

Investigating and Enhancing Concept Induction in LLMs: A Prototype-based Property-augmented Approach

Mingyang Lyu*, Bin Zhang*, Chenlong Bao, Jintao Tang†, Shasha Li†, Ting Wang†
College of Computer Science and Technology, National University of Defense Technology
{lvmingyang17, zhangbin2021, baochenlong, tangjintao, shashali, tingwang}@nudt.edu.cn

Abstract

Concept induction, which infers appropriate abstract categories from concrete instances, is both a core ontology engineering task and a testbed for LLM semantic abstraction. Although LLMs offer broad knowledge and reasoning that may ease expert-driven ontology construction, prior work shows that knowledge deficiencies still limit them in specialized domains, while systematic evaluation of their induction failures remains lacking. To address this gap, we propose a familiarity-controlled evaluation framework that uses knowledge probing to separate item familiarity from concept induction ability. The evaluation reveals a systematic tendency to induce overly general concepts: LLMs often produce semantically correct but overly broad labels, and direct property augmentation improves performance without resolving this granularity mismatch. We further propose PPACI, an ontology-prior-guided approach that builds concept prototypes by integrating concept definitions with member item properties. For each query group, PPACI uses item properties to retrieve local item-level and global group-level candidate concepts as induction anchors. Experiments across multiple LLMs show that PPACI reduces overly general predictions by 17–28 percentage points while maintaining competitive induction quality. Our code and data are publicly available at <https://github.com/Orchideous77/PPACI>.

1 Introduction

Ontology learning transforms raw data into structured knowledge, and concept induction forms a key step in this process. In the ontology learning layered cake model (Buitelaar et al., 2005), it corresponds to the layer that bridges concrete instances and abstract concepts, providing the taxonomic backbone for downstream ontological reasoning.

Given a set of related items, concept induction requires identifying their shared semantics and generating an appropriately specific conceptual label. This makes it not only a core task in ontology engineering, but also a direct manifestation of semantic abstraction. However, because this process requires both domain knowledge and fine-grained semantic reasoning, traditional expert-driven methods suffer from knowledge acquisition bottlenecks, high labor costs, and inconsistency among annotators.

Large language models (LLMs) offer a promising alternative for automating concept induction, as they possess broad semantic knowledge and unified reasoning capabilities. At the same time, concept induction provides a concrete testbed for examining whether LLMs can abstract from instances to concepts at the proper level of granularity. Existing studies have explored LLMs for ontology learning and concept generation, but they often use LLMs as black-box generators in end-to-end pipelines (Lam et al., 2024; An et al., 2024). Samdarshi et al. (2024) further evaluate LLMs on the NYT Connections word game, which involves concept induction, but this task also requires combinatorial grouping and deliberately includes misleading clues, making it difficult to isolate induction ability itself. However, there is still no systematic evaluation framework for assessing the standalone concept induction capability of LLMs under such black-box use. As a result, it remains unclear whether LLM failures stem from missing item knowledge, insufficient induction ability, or an inability to control conceptual granularity.

In this paper, we investigate LLMs’ standalone concept induction capability (Figure 1): given a set of items that belong to the same concept, can the model generate an appropriate conceptual label?¹

*These authors contribute equally to this work.

†Corresponding author.

¹Following Wikidata terminology, we use *item* to refer to an entity or instance, *concept* to refer to a semantic category, class, or type, and *property* to refer to an attribute or relation associated with an item.

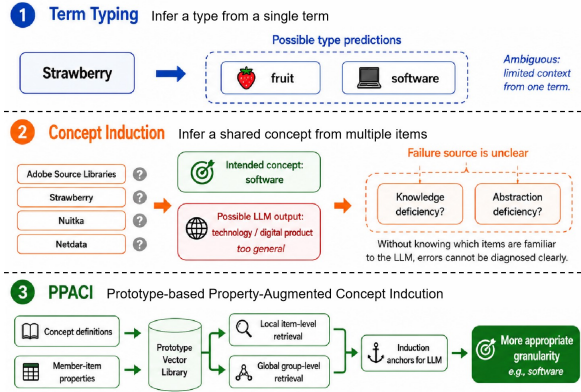


Figure 1: Overview. We move from ambiguous single-term typing to familiarity-controlled multi-item concept induction, and use PPACI to guide LLMs toward concepts at a more appropriate granularity.

To disentangle knowledge deficiency from induction failure, we introduce a familiarity-controlled evaluation framework. First, we use question-answering (QA) knowledge probing (Kandpal et al., 2023) with true-false description pairs to classify each item as known or unknown to the target LLM. Based on this classification, we construct datasets with difficulty levels ranging from all-known to all-unknown groups, and evaluate models under both label-only and property-augmented settings. Across multiple LLMs, our evaluation reveals two key findings: (1) LLMs systematically tend to induce overly general concepts rather than appropriately specific ones, indicating difficulty in controlling conceptual granularity; (2) item properties provide crucial semantic signals, especially for unfamiliar items, but direct property augmentation still fails to eliminate this group-level granularity mismatch.

Motivated by these findings, we propose PPACI (Prototype-based Property-Augmented Concept Induction), which leverages properties not merely as static context, but as retrieval cues for prototype-based concept guidance. Inspired by Prototype Theory (Langacker, 1987; Reed, 2013) and Idealized Cognitive Model (ICM) (Lakoff, 2024), PPACI constructs a prototype feature vector library over an ontology prior, instantiated with Wikidata (Vrandečić and Krötzsch, 2014) in our experiments. Each concept prototype integrates intensional features from concept definitions and extensional features from member-item properties, serving as a prior representation of the concept. During induction, PPACI retrieves candidate concepts from two complementary perspectives: global candidates based on ag-

gregated group features and local candidates based on individual item features. These candidates are then supplied to the LLM as induction anchors, guiding it toward concepts at an appropriate level of specificity. Experiments show that PPACI effectively mitigates the tendency to induce overly general concepts while maintaining competitive overall performance.

Our main contributions are threefold:

- We introduce a familiarity-controlled evaluation framework for black-box LLM concept induction, which uses QA-based knowledge probing to disentangle item familiarity from induction ability.
- We provide a systematic analysis showing that LLMs tend to induce overly general concepts, and properties improve induction for unfamiliar items, but direct property augmentation does not resolve this granularity mismatch.
- We propose PPACI, a prototype-based property-augmented method that builds a concept prototype vector library and retrieves local and global candidate concepts as induction anchors, reducing overly general predictions by 17–28 percentage points while maintaining competitive overall performance.

2 Related Work

2.1 Term Typing

In the framework of LLMs4OL (Babaei Giglou et al., 2023), item-to-concept abstraction is formulated as term typing, where the model predicts the semantic type of a single lexical term within a target ontology, optionally with contextual information. Existing approaches in the LLMs4OL challenge series mainly rely on fine-tuning (Barua et al., 2024; Peng et al., 2024), few-shot prompting (Rahnamoun and Shamsfard, 2025), and retrieval-augmented generation (RAG) (Abi Akl, 2024; Latipov et al., 2025). Although these efforts have made incremental progress, LLMs still fall short in specialized domains with high knowledge requirements (Babaei Giglou et al., 2024, 2025), partly because domain-specific knowledge is sparse in training data and often evolves faster than model updates. Unlike single-term typing, which is often ambiguous because one term may evoke multiple semantic dimensions, multi-item concept induction provides stronger contextual constraints and better isolates semantic abstraction.

2.2 Concept Induction

The concept layer in the ontology learning cake model involves both clustering items into groups and inducing conceptual labels for these groups. Prior work has mainly emphasized the clustering or taxonomy construction aspect (Zeng et al., 2024; Shen et al., 2025), with TaxoRankConstruct further noting the importance of properties for conceptualization quality (Marchenko and Dvoichenkov, 2024). Different from taxonomy construction, which focuses on organizing concepts, our work focuses on discovering and naming concepts for given item groups. Integrated methods such as Lloom (Lam et al., 2024) and Loop (An et al., 2024) incorporate LLMs into concept induction workflows, but they typically use LLMs as black-box generators that map item lists directly to conceptual labels. They therefore do not systematically evaluate LLMs’ standalone induction capability or explore how to improve it. This gap motivates our work: we isolate, evaluate, and enhance LLM concept induction as a specific subtask.

3 Investigating LLMs’ Ability

As discussed in Section 1, applying LLMs to concept induction faces two fundamental challenges: the lack of specific knowledge and systematic evaluation of induction capabilities. To deal with these challenges, we design an evaluation framework based on knowledge probing that enables investigation under conditions where LLMs either possess relevant item knowledge or do not, and helps identify patterns in LLM induction behavior.

3.1 Task Formulation

We define the concept induction task as a generative transformation process. Given a query set of items, the goal is to produce a natural language descriptor of a concept that generalizes these items at the most specific level of granularity. Formally, let $\mathcal{S} = \{i_1, i_2, \dots, i_n\}$ be the input set consisting of n specific items from the item space $\mathcal{I}$. The task is to define a generative mapping $f : \mathcal{I}^n \rightarrow \mathcal{C}$, where $\mathcal{C}$ denotes the space of all possible natural language sequences for concepts. In the evaluation process, this mapping is realized by an LLM that produces the optimal concept sequence $c^* \in \mathcal{C}$:

$$c^* = M(i_1, i_2, \dots, i_n) \quad (1)$$

Following the NYT Connections setting, this paper sets $n = 4$.

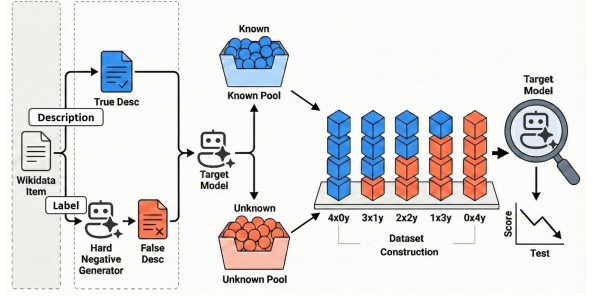


Figure 2: Evaluation framework overview. We employ QA probing to classify items as known or unknown to the target LLM, and then construct datasets with five difficulty levels (all known ($4x0y$) to all unknown ($0x4y$)) based on varying compositions of known and unknown items for subsequent systematic evaluation.

3.2 Evaluation Framework

The overall evaluation framework consists of three steps to systematically assess LLMs’ concept induction capabilities (Figure 2), including knowledge probing, dataset construction and evaluation. To investigate how the target LLM’s familiarity with each test item affects concept induction performance, we utilize a pairwise QA knowledge probing method (Kandpal et al., 2023) to classify items as known or unknown, enabling dataset construction and evaluation across different item familiarity conditions. This requires a knowledge graph that provides concept–item hierarchical relations with sufficient item-level semantic information. In this work, we use Wikidata as the underlying knowledge source, while the framework itself is not intrinsically tied to Wikidata.

Knowledge Probing. The key to the pairwise QA knowledge probing method is generating high-quality false descriptions which are semantically reasonable but factually inaccurate. For each item in Wikidata, we obtain its label and description, with the original description serving as the true description (*true_desc*). To generate a false description (*false_desc*), we employ a two-step process using an auxiliary LLM as Hard Negative Generator. First, we input the item’s label and ask what concept it appears to belong to. Second, the generator generates a factually incorrect description based on the predicted concept. Given the label and each description separately, the target LLM judges whether *true_desc* and *false_desc* match the label, responding with *True*, *False*, or *Unknown*. Each description is judged three times to improve judgment reliability. An item is classified as known

to the target LLM if it identifies *true_desc* as *True* at least twice and *false_desc* as *False* at least twice; otherwise, it is classified as unknown.

Dataset Construction. We leverage taxonomic edges, P31 (instance of) and P279 (subclass of) in Wikidata, to establish relationships between items and concepts. This enables us to identify concepts and their associated item sets. For each group in our dataset, all four items are drawn from the item set belonging to the same concept, ensuring they share a common conceptual parent. The label of this parent concept serves as the ground truth label. Based on the classification from the knowledge probing step, we construct datasets with five difficulty levels by varying the composition of known and unknown items. We denote each difficulty level as *axby*, where *x* represents known items, *y* represents unknown items, and *a*, *b* indicate their respective counts. For example, *4x0y* denotes all items known, *3x1y* denotes three known and one unknown, and *0x4y* denotes all items unknown as shown in Figure 2.

Evaluation Procedure. We evaluate the target LLM by prompting it to induce a concept for each group of four items in the constructed datasets. The LLM’s generated answer is then compared against the ground truth label by an independent judge LLM, which assesses the semantic alignment and assigns one of five ratings. The five ratings and their assigned scores are: **Synonym** (1.0), **More Specific** (0.8), **More General** (0.5), **Related** (0.2) and **Wrong** (0.0). We calculate the average score as a coarse-grained indicator of overall performance. Notably, More Specific predictions receive a score of 0.8 because they may be too narrow to cover all input items, given that Wikidata labels define the reference granularity. More General predictions receive a score of 0.5 because they are technically valid but overly conservative, reflecting insufficient specificity and contradicting the goal of precise concept induction.

3.3 Experimental Setup

We construct a controlled, diverse, and stratified evaluation set of 37 concepts from Wikidata. The full list of concepts is provided in Appendix A. To cover different concept-induction difficulties, these concepts are stratified into four label-transparency levels according to how strongly item labels indicate the target concept, with 8/12/8/9 concepts in the four levels respectively. They span at least nine

primary-domain buckets, including commonsense knowledge, medicine, sports, military, technology, science, arts, history, and fictional domains. The selected concepts also vary substantially in item-set size, ranging from 12 to 2,000 instances, with a median of 250; 11 concepts contain no more than 100 items, while 10 reach the 2,000-item cap. This design enables evaluation across heterogeneous naming patterns, domains, and data-density conditions.

We evaluate four representative LLMs covering different scales and reasoning settings: Llama-3.1-8B-Instruct (Llama) (Grattafiori et al., 2024), Qwen3-8B (Qwen), Qwen3-235B-A22B-Thinking-2507 (Qwen235) (Yang et al., 2025), and DeepSeek-V4-Pro (DSv4) (DeepSeek-AI, 2026). Llama represents an instruction-tuned model without explicit deep-thinking mode, while Qwen and Qwen235 are reasoning models at small and large scales, respectively. DSv4 is included as one of the state-of-the-art open-source frontier models. For the two 8B models, we construct 500 test groups for each difficulty level, and for the two larger models, we construct 200 groups per difficulty level. In the evaluation pipeline, we use Qwen3-Next-80B as the Hard Negative Generator for producing false descriptions during knowledge probing, and GPT-5.2 (OpenAI, 2025) as the judge model for assigning induction-quality ratings. To assess the robustness of this evaluator choice, we further validate GPT-5.2 against human judgments and an alternative LLM judge in Section 5.3.

3.4 Results and Analysis

We conduct experiments under two input configurations to assess the role of property information in concept induction. Table 1 compares the performance across difficulty levels under two settings.

Label-only (LO) Setting. In this baseline configuration, each item is represented solely by its label. We evaluate all four models across all five difficulty levels (*4x0y*, *3x1y*, *2x2y*, *1x3y*, *0x4y*) to comprehensively assess how item familiarity affects concept induction performance.

Property-augmented (PA) Setting. To examine whether property information can enhance concept induction, we conduct additional experiments on the two extreme difficulty levels (*4x0y* and *0x4y*). Items are represented as:

```
{label} [{prop}:{val}] [{prop}:{val}] ...
```

where *prop* represents the property name and *val* represents the property value. All properties are

	Llama	Qwen	Qwen235	DSv4
<i>Label-only (LO)</i>				
4x0y	0.5968	0.6826	0.7070	0.7555
3x1y	0.5750	0.6652	0.7290	0.7560
2x2y	0.5698	0.6464	0.6795	0.7355
1x3y	0.5334	0.6122	0.6210	0.7095
0x4y	0.4942	0.5666	0.6290	0.6565
<i>Property-augmented (PA)</i>				
4x0y	0.6378	0.7352	0.7495	0.7920
0x4y	0.6328	0.6840	0.7160	0.7145

Table 1: Evaluation results under the label-only (LO) and property-augmented (PA) settings. PA experiments are conducted only on the two extreme difficulty levels.

extracted from Wikidata claims after removing explicit taxonomic claims used for dataset construction (P31/P279), organizational metadata, and unsupported value types.

Analysis of the overall scores. Table 1 shows that stronger models generally achieve higher scores within their model-specific familiarity-controlled datasets. While cross-model comparisons are not strictly paired, the trend suggests that semantic abstraction ability also matters beyond item familiarity. Across models, performance generally decreases as the proportion of unknown items increases from 4x0y to 0x4y, confirming that item familiarity remains an important factor. Properties further improve induction performance under both difficulty levels, with larger gains on all-unknown groups. For example, under 0x4y, Llama improves from 0.4942 to 0.6328 and DSv4 from 0.6565 to 0.7145. This indicates that properties provide explicit semantic signals that compensate for missing item knowledge, making unfamiliar items more usable for concept induction.

Analysis of the rating distributions. Beyond overall scores, Figure 3 reveals a recurring pattern: More General predictions occupy a consistently large proportion across models, difficulty levels, and input settings, as highlighted in the figure. Full numerical distributions are provided in Appendix E. Even under the PA 0x4y setting, More General remains high, reaching 40% for Qwen, Qwen235, and DSv4 (43.2%, 40.0%, and 41.5%). Meanwhile, properties mainly improve item recognition by increasing Synonym predictions and reducing Related and Wrong ones. These results indicate that property information helps LLMs understand unfamiliar items, but does not eliminate their tendency to induce overly general concepts. This may reflect a conservative strategy: broader concepts are more

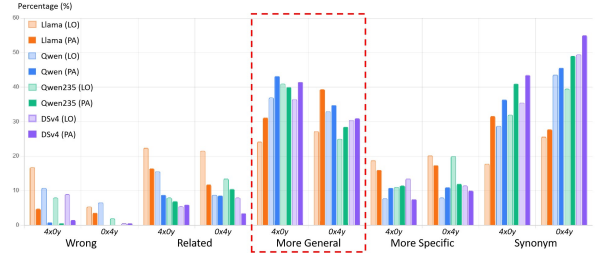


Figure 3: Rating distributions under label-only and property-augmented settings. The More General rating (in the red box) consistently constitutes a major portion across models and conditions, revealing LLMs’ preference for general over specific concepts.

likely to cover all input items and require less fine-grained reasoning, but they fail to satisfy the goal of appropriately specific concept induction.

Analysis of the reasoning process. We further inspect models’ reasoning traces before final prediction, with examples provided in Appendix F. The traces show that LLMs often rely on label semantics and shared naming patterns when item knowledge is uncertain, suggesting that labels themselves function as important item properties. They also support our multi-item formulation: multiple items provide contextual constraints for disambiguation and pattern recognition, making concept induction more focused than single-term typing.

In general, the evaluation reveals two critical insights:

- **Tendency toward overly general concepts.** LLMs systematically prefer to induce higher-level, more general concepts rather than appropriately specific ones.
- **Importance of property information.** Properties, including names (or labels), are crucial to concept induction, but do not mitigate the tendency toward overly general concepts.

4 Prototype-based Property-augmented Concept Induction (PPACI)

Motivated by the two key findings, we ask how property information can be used more effectively to guide LLMs toward appropriately specific concept induction. Based on Prototype Theory (Lan-gacker, 1987; Reed, 2013) and Idealized Cognitive Model (Lakoff, 2024) from cognitive linguistics, we propose PPACI (Prototype-based Property-Augmented Concept Induction), with two key mechanisms: prototype modeling and a dual-level retrieval strategy.

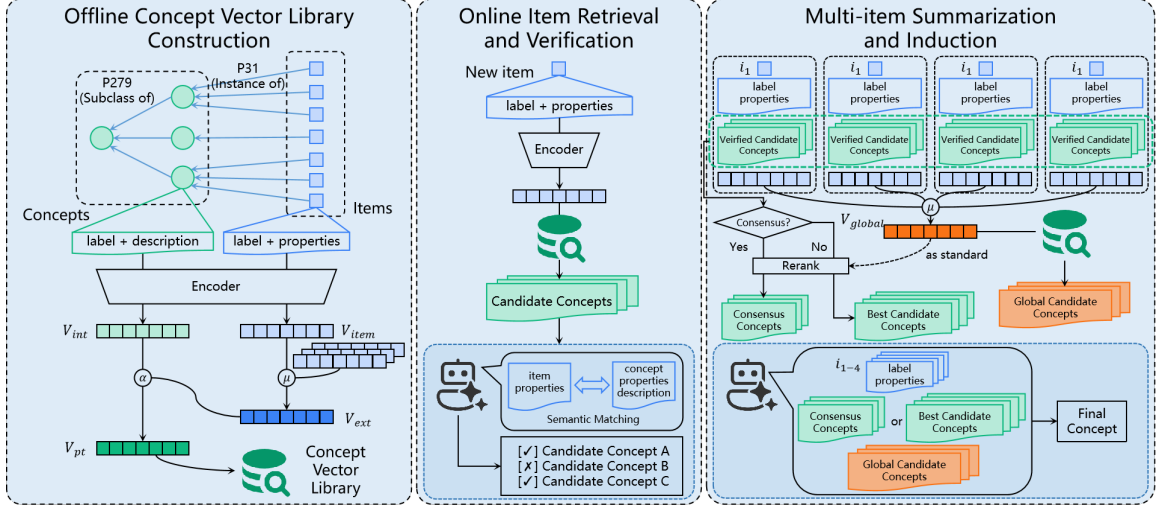


Figure 4: Overview of PPACI. There are three main steps in the workflow: offline concept vector library construction, online item retrieval and verification, and multi-item summarization and induction.

4.1 Method Overview

PPACI is inspired by three intuitions from cognitive linguistics with theoretical details provided in Appendix B. First, the essential role of prior knowledge in human categorization (Wang, 2007) motivates us to introduce an established ontology as structured prior knowledge for LLM-based concept induction, rather than relying solely on the model to generate concepts independently. Second, Prototype Theory (Langacker, 1987; Reed, 2013) motivates us to model each concept with both intensional features from its definition and extensional features from the averaged properties of its member items. Third, based on Idealized Cognitive Model (ICM) (Lakoff, 2024), we assume that the aggregation of an item’s properties can serve as an approximate ICM-based representation of that item, providing semantic features for retrieval.

Following this view, PPACI does not induce concepts in an ontology-free vacuum. Instead, it encodes items and established concepts into a shared vector space where similarity indicates conceptual relatedness. We retrieve candidate concepts by computing cosine similarity between item vectors and concept prototype vectors. As properties introduce item knowledge but fail at the group level, PPACI retrieves candidate concepts from both local item-level and global group-level perspectives. By retrieving and presenting these candidate concepts to the LLM, PPACI anchors the induction process toward appropriate specificity levels. Figure 4 provides an overview of PPACI, which consists of three main steps: offline concept vector library con-

struction, online item retrieval and verification, and multi-item summarization and induction.

4.2 Offline Concept Vector Library Construction

PPACI only requires an ontology prior containing concepts, concept-item links, and item properties, which we instantiate with Wikidata (Section 3.2). Following ICM (Equation 6), we linearize the properties of item i as:

$$\text{Label}:\{\text{label}\}|\{\text{prop}\}:\{\text{val}\}|\dots$$

which is then encoded to produce the item property feature vector $V_{item}(i)$. We concatenate properties before encoding to preserve inter-property interactions. For a concept c , we concatenate its label and description into the sequence:

$$\text{Label}:\{\text{label}\}|\text{Description}:\{\text{desc}\}$$

The Wikidata description usually provides a concise definition. This sequence is encoded as the intensional feature vector $V_{int}(c)$. Then, the extensional feature vector $V_{ext}(c)$ is computed by averaging all member items’ property feature vectors:

$$V_{ext}(c) = \frac{1}{N} \sum_{j=1}^N V_{item}(i_j) \quad (i_j \in \mathcal{I}(c)) \quad (2)$$

where $\mathcal{I}(c)$ is the member item set of concept c and $N = |\mathcal{I}(c)|$. Then, following Prototype Theory, we integrate intensional and extensional features through a weighted average with parameter α :

$$V_{pt}(c) = \text{Norm}(\alpha V_{int}(c) + (1 - \alpha) V_{ext}(c)) \quad (3)$$

where $V_{pt}(c)$ denotes the prototype feature vector of concept c . To facilitate efficient cosine

similarity-based retrieval in the subsequent steps, we normalize $V_{pt}(c)$ to unit length. Considering the influence of N , when concept c has few member items, $V_{int}(c)$ should be more reliable than $V_{ext}(c)$, since these few member items may be unrepresentative. Therefore, α is set as a decay function of N with decay coefficient k :

$$\alpha(N) = \alpha_{base} + (1 - \alpha_{base})e^{-kN} \quad (4)$$

When $N = 0$, $\alpha = 1$, which means $V_{pt}(c)$ depends entirely on the label and description. As more items accumulate, $V_{ext}(c)$ gains weight up to $1 - \alpha$.

4.3 Online Item Retrieval and Verification

Given a new query item i_q for concept induction, we first compute its property feature vector $V_{item}(i_q)$ using the same encoding procedure as during library construction: concatenating the label and properties into a sequence and passing it through the encoder. We then retrieve the top- K nearest concepts from the vector library based on cosine similarity, forming an initial candidate concept set $\mathcal{C}_{init}(i_q)$.

To reduce noise injection and ensure semantic validity, we verify each initial candidate concept through the target LLM. For each candidate concept, we provide the LLM with i_q 's label and properties along with the concept's label, description, and properties, and prompt it to assess whether the item semantically aligns with the concept. The target LLM is supposed to output *True* or *False*. Only concepts that pass this verification step are retained in the verified candidate concept set $\mathcal{C}_v(i_q)$.

4.4 Multi-item Summarization and Induction

For the four items in a test group, $\mathcal{I}_{test} = \{i_j \mid j = 1, 2, 3, 4\}$, each has its property feature vector $V_{item}(i_j)$ and verified candidate concept set $\mathcal{C}_v(i_j)$ after online retrieval and verification. We select final candidate concepts from two complementary perspectives to provide to the target LLM.

Global aspect. We compute the global feature vector by averaging item property feature vectors:

$$V_g(\mathcal{I}_{test}) = \frac{1}{4} \sum_{j=1}^4 V_{item}(i_j) \quad (5)$$

which represents the common characteristics shared across the item group. We then retrieve the top- K' nearest concepts from the concept vector library using $V_g(\mathcal{I}_{test})$, forming the global candidate concept set $\mathcal{C}_g(\mathcal{I}_{test})$.

Local aspect. We analyze the verified candidate sets for consensus patterns. Since there are four items, we consider it a strong consensus case if any concept appears in at least three verified candidate concept sets. Such concepts form the consensus concepts and are reranked by their cosine similarity to $V_g(\mathcal{I}_{test})$ to select the top- K'' concepts as $\mathcal{C}_{con}$. Otherwise, we consider it a weak consensus case and rerank each item's verified candidate concepts by their cosine similarity to $V_g(\mathcal{I}_{test})$ and select the top-ranked concept as the best candidate $c_{i_j}^*$.

Finally, we provide different information to the LLM in different consensus cases. Under strong consensus, we input items along with the consensus concepts $\mathcal{C}_{con}$ and global candidate concepts $\mathcal{C}_g(\mathcal{I}_{test})$. Under weak consensus, we input items along with corresponding best candidate concepts $\{c_{i_j}^* \mid j = 1, 2, 3, 4\}$ and the global candidate concepts $\mathcal{C}_g(\mathcal{I}_{test})$. For items, their labels and properties are provided; for candidate concepts, their labels and descriptions are provided. This dual-level strategy leverages both local retrieval (consensus concepts or best candidate concepts) and global retrieval (global candidate concepts) as reference anchors, rather than a closed label set, to guide the LLM toward appropriate specificity.

5 Experiments

5.1 Experimental Setup

We construct the concept prototype library from Wikidata concepts with at least 10 member items, retaining at most the 2,000 most recently modified items for concepts with excessive members. The resulting library contains 36,303 concepts. To avoid query item leakage into prototype construction, for the 37 evaluation concepts used in Section 3, we compute their extensional feature vectors only from known items, and evaluate PPACI only on the $0x4y$ groups where all query items are unknown to the target LLM. We evaluate the same four target LLMs as in Section 3.3, using 500 test groups for each 8B model and 200 test groups for each larger model. We use BGE-M3 (Chen et al., 2024) for embedding, and set $\alpha_{base} = 4/9$, $k = 0.23$, $K = 5$, and $K' = K'' = 3$. The derivation of α_{base} and k is provided in Appendix C.

We compare PPACI with the PA setting, a Naive RAG baseline, and three ablations. Naive RAG builds a non-prototype index from each concept's label, description and properties, and retrieves candidates with a single query concatenating the four

Methods	Overall Score				More General (%)				Synonym (%)			
	Llama	Qwen	Qwen235	DSv4	Llama	Qwen	Qwen235	DSv4	Llama	Qwen	Qwen235	DSv4
PA	0.6328	0.6840	0.7160	0.7145	31.2	43.2	40.0	41.5	31.6	36.4	41.0	43.5
Naive RAG	0.4890	0.4819	0.6120	0.6930	11.0	8.6	17.0	31.0	20.2	20.2	22.5	34.0
PPACI _{w/o PV}	0.5148	0.4756	0.6305	0.6560	15.6	20.4	32.5	38.0	24.8	21.4	32.0	38.5
PPACI _{w/o LC}	0.6734	0.6786	0.6955	0.7355	13.8	21.8	21.5	30.5	48.2	45.6	47.5	51.0
PPACI _{w/o GC}	0.6736	0.6762	0.7080	0.6955	16.0	28.6	28.0	39.5	36.2	39.0	43.0	35.5
PPACI	0.6808	0.6964	0.7080	0.7535	10.8	14.8	23.0	22.5	49.4	52.6	49.0	55.5

Table 2: Performance comparison and ablation study results on $0x4y$ groups.

items’ labels and properties. PPACI w/o PV uses the non-prototype index but retains dual-level retrieval, while w/o LC removes local candidates and w/o GC removes global candidates.

5.2 Results

Table 2 reports the results on the all-unknown $0x4y$ groups. Compared with the PA baseline, PPACI clearly reduces the tendency to induce overly general concepts while increasing appropriately specific predictions. On average, More General drops from 39.0% to 17.8%, while Synonym increases from 38.1% to 51.6%. This pattern holds across all models. At the same time, PPACI preserves strong overall performance, improving the average score from 0.6868 to 0.7097; the score improves on Llama, Qwen, and DSv4, and remains comparable on Qwen235. Naive RAG also reduces More General predictions, but this reduction comes at the cost of much lower overall score and Synonym rate, indicating much less accurate performance.

The ablation results show that both prototype modeling and dual-level retrieval are important. Removing the prototype vector library (w/o PV) causes the largest degradation, with the average overall score falling to 0.5692 and Synonym to 29.2%, showing the importance of prototype-based concept representation. Removing local candidates (w/o LC) or global candidates (w/o GC) also weakens performance compared with full PPACI, indicating that item-level and group-level candidates provide complementary anchors for inducing appropriately specific concepts.

PPACI differs from standard RAG pipelines because concept induction is not inherently a text retrieval task. In Naive RAG, concepts are retrieved mainly through textual similarity. Such retrieval may return textually similar but conceptually distant candidates, introducing noisy anchors and even hurting performance compared with the PA setting. In contrast, PPACI retrieves concept-level knowl-

edge representations. Thus, retrieval in PPACI is not used as a standard RAG pipeline, but as a way to inject structured prior taxonomic knowledge and provide granularity-aware induction anchors.

5.3 Analysis and Discussion

Beyond the main performance results, we conduct further analyses to assess the reliability of our evaluation and better understand the role of property information. We first validate the LLM-based evaluator through human annotation and an alternative judge model. We then analyze the effect of property quantity from both concept-level and query-group-level perspectives, followed by a qualitative discussion of how property quality influences PPACI’s prototype construction and retrieval. Details are provided in Appendix G.

Judge Reliability. Since all induction outputs are rated by GPT-5.2, we further validate whether our findings are robust to evaluator choice. First, we manually audit 200 predictions covering different target models, methods, difficulty levels, and rating categories. Taking the manual labels as reference, GPT-5.2 achieves 80.5% exact agreement and a Cohen’s (κ) of 0.743, with a macro F1 of 0.767. Agreement is particularly strong for the two categories central to our analysis: Synonym achieves an F1 of 0.908 and More General an F1 of 0.850. Second, we re-evaluate all 2,800 $0x4y$ predictions under the PA and PPACI settings using DeepSeek-V4-Pro (DSv4) with the same rubric. GPT-5.2 and DSv4 agree on 88.86% of the ratings, with a Cohen’s κ of 0.8413 and a score-weighted κ of 0.9091. More importantly, the main trend remains unchanged under DSv4 evaluation: across the 1,400 predictions for each method, PPACI reduces More General from 38.7% to 17.0% while increasing Synonym from 36.4% to 50.5%. Since DSv4 is also one of our target models, we additionally exclude DSv4-generated predictions and obtain nearly identical agreement (88.67%, Cohen’s

$\kappa=0.8406$, weighted $\kappa=0.9093$). These results indicate that both the observed over-generalization tendency and PPACI’s improvement are robust across human and model-based evaluation rather than artifacts of a single judge.

Property Analysis. We first examine whether the amount of available property information is associated with induction performance from two complementary perspectives.

At the concept level, we compute the average number of properties of member items used to construct the extensional representation of each target concept, and group predictions into three bins: ([0,3)), ([3,7)), and ([7,+)). Under PPACI, the Synonym rate increases from 50.8% in the low-property bin to 60.3% in the high-property bin, while Wrong decreases sharply from 13.2% to 1.0%. However, More General does not decrease monotonically with property quantity, reaching its highest rate of 20.7% in the middle bin. This suggests that richer property coverage generally supports more accurate concept representations, but property quantity alone does not determine whether the induced concept has the appropriate granularity.

At the query-group level, we compute the average number of properties associated with the four query items, which directly affects the information available to PA and the retrieval signals used by PPACI. PA exhibits consistently high More General rates across property-count bins, reaching 32.9%, 45.9%, and 39.9%, respectively. In contrast, PPACI substantially reduces these rates to 12.5%, 21.1%, and 13.1%, while achieving higher Synonym rates in all three bins (49.1%–55.1% versus 33.3%–39.9% for PA). Meanwhile, PPACI’s Wrong rate decreases markedly as more properties become available, from 12.9% to 0.5%. These results indicate that the reduction in over-generalization cannot be explained simply by having more property information: how properties are organized and used as retrieval guidance is also critical. The relatively higher error rate under sparse-property conditions further motivates examining the quality of the available properties.

However, retrieving knowledge representations does not make PPACI noise-free. Since properties are central to PPACI, its reliability depends on property quality. Wikidata properties are not uniformly informative: some are verbose and contain irrelevant details, while others are sparse, weakly discriminative, or even missing. Such noisy properties

may retrieve misleading candidates, causing some predictions to shift toward Related or Wrong rather than directly becoming Synonym. This also explains why the large reduction in More General predictions is not always accompanied by an equally large increase in the coarse-grained overall score.

Property quality affects concept-level and item-level representations differently. For concept prototype construction, noise can be mitigated by averaging over many member items and integrating definitions with member-item properties. In contrast, online item-level retrieval depends directly on the properties of each query item, making it more sensitive to verbose, sparse, or missing properties. In extreme cases where item properties provide little reliable information, PPACI may degrade toward label-based retrieval with noisy candidates. This explains why both local and global candidates are necessary. They provide complementary signals that help reduce the impact of noise and guide LLMs toward appropriately specific concepts.

6 Conclusion

To evaluate LLMs’ concept induction capabilities, we define a task where models induce a concept from several related items. Using QA probing to distinguish known and unknown items, we find that LLMs systematically prefer overly general concepts. This finding motivates our question: how can property information guide models toward greater specificity? Inspired by Prototype Theory and ICM, we propose PPACI, which models concept prototypes and retrieves local and global candidate concepts from a prototype feature vector library and uses them to anchor LLM induction. Experiments show PPACI substantially reduces LLMs’ preference for general concepts while maintaining competitive overall performance.

Limitations

Three key limitations point to future directions. First, our binary knowledge classification is overly strict. Items are classified as known only when LLMs completely understand all details, yet partial knowledge of general categories could still help concept induction. Developing fine-grained knowledge probing to assess varying levels of item familiarity would enable deeper investigation of how partial knowledge affects induction performance.

Second, PPACI depends on the coverage and property quality of the external ontology prior,

and our current property processing is relevance-agnostic. After removing explicit taxonomic and metadata claims, all retained properties are treated equally despite their varying relevance to concept induction. Future work should develop more robust filtering mechanisms and property importance assessment methods that identify critical properties and assign differential weights, improving both retrieval precision and induction accuracy.

Third, our evaluation uses a coarse-grained rating scheme to explore general patterns in LLM concept induction, rather than fine-grained semantic distances between predicted and optimal concepts. While the five-way categorization captures major error types such as overly general predictions, it does not quantify how far a prediction deviates from a human-selected concept or from the target concept in the ontology. Future work should incorporate finer human evaluation and taxonomy-based or semantic-distance-based metrics to better measure conceptual mismatch.

Acknowledgments

This work was supported by the Key Research and Development Project of Hunan Province (No. 2025JK2119), Foundation of NUDT (HQKYZH2025KD004), the Leading Science and Technology Innovation Talents Program of Furong Project (2025RC1048).

References

- Hanna Abi Akl. 2024. [DSTI at LLMs4OL 2024 Task A: Intrinsic versus extrinsic knowledge for type classification: Applications on wordnet and geonames datasets](#). *Open Conference Proceedings*, 4:93–101.
- Wenbin An, Wenkai Shi, Feng Tian, Haonan Lin, Qianying Wang, Yaqiang Wu, Mingxiang Cai, Luyan Wang, Yan Chen, Haiping Zhu, and Ping Chen. 2024. [Generalized category discovery with large language models in the loop](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8653–8665, Bangkok, Thailand. Association for Computational Linguistics.
- Hamed Babaei Giglou, Jennifer D’Souza, and Sören Auer. 2023. [LLMs4OL: Large language models for ontology learning](#). In *The Semantic Web – ISWC 2023*, pages 408–427.
- Hamed Babaei Giglou, Jennifer D’Souza, Nandana Mihindukulasooriya, and Sören Auer. 2025. [LLMs4OL 2025 overview: The 2nd large language models for ontology learning challenge](#). *Open Conference Proceedings*, 6.
- Hamed Babaei Giglou, Jennifer D’Souza, and Sören Auer. 2024. [LLMs4OL 2024 overview: The 1st large language models for ontology learning challenge](#). *Open Conference Proceedings*, 4:3–16.
- Adrita Barua, Sanaz Saki Norouzi, and Pascal Hitzler. 2024. [DaSeLab at LLMs4OL 2024 Task A: Towards term typing in ontology learning](#). *Open Conference Proceedings*, 4:77–84.
- Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini. 2005. [Ontology learning from text: An overview](#). *Ontology learning from text: Methods, evaluation and applications*, pages 3–12.
- Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. [M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2318–2335, Bangkok, Thailand. Association for Computational Linguistics.
- DeepSeek-AI. 2026. [Deepseek-v4: Towards highly efficient million-token context intelligence](#). *arXiv preprint arXiv:2606.19348*.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. [The llama 3 herd of models](#). *arXiv preprint arXiv:2407.21783*.
- Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. [Large language models struggle to learn long-tail knowledge](#). In *Proceedings of the 40th International Conference on Machine Learning, ICML’23*.
- George Lakoff. 2024. *Women, fire, and dangerous things: What categories reveal about the mind*. University of Chicago press.
- Michelle S Lam, Janice Teoh, James A Landay, Jeffrey Heer, and Michael S Bernstein. 2024. [Concept induction: Analyzing unstructured text with high-level concepts using lloom](#). In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pages 1–28.
- Ronald W Langacker. 1987. *Foundations of cognitive grammar: Volume I: Theoretical prerequisites*, volume 1. Stanford University Press.
- Insan-Aleksandr Latipov, Mike Holenderski, and Nirvana Meratnia. 2025. [IRIS at LLMs4OL 2025 Tasks B, C and D: Enhancing ontology learning through data enrichment and type filtering](#). *Open Conference Proceedings*, 6.
- Oleksandr Marchenko and Danylo Dvoichenkov. 2024. [Taxorankconstruct: A novel rank-based iterative approach to taxonomy construction with large language models](#). In *Proceedings of the Information Technology and Implementation Workshop: Intelligent Systems and Security (IT&I-WS 2024: ISS)*.

OpenAI. 2025. Introducing GPT-5.2. <https://openai.com/index/introducing-gpt-5-2/>.

Yixin Peng, Yongli Mou, Bozhen Zhu, Sulayman Sowe, and Stefan Decker. 2024. RWTH-DBIS at LLMs4OL 2024 Tasks A and B: Knowledge-enhanced domain-specific continual learning and prompt-tuning of large language models for ontology learning. *Open Conference Proceedings*, 4:49–63.

Rashin Rahnamoun and Mehrnosh Shamsfard. 2025. SBU-NLP at LLMs4OL 2025 Tasks A, B, and C: Stage-wise ontology construction through llms without any training procedure. *Open Conference Proceedings*, 6.

Stephen K Reed. 2013. *Psychological processes in pattern recognition*. Academic Press.

Prisha Samdarshi, Mariam Mustafa, Anushka Kulkarni, Raven Rothkopf, Tuhin Chakrabarty, and Smaranda Muresan. 2024. Connecting the dots: Evaluating abstract reasoning capabilities of LLMs using the New York Times connections word game. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21219–21236, Miami, Florida, USA. Association for Computational Linguistics.

Yanzhen Shen, Yu Zhang, Yunyi Zhang, and Jiawei Han. 2025. A unified taxonomy-guided instruction tuning framework for entity set expansion and taxonomy expansion. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 3208–3220, Vienna, Austria. Association for Computational Linguistics.

Friedrich Ungerer and Hans-Jorg Schmid. 2013. *An introduction to cognitive linguistics*. Routledge.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Yin Wang. 2007. *Cognitive Linguistics*. Shanghai Foreign Language Education Press, Shanghai, China.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Qingkai Zeng, Yuyang Bai, Zhaoxuan Tan, Shangbin Feng, Zhenwen Liang, Zhihan Zhang, and Meng Jiang. 2024. Chain-of-layer: Iteratively prompting large language models for taxonomy induction from limited examples. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, CIKM '24*, page 3093–3102.

A Details for Chosen Concepts

This appendix presents details for the concepts we chose from Wikidata for dataset construction. The

selected 37 concepts encompass a broad spectrum of domains, ensuring a comprehensive evaluation across diverse data distributions. To capture the variation in label informativeness, we stratify these concepts into four categories based on the semantic transparency of their labels. As shown in Table 9, this taxonomy reflects a progression from directly interpretable labels to highly abstract labels.

B Cognitive Linguistic Foundation

In cognitive linguistics, the process analogous to concept induction is categorization. Categorization is a mental process grounded in experiential interaction between subjects and objects, through which humans subjectively generalize and classify external entities, events, and phenomena (Wang, 2007). This perspective emphasizes the critical role of prior knowledge in human categorization. Thus, we argue that LLM-based concept induction should be grounded in a prior ontology rather than independent generation. This motivates us to utilize an established ontology as induction guidance.

Prototype Theory describes the mechanisms underlying human categorization. A prototype admits two complementary interpretations. (1) Typical member (Langacker, 1987): a prototype is a typical instance of a category; (2) Schematic representation (Reed, 2013): a prototype represents the average properties or central tendency of a category across category members. These two perspectives give rise to two approaches to categorization: categorization based on prototype members and categorization based on schematic representations. When children construct categories, they initially identify category members by comparing them to typical members, then progressively extract commonalities to establish low-level schematic representations (Langacker, 1987). As more category members are encountered, they gradually build higher-level schematic representations that capture increasingly abstract generalizations. This cognitive process inspires our approach to modeling concepts in an existing ontology. For each concept, both its explicit definition or description and the common or average properties of its member items should be leveraged as intensional and extensional features. By integrating these two sources of information, we can construct a high-level schematic representation as the prototype that captures the concept’s essence from both top-down and bottom-up perspectives. Moreover, the similarity between an item and a

concept’s prototype can serve as a useful signal for determining concept membership.

While Prototype Theory guides how to model concepts, Idealized Cognitive Model (ICM) provides a theoretical foundation for modeling items. A Cognitive Model (CM) represents all relevant knowledge about a particular domain, stored in human memory based on related scenarios and contexts (Ungerer and Schmid, 2013). An ICM builds upon multiple CMs to form a complex structured whole (Lakoff, 2024). It is an abstract, unified, and idealized understanding that integrates diverse knowledge components. Formally, an ICM can be expressed as the integration of constituent CMs:

$$ICM = CM_1 + CM_2 + \dots + CM_n \quad (6)$$

As each item property is a kind of knowledge representation, we assume that each property-value pair can be approximated as a CM. Thus, the aggregation of all properties constitutes an item’s ICM.

A key hypothesis is that items typically lie closer to specific subconcepts than to parent concepts in the vector space. Parent concept vectors, computed as averages over multiple subconcepts, are naturally pulled away by other subconcepts. Even when an item is closer to a parent concept, this suggests the item is atypical within its subconcept, making the more general candidate acceptable.

C Hyperparameters

In Section 5.1, we set $\alpha_{base} = 4/9$ and $k = 0.23$. Due to BGE-M3’s unit-norm outputs, $\|V_{item}\| = 1$ and $\|V_{int}\| = 1$. Since V_{ext} is the average of V_{item} , $\|V_{ext}\| \in [0, 1]$, reflecting item cohesion. Higher $\|V_{ext}\|$ indicates more concentrated and similar items with stronger commonalities, lending greater reliability to V_{ext} . Meanwhile, $\|V_{int}\| = 1$, so simple arithmetic averaging ($\alpha = 0.5$) inherently biases V_{pt} toward the direction of V_{int} , which means greater influence from V_{int} than V_{ext} . However, when items are highly cohesive ($\|V_{ext}\|$ close to 1), V_{ext} should carry greater influence. We therefore set α such that when $\|V_{ext}\| = l$, both components have equal effective magnitude:

$$\alpha \times 1 = (1 - \alpha) \times l \quad (7)$$

Assuming equal influence at $l = 0.8$, we obtain $\alpha_{base} = 4/9$. For the decay coefficient k , we assume that $\alpha(10) \approx 0.5$, and thus k is calculated to be 0.23.

D Prompts

This appendix lists the prompts used for LLM calls..

The prompts for Hard Negative Generator to generate false descriptions:

Step 1

You are a linguistic analyzer.
Analyze the given term and list 3-4 distinct categories it could plausibly belong to based on its name structure or polysemy.
Include both Real categories and Plausible sounding categories.
Output ONLY a JSON list of strings, e.g., ["Cat A", "Cat B"].
Term: "{item_label}"

Step 2

You are a 'Hard Negative' Generator.
Your goal is to create a brief plausible-sounding but **FACTUALLY INCORRECT** description for an entity.
You will be provided with the **REAL TRUTH**. You must use it as a Negative Constraint—do not repeat its facts.
TARGET ENTITY: "{entity}"
TARGET FAKE CATEGORY: "{target_type}"
[REFERENCE - THE REAL TRUTH]
REAL CATEGORY: "{true_type}"
"{true_desc}"
(Warning: The information above is TRUE. You must NOT output this.)
[GENERATION RULES]
1. Define "{entity}" strictly as a "{target_type}".
2. ****Contrastive Fabrication Strategy****:
- Compare with [REFERENCE].
- If [REFERENCE] mentions a specific ****Year****, fabricate a DIFFERENT decade.
- If [REFERENCE] mentions a specific ****Author / Company****, fabricate a DIFFERENT one.
- If [REFERENCE] mentions a specific ****Genre / Function****, fabricate an OPPOSITE one.
3. Even if Target Category == Real Category, create an "Alternate Universe" version (change attributes entirely).
[OUTPUT FORMAT]
- A single, confident encyclopedic sentence.
- Output ONLY the description within 20 words without "TARGET ENTITY is a" at the beginning.
Target Format Examples:
- Input: "Elden Ring" -> Output: 1980s high-fantasy novel written by Ursula K. Le Guin about a shattered magical circle.
- Input: "Call of Duty" -> Output: gritty black-and-white anti-war film released in 1955 depicting a soldier’s regret.

The concept induction prompt under PA setting:

Output ONLY the concept name.
Do not include any prefixes, suffixes, punctuation, or explanation.
You are an expert in Ontology Induction. I have 4 entities:

```
{i}.{item_label} [{item_props}]
```

Please induce the most specific common concept (hypernym) that accurately generalizes these 4 entities.

The prompt under LO setting is similar but without `[{item_props}]`.

The prompt for answer rating:

You are an expert Ontology Evaluator.

Your task is to compare a "Predicted Concept" against a "Ground Truth Label" for a group of 4 entities.

Determine the semantic relationship between the Prediction and the Label.

IMPORTANT CONSTRAINTS:

1. The "Context Entities" are provided ONLY for disambiguation (to help you understand the specific domain of the Label).

2. DO NOT evaluate whether the entities actually belong to the Predicted Concept. Assume the prediction is based on them.

3. DO NOT generate your own concept for the entities. Focus strictly on the relationship between Prediction and Label.

Output JSON with:

1. "category": One of ["Synonym", "More Specific", "More General", "Related", "Wrong"]

2. "reason": A concise explanation up to 50 words.

Definitions & Examples:

1. "Synonym": Identical meaning, abbreviations, or singular/plural variations.

- Label: "Film" | Pred: "Movie" -> Synonym

- Label: "Software" | Pred: "Computer Software" -> Synonym

2. "More Specific": Prediction is a valid HYPONYM (subset) of Label. It is narrower but accurate for the entities.

- Label: "Software" | Pred: "Open Source Software" -> More Specific (If entities are indeed open source)

- Label: "Animal" | Pred: "Cat" -> More Specific

3. "More General": Prediction is a HYPERNYM (superset) of Label. It is correct but too broad.

- Label: "Novel" | Pred: "Book" -> More General

- Label: "iPhone" | Pred: "Electronic Device" -> More General

4. "Related": Concepts are in the same domain but strictly different (e.g., sibling concepts, part-whole).

- Label: "Car" | Pred: "Engine" -> Related (Part-of relation)

- Label: "Song" | Pred: "Music Album" -> Related (Container relation)

5. "Wrong": Completely unrelated or factually incorrect.

- Label: "Song" | Pred: "Food" -> Wrong

Context Entities: `{item_labels}`

Ground Truth Label: `"{label}"`

Predicted Concept: `"{answer}"`

Analyze the relationship of Predicted Concept relative to Ground Truth Label.

The prompt for candidate verification in PPACI:

You are a knowledge graph expert. Respond in JSON.

Task: Verify if the Item is an instance or subclass of the Candidate Type.

[Item Info]

Label: `{item_label}`

Properties: `{item_props}`

[Candidate Type Info]

Label: `{cand_label}`

Description: `{cand_desc}`

Claims: `{cand_props}`

Requirement:

1. Analyze the semantic relationship between the Item and the Candidate Type.

2. Determine if the Item logically belongs to this Type.

3. Output JSON with "result" (boolean) and "reason" (short string).

JSON Output Example:

```
{ "result": true, "reason": "The item is a specific model of car, which fits the type Automobile." }
```

The prompt for final induction in PPACI:

Strong consensus case

You are a Knowledge Graph expert. Output valid JSON only.

You are a concept induction expert. We have identified at least one STRONG CONSENSUS candidate concept for a group of 4 entities.

You are also provided with several global candidates related to the entire 4 entities.

Please take the consensus category and global candidates as reference and induce the most specific common concept (hypernym) that accurately generalizes these 4 entities.

Entities

```
{i}.{item_label} [{item_props}]
```

Consensus Category

```
{i}.{consensus_label}: {consensus_desc}
```

Global Candidates

```
{i}.{global_label}: {global_desc}
```

Final Output

Output a valid JSON object:

- "answer": The induced specific concept name (String).

- "reason": A concise reason (String, within 50 words).

Weak consensus case

You are a Knowledge Graph expert. Output valid JSON only.

You are a concept induction expert. You are given 4 entities with their properties in '[...]' and potentially DIFFERENT local candidates.

You are also provided with several global candidates related to the entire 4 entities.

Please take the global and local candidates as reference and induce the most specific common concept (hypernym) that accurately generalizes these 4 entities.

Entities & Local Candidates

```
{i}. Entity: {item_label} [{item_props}]
```

```
* Local Candidate: {cand_label}: {cand_desc}
```

Global Candidates

```
{i}.{global_label}: {global_desc}
```

Final Output

Output a valid JSON object:

- "answer": The induced specific concept name (String).

- "reason": A concise reason (String, within 50 words).

E Rating Distribution

This appendix reports the full statistics about the rating distribution for Section 3. Table 3 shows the results for 4x0y and 0x4y groups under both LO and PA settings, while Table 4 shows the results for 3x1y, 2x2y and 1x3y groups only under LO setting. In these tables, Syn represents Synonym, M.S. represents More Specific, M.G. represents More General, Rel represents Related and Wrg represents Wrong. Across all difficulties and settings, More General accounts for a large proportion.

F Reasoning Examples

This appendix presents two examples of reasoning content during induction under the LO setting.

Example 1

Model: Qwen3-235b-a22b-thinking-2507
Items: UPSFRi005-A-1, PENN045i-25-12, STAN254i-647C3, WIC05i-127-325 (all unknown)
Label: induced pluripotent stem cell line
Answer: induced pluripotent stem cell line
Reasoning content:
Okay, the user is asking me to act as an expert in Ontology Induction.
.....
Looking at the entities, they all seem to follow a similar naming pattern. Let me break them down:
- UPSFRi005-A-1: Starts with "UPSFR", then "i", followed by numbers and letters.
- PENN045i-25-12: "PENN" prefix, "045i", then hyphenated numbers.
- STAN254i-647C3: "STAN" prefix, "254i", then "647C3".
- WIC05i-127-325: "WIC" prefix, "05i", then hyphenated numbers.
I notice consistent elements: each has a multi-letter prefix (like UPSFR, PENN, STAN, WIC), followed by numbers, then "i", and more alphanumeric characters with hyphens.
.....
This "i" seems crucial. In many contexts, especially in scientific or technical naming, "i" could stand for "induced" or "iPS" (induced pluripotent stem cells). I recall that in stem cell biology, cell lines often have identifiers like this. For instance, iPS cells are commonly labeled with prefixes indicating the institution or lab, followed by codes.
Let me verify this. Searching my knowledge: Yes, in stem cell registries, such as those from the International Stem Cell Initiative (ISCI) or similar databases, cell line names often start with institution codes. For example:
- UPSFR might be Université de Paris-Saclay or something similar
- PENN could be University of Pennsylvania
- STAN is likely Stanford University
- WIC might be WiCell, a known stem cell bank

The "i" is probably part of "iPS", meaning induced pluripotent stem cell. So, UPSFRi005-A-1 would be an iPS cell line from UPSFR.

Confirming the pattern: All entities have "i" embedded, which aligns with "iPS". The suffixes like -A-1 or -25-12 are specific identifiers for sub-lines or passages.

Therefore, the common concept isn't just "cell line" but specifically "induced pluripotent stem cell line" because:

- They all contain "i" in a position typical for iPS notation.
- The prefixes match known institutional codes in stem cell research.
- Generalizing to just "cell line" would be too broad, as there are many types (e.g., embryonic, cancer), but the "i" points to induced pluripotent.

.....

Thus, the most specific hypernym is "induced pluripotent stem cell line".

In this example, the target LLM does not know these four iPS cell line items at all but it discovers the same naming pattern and infers that "i" means "iPS" and successfully induces the concept "induced pluripotent stem cell line".

Example 2

Model: Qwen3-235b-a22b-thinking-2507
Items: Adobe Source Libraries, Strawberry, Nuitka, Netdata (all unknown)
Label: software
Answer: software product
Reasoning content:
Okay, the user is asking me to act as an expert in Ontology Induction.
.....
Let me analyze each entity one by one to understand what they are:
- **Adobe Source Libraries**: I know this is a collection of C++ libraries developed by Adobe. It's software-related, specifically open-source code libraries for developers.
- **Strawberry**: This is tricky. It could refer to the fruit, but in a tech context, it might be Strawberry Perl—a programming language distribution. Given the other entities are tech-oriented, I'll assume it's the software version, not the fruit. Confirming: yes, Strawberry Perl is a well-known Perl distribution for Windows.
- **Nuitka**: This is a Python compiler. It translates Python code into C and compiles it, so it's definitely a software tool for programming.
- **Netdata**: This is a real-time performance monitoring tool. It's open-source software used for system and application metrics visualization.
All four are clearly software-related. But I need to dig deeper to find a tighter commonality.
- Adobe Source Libraries: Libraries (code components)
- Strawberry Perl: A distribution of a programming language
- Nuitka: A compiler (tool for code translation)
- Netdata: A monitoring application
They're all types of software, but "software" is too broad. What's more specific? Let me think about categories:
.....

Models		4x0y					0x4y				
		Syn	M.S.	M.G.	Rel	Wrg	Syn	M.S.	M.G.	Rel	Wrg
Llama	LO	25.6	20.2	27.2	21.6	5.4	17.8	18.8	24.2	22.4	16.8
	PA	27.8	17.4	39.4	11.8	3.6	31.6	16.0	31.2	16.4	4.8
Qwen	LO	43.6	8.0	33.0	8.8	6.6	28.8	7.8	37.0	15.6	10.8
	PA	45.6	11.0	34.8	8.6	0	36.4	10.8	43.2	8.8	0.8
Qwen235	LO	39.5	20.0	25.0	13.5	2.0	32.0	11.0	41.0	8.0	8.0
	PA	49.0	12.0	28.5	10.5	0	41.0	11.5	40.0	7.0	0.5
DSv4	LO	49.5	11.5	30.5	8.0	0.5	35.5	13.5	36.5	5.5	9.0
	PA	55.0	10.0	31.0	3.5	0.5	43.5	7.5	41.5	6.0	1.5

Table 3: Full results of rating distributions for 4x0y and 0x4y (%).

Models		Syn	M.S.	M.G.	Rel	Wrg
Llama	3x1y	22.0	20.6	30.2	19.6	7.6
	2x2y	22.4	22.6	25.4	19.0	10.6
	1x3y	18.8	22.4	24.6	21.6	12.6
Qwen	3x1y	37.8	11.6	34.8	10.2	5.6
	2x2y	36.6	9.2	37.2	10.4	6.6
	1x3y	32.2	8.8	39.0	12.4	7.6
Qwen235	3x1y	42.5	15.0	34.0	7.0	1.5
	2x2y	37.0	17.0	30.5	10.5	5.0
	1x3y	32.0	10.0	39.0	13.0	6.0
DSv4	3x1y	49.0	10.5	34.0	6.0	0.5
	2x2y	47.0	11.0	32.5	7.5	2.0
	1x3y	42.5	14.0	30.5	10.0	3.0

Table 4: Full results of rating distributions for 3x1y, 2x2y and 1x3y under LO setting (%).

- Adobe Source Libraries: A software system (collection of libraries)
- Strawberry Perl: A software distribution (product)
- Nuitka: A software compiler (product)
- Netdata: A software application (product)
Yes! All are distinct **software products** or **software systems**.
.....
So the most specific common concept is software product.

In this example, the target LLM knows part of each item, especially the type information. However, some details about these items are wrong so that all items are classified as unknown. For example, Wikidata describes Adobe Source Libraries as a collection of GUI libraries, and the target LLM explains it as a collection of C++ libraries. The target LLM knows the broad types but gets some details wrong. The type information is enough for induction at this granularity in this example.

G Additional Analyses for Section 5.3

This appendix provides detailed statistics for the judge-reliability and property-quantity analyses reported in Section 5.3.

Category	Precision	Recall	F1
Syn	0.885	0.931	0.908
M.S.	0.677	0.808	0.737
M.G.	0.885	0.818	0.850
Rel	0.645	0.625	0.635
Wrg	0.750	0.667	0.706
Macro Avg.	0.769	0.770	0.767

Table 5: Agreement between GPT-5.2 ratings and manual labels on 200 sampled predictions.

G.1 Judge Reliability

Human audit. We manually audit 200 predictions sampled across target models, methods, difficulty levels, and rating categories, and treat the manual labels as references. Table 5 reports the category-level agreement with GPT-5.2. Besides, Exact agreement is 80.5% (161/200), Cohen’s $\kappa = 0.743$, and weighted F1 = 0.805.

Cross-judge validation. We re-evaluate all PA and PPACI predictions under the 0x4y setting using DSv4 with the same evaluation rubric, yielding 2,800 paired judgments. Table 6 lists the scores with percentage details and Table 7 reports agreement statistics between GPT-5.2 and DSv4.

G.2 Property Quantity Analysis

We analyze property quantity using three average-property-count intervals, [0, 3), [3, 7), and [7, +), from concept-level and query-group-level perspectives (Table 8).

Concept-level analysis. For each target concept, we compute the average number of retained properties of the member items used to construct its extensional prototype representation.

Query-group-level analysis. For each 0x4y query group, we compute the average number of retained properties across its four query items.

Target Model	Judge	PA						PPACI					
		Score	Syn	M.S.	M.G.	Rel	Wrg	Score	Syn	M.S.	M.G.	Rel	Wrg
Llama	GPT-5.2	0.6328	31.6	16.0	31.2	16.4	4.8	0.6808	49.4	12.2	10.8	17.6	10.0
	DSv4	0.6450	32.2	16.0	33.4	14.0	4.4	0.6868	48.6	13.8	10.4	19.2	8.0
Qwen	GPT-5.2	0.6840	36.4	10.8	43.2	8.8	0.8	0.6964	52.6	7.6	14.8	17.8	7.2
	DSv4	0.6848	36.0	10.8	44.4	8.2	0.6	0.7044	51.2	10.2	15.6	16.4	6.6
Qwen235	GPT-5.2	0.7160	41.0	11.5	40.0	7.0	0.5	0.7080	49.0	10.0	23.0	11.5	6.5
	DSv4	0.7120	41.5	11.5	37.0	10.0	0.0	0.7300	50.0	9.5	26.0	12.0	2.5
DSv4	GPT-5.2	0.7145	43.5	7.5	41.5	6.0	1.5	0.7535	55.5	8.0	22.5	11.0	3.0
	DSv4	0.7035	43.0	7.0	39.5	10.0	0.5	0.7570	54.0	7.5	28.0	8.5	2.0

Table 6: Detailed PA and PPACI results under GPT-5.2 and DSv4 evaluation on the 0x4y groups. Rating proportions are reported in percentages (%).

Setting	Scope	N	Exact (%)	Cohen’s κ	AC1	Weighted κ	Pearson r	MAE	Bias
All models	Overall	2800	88.86	0.8413	0.8649	0.9091	0.9095	0.0416	+0.0056
	PA	1400	88.14	0.8288	0.8566	0.8806	0.8807	0.0445	+0.0025
	PPACI	1400	89.57	0.8453	0.8746	0.9283	0.9292	0.0386	+0.0086
Excl. DSv4 outputs	Overall	2400	88.67	0.8406	0.8622	0.9093	0.9098	0.0424	+0.0071
	PA	1200	88.00	0.8286	0.8545	0.8806	0.8808	0.0451	+0.0048
	PPACI	1200	89.33	0.8432	0.8715	0.9281	0.9289	0.0397	+0.0095

Table 7: Agreement between GPT-5.2 and DSv4 as evaluators. N denotes the number of paired judgments; Exact is the percentage of predictions receiving identical five-way ratings from both judges; Cohen’s κ measures chance-corrected categorical agreement; AC1 denotes Gwet’s AC1, which is more robust to imbalanced category distributions; Weighted κ measures agreement while accounting for the score distances among the five rating categories (1.0/0.8/0.5/0.2/0.0); Pearson r is the correlation between the corresponding numerical scores; MAE is the mean absolute score difference; and Bias is the mean score difference (DSv4 minus GPT-5.2), with positive values indicating higher scores from DSv4. "Excl. DSv4 outputs" excludes predictions generated by DSv4 itself.

Method	Avg. Props.	Concept-level						Query-group-level					
		N	Syn	M.S.	M.G.	Rel	Wrg	N	Syn	M.S.	M.G.	Rel	Wrg
PA	[0, 3)				/			680	38.2	14.0	32.9	13.7	1.2
	[3, 7)				/			492	33.3	12.2	45.9	6.7	1.8
	[7, +)				/			213	39.9	8.0	39.9	12.2	0.0
PPACI	[0, 3)	660	50.8	11.4	12.1	12.6	13.2	688	49.1	10.9	12.5	14.5	12.9
	[3, 7)	546	48.9	9.5	20.7	17.9	2.9	498	52.8	8.4	21.1	14.7	3.0
	[7, +)	194	60.3	4.1	13.4	21.1	1.0	214	55.1	8.4	13.1	22.9	0.5

Table 8: Property-quantity analysis at the concept and query-group levels. For the concept-level analysis, Avg. Props. denotes the average number of retained properties of member items used to construct the target concept’s extensional representation; for the query-group-level analysis, it denotes the average across the four query items. Rating values are percentages. 15 Error cases from Llama under PA are excluded from the statistics. "/" indicates not applicable.

Concept	Domain	ID in Wikidata	# items
Most transparent. Items whose labels explicitly contain conceptual information, where the concept is directly embedded in the name.			
theory	science	Q17737	559
paradox	science	Q483372	146
prisoner-of-war camp	history; geography	Q1070290	266
zhou	history; geography	Q1070912	395
medical test	medicine	Q2671652	206
adviser	society	Q2994387	51
fictional dimension	fiction	Q100706037	89
type of crime	society	Q130583773	108
Moderately transparent. Items follow semantic naming conventions that provide conceptual hints, where the label structure suggests the concept.			
stomach disease	medicine	Q175827	30
technique	technology	Q2695280	1108
political movement	society	Q2738074	701
golf tournament	sports	Q2990946	1700
biological process	biology	Q2996394	2000
integral graph	science	Q3036983	34
intestinal disease	medicine	Q3055380	77
literary movement	arts	Q3326717	213
class of chemical substances by use	chemistry	Q100434640	78
type of musical work/composition	arts	Q107487333	269
video game weapon	game	Q124980912	164
intersex variation	medicine	Q130487634	39
Pattern-based. Items adhere to systematic naming patterns, where labels follow domain-specific naming rules that may not be immediately obvious.			
dinosaur	biology	Q430	24
enzyme	biology	Q8047	250
utility helicopter	technology	Q2735392	79
cell line from embryonic stem cells	medicine	Q107102664	2000
induced pluripotent stem cell line	medicine	Q107103143	2000
class of protein-containing complexes	biology	Q107509287	2000
gun-brig	military	Q130396697	298
armed merchant cruiser	military	Q131424998	70
Most challenging. Items with opaque labels including common nouns, neologisms, abstract terms, or proper names that offer minimal conceptual cues.			
song	arts	Q7366	2000
software	technology	Q7397	2000
video game	game	Q7889	2000
ghost town	geography	Q74047	2000
album	arts	Q482994	2000
unfinished creative work	arts	Q2910675	152
anthropomorphic spider	fiction	Q100592518	12
mini album	arts	Q107154516	121
type of dance	arts	Q107357104	2000

Table 9: Four difficulty categories for the 37 chosen concepts.