13	37.08 ± 5.91
100	38.75 ± 8.75	3.18 ± 0.47	12.08 ± 7.11	61.25 ± 4.51	4.55 ± 0.09	54.58 ± 5.91
150	23.75 ± 5.45	2.33 ± 0.24	2.08 ± 0.72	82.50 ± 4.51	4.25 ± 0.02	64.58 ± 7.32
200	13.33 ± 10.41	1.76 ± 0.31	0.83 ± 1.44	80.42 ± 3.15	3.32 ± 0.07	34.17 ± 1.91
300	4.58 ± 7.94	1.50 ± 0.25	0.00 ± 0.00	60.00 ± 13.23	1.85 ± 0.18	3.75 ± 3.31
500	1.67 ± 2.89	1.26 ± 0.16	0.00 ± 0.00	15.42 ± 11.61	1.34 ± 0.11	0.00 ± 0.00

C.3 Additional Recognition and Task Results

Coarse-grained concept recognition. Table C.5 supplements the fine-grained recognition results in the main table. H-CBLM has higher mean coarse-grained Macro-F1 on CEBaB, DBpedia, and GoEmotions, while the baseline has a slightly higher mean on IMDb.

Native task performance. We additionally evaluate unsteered performance on the original QA-related tasks. GSM8K uses all 1,319 official test questions, greedy generation with a 384-token limit, and strict numeric exact match for the final answer following `####`. ToxicChat uses all 5,000 official test inputs and predicts input toxicity from the raw `prompt_harmful` logit at the first-answer prediction boundary, thresholded at zero. This evaluation supplies no assistant answer and reports positive-class toxicity F1. Its prediction position differs from the answer-token positions used for concept supervision. Table C.6 reports these task metrics separately from concept recognition and steering success.

C.4 Qualitative Examples

Table C.7 illustrates different fluency levels under coarse-grained intervention. Each pair uses the same dataset, target, training seed, and sampling index. Excerpts preserve the generated wording, and scores refer to the complete outputs.

D Agent Experiment Details

D.1 Task Setup and Training

We use When2Tool (Sun et al., 2026) for an exploratory application separate from the six Llama-based generation benchmarks. The backbone is Qwen3-4B-Instruct-2507, with a 768-dimensional residual representation and LoRA rank 8, scaling parameter 8, and the same target projection modules as in the main experiments. The concept inventory contains two coarse-grained concepts, `non_tool_call` and `tool_call`, and 15 fine-grained tool families. All tool families are associated with `tool_call`; `non_tool_call` has no fine-grained concepts.

Table C.3: GoEmotions intervention-strength sweep with a 100-token generation cap. Results are mean ± standard deviation across three seeds.

	CB-LLM			H-CBLM		
τ	Hit (%)	Fluency	JS (%)	Hit (%)	Fluency	JS (%)
Natural	14.05 ± 2.15	4.47 ± 0.32	13.33 ± 1.35	12.26 ± 0.55	4.47 ± 0.06	12.26 ± 0.55
10	13.33 ± 2.58	4.56 ± 0.11	12.26 ± 2.58	14.64 ± 1.99	4.45 ± 0.05	13.81 ± 2.03
50	15.24 ± 6.61	4.17 ± 0.32	12.50 ± 3.41	32.26 ± 0.90	4.43 ± 0.06	31.07 ± 0.94
100	18.57 ± 5.06	2.91 ± 0.53	4.40 ± 3.00	56.31 ± 0.55	4.34 ± 0.04	51.79 ± 1.29
150	17.14 ± 6.52	2.23 ± 0.45	2.50 ± 1.89	72.02 ± 2.68	4.12 ± 0.04	59.17 ± 3.00
200	15.00 ± 5.75	1.93 ± 0.42	0.60 ± 0.74	79.40 ± 2.68	3.84 ± 0.11	55.83 ± 5.74
300	12.14 ± 4.39	1.71 ± 0.39	0.48 ± 0.41	80.83 ± 3.04	3.11 ± 0.12	28.69 ± 4.52
500	11.55 ± 1.83	1.55 ± 0.30	0.00 ± 0.00	78.69 ± 3.93	2.76 ± 0.07	15.00 ± 3.98

Table C.4: DBpedia intervention-strength sweep with a 100-token generation cap. Results are mean ± standard deviation across three seeds.

	CB-LLM			H-CBLM		
τ	Hit (%)	Fluency	JS (%)	Hit (%)	Fluency	JS (%)
Natural	12.86 ± 0.71	3.53 ± 0.06	6.90 ± 0.41	12.38 ± 1.09	3.50 ± 0.10	4.76 ± 2.30
10	20.48 ± 5.27	3.62 ± 0.08	11.19 ± 2.97	16.90 ± 6.40	3.44 ± 0.03	7.62 ± 5.17
50	77.14 ± 2.58	2.80 ± 0.09	18.10 ± 1.49	36.90 ± 1.49	3.40 ± 0.05	19.76 ± 1.49
100	82.14 ± 7.56	1.88 ± 0.18	0.48 ± 0.41	55.24 ± 5.73	3.61 ± 0.10	37.38 ± 5.07
150	77.14 ± 3.11	1.65 ± 0.16	0.00 ± 0.00	64.29 ± 5.15	3.81 ± 0.08	50.71 ± 4.46
200	76.90 ± 1.49	1.54 ± 0.13	0.00 ± 0.00	75.95 ± 4.06	3.92 ± 0.09	59.76 ± 6.07
300	75.48 ± 6.24	1.48 ± 0.12	0.00 ± 0.00	85.00 ± 3.57	3.62 ± 0.11	52.14 ± 5.39
500	76.90 ± 7.23	1.48 ± 0.14	0.00 ± 0.00	79.05 ± 5.73	2.30 ± 0.07	3.57 ± 1.24

We train one model with seed 42 for 10 epochs using Adam with a learning rate of 5×10^{-5}, one assistant turn per microbatch, and gradient accumulation over four turns.

Concept labels align with the token being predicted. Tokens in a well-formed executed XML call span are labeled as tool use and assigned the executed tool family; tokens outside calls receive negative fine-grained labels. Ambiguous call boundaries and EOS concept labels are excluded from supervision. Intervention reconstruction assigns positive concepts a value of 100 and negative concepts zero, updating only the concept-to-token weights. The residual probe predicts all 17 concepts.

D.2 Rollout and Intervention Protocol

We evaluate all 2,700 test tasks, comprising 2,250 single-hop and 450 multi-hop tasks, with 900 tasks at each difficulty level. Both hop categories use an iterative agent loop; single-hop tasks are not restricted to one assistant turn. Decoding is greedy, with an 8,192-token context limit, at most 512 new tokens per assistant turn, and at most eight assistant turns. Parsed tool calls are executed in the benchmark environment, and tool observations are appended to the conversation before the next turn.

Local coarse-grained suppression. We first generate a natural trajectory and record the complete prompt and first `<tool_call>` marker position for each assistant turn. We then restart the same task. If an assistant-turn prompt exactly matches a recorded natural prompt with a call marker, we set `non_tool_call` to 100 and `tool_call` to zero at the single prediction step corresponding to that marker. The intervention is released afterward. Setting the tool-call parent to zero also closes its associated fine-grained gates.

Prompt matching includes the complete conversation history and tool observations. After an intervened trajectory diverges from its natural counterpart, subsequent turns without matching prompts

Table C.5: Coarse-grained Macro-F1 on unsteered reference text (%). Results are mean ± standard deviation across three seeds.

Dataset	CB-LLM	H-CBLM
CEBaB	84.76 ± 1.98	85.20 ± 1.58
DBpedia	99.28 ± 0.09	99.40 ± 0.10
GoEmotions	34.73 ± 8.81	47.20 ± 12.02
IMDb	66.14 ± 0.66	65.40 ± 1.47

Table C.6: Unsteered native task performance (%). Results are mean ± standard deviation across three seeds.

Dataset	Metric	CB-LLM	H-CBLM
GSM8K	Answer exact match	58.02 ± 0.64	57.95 ± 0.77
ToxicChat	Input-toxicity F1	80.18 ± 4.55	81.82 ± 3.09

receive no intervention. This protocol measures local effects at natural tool-call locations; it does not impose a continuous tool-use ban or implement an autonomous online intervention policy.

D.3 Additional Agent Results

Metric definitions. Task accuracy is determined by the benchmark's task checker. The any-tool rate is the proportion of tasks with at least one recorded call, and mean calls is the total number of recorded calls divided by all tasks, including tasks without calls.

Table D.2 supplements the difficulty-level results in the main text with the fraction of tasks using any tool. Table D.3 reports results by hop category. Suppression reduces both the frequency of tool use and the average number of calls in each category, with an accompanying decrease in task accuracy.

Recognition. Teacher-forced validation yields PPL 1.56, coarse-grained token Macro-F1 of 99.88%, and fine-grained Macro-F1 of 92.80% across all 15 tool families. Fine-grained top-1 accuracy at the first call token is 95.16%.

Table C.7: Examples under coarse-grained intervention. Ellipses in brackets indicate omitted text.

Target	Model	Fluency	Generated excerpt
CEBaB: Food	CB-LLM	3	A bistro off the usual bistro bistro fixerel and on the right side of the price- quality choice. The only downside is that the ambiance is lacking.
CEBaB: Food	H-CBLM	5	One of the top three restaurants in Cleveland and another success for Zach Bruell. The ambiance is lacking though!
IMDb: Entertainment	CB-LLM	1	to the'only in ", one on all that'and'there for'in the'only in the'at the'in the'with the'as the'in'in'it's'so'the'from the'by'in'in'in'in'in'in'in'in'in'in [. . .]
IMDb: Entertainment	H-CBLM	4	In this episode, we meet three great chefs who use local ingredients to create delicious dishes that celebrate the traditions and tastes surrounding charity parties; Ajay Balahuri gets his first restaurant in Mexico City's high-end shopping [. . .]

Table D.1: Agent tool-family concepts and the number of assistant turns with positive labels. All families share the coarse-grained concept `tool_call`.

Fine-grained concept	Training	Validation
CalculatorEnv	78	9
CodeExecutorEnv	26	4
CountingEnv	18	3
DateTimeEnv	28	2
DecodingEnv	48	6
GameRuleEnv	26	2
HashEnv	50	6
HistoricalYearEnv	13	2
ListManipulationEnv	2	0
MatrixEnv	37	4
PrimeEnv	10	2
RegexMatchEnv	9	1
RetrieverEnv	83	12
ScheduleEnv	37	4
StatisticsEnv	33	5

Table D.2: Percentage of tasks using at least one tool, by difficulty. Results use one training seed.

Difficulty	Tasks	Natural	Suppression
Easy	900	26.33	17.78
Medium	900	39.00	26.33
Hard	900	82.11	76.78
Overall	2,700	49.15	40.30

Table D.3: Agent results by hop category. Accuracy and any-tool rate are percentages; mean calls includes tasks without calls. Results use one training seed.

Task category	Condition	Tasks	Accuracy	Any-tool rate	Mean calls
Single-hop	Natural	2,250	63.56	50.18	0.939
Single-hop	Suppression	2,250	60.93	42.36	0.507
Multi-hop	Natural	450	61.11	44.00	2.002
Multi-hop	Suppression	450	60.22	30.00	1.018