GPTKB: Comprehensively Materializing Factual LLM Knowledge

Anonymous ACL submission

Abstract

LLMs have majorly advanced NLP and AI, and next to their ability to perform a wide range of procedural tasks, a major success factor is their internalized factual knowledge. Since (Petroni et al., 2019), analyzing this knowledge has gained attention. However, most approaches investigate one question at a time via modest-sized pre-defined samples, introducing an *"availability bias"* (Tversky and Kahneman, 1973) that prevents the discovery of knowledge (or beliefs) of LLMs beyond the experimenter's predisposition.

To address this challenge, we propose a novel methodology to comprehensively materialize an LLM's factual knowledge through recursive querying and result consolidation.

As a prototype, we employ GPT-4o-mini to construct GPTKB, a large-scale knowledge base (KB) comprising 101 million triples for over 2.9 million entities - achieved at 1% of the cost of previous KB projects. This work marks a milestone in two areas: (1) For LLM research, for the first time, it provides *constructive* insights into the scope and structure of LLMs' knowledge (or beliefs). For KB construction, it pioneers *new pathways* for the long-standing challenge of general-domain KB construction. GPTKB is accessible at `anonymized`.

1 Introduction

LLMs have been one of the most exciting recent developments in NLP and AI, and next to their ability to perform a wide set of procedural tasks, a major success factor is the factual knowledge that they have internalized (Bubeck et al., 2023). Their potential to store factual knowledge, like *(Nelson Mandela, award, Nobel Peace Prize)*, was first highlighted by Petroni et al. (2019), and this has generated an own (sub-)field of studying all aspects of factual knowledge in LLMs, from trying to locate it, to estimates of their storage potential, to techniques to effectively elicit it (Jiang et al., 2020; Roberts

Existing LLM analyses	Our proposal
✗ *Prompt-and-discard:* - Transient single-focus explorations (e.g., factuality OR bias OR cutoff)	✓ *Persistent resource:* - Materialized KB reusable for a wide array of questions
✗ *Sample availability bias:* - Insights bound to experimenter's predisposition	✓ *Recursive materialization:* - Discover unconceived LLM knowledge/beliefs
✗ *Scratching the surface:* - Few 100s-100K samples - Breadth and depth of LLM knowledge untouched	✓ *Massive-scale:* - Over 100M records - Recursive crawl to unprecedented breadth and depth

Table 1: Comparison of existing LLM knowledge analysis approaches and our proposal.

et al., 2020; Veseli et al., 2023; Sun et al., 2024; Wu et al., 2024). A large set of benchmarks and studies transiently investigate the knowledge storage ability via example-based prompting, e.g., to determine, how many triples from common knowledge bases (KBs) or question answering (QA) datasets are known to LLMs. However, all these works are subject to an **availability bias** (Tversky and Kahneman, 1973), i.e., they only surface LLM knowledge/beliefs[1] on topics that were known to and pre-designed by the experimenter. They inherently cannot discover knowledge on topics outside of their benchmarks.

To enable comprehensive insights into LLM knowledge, we propose to extensively materialize the knowledge of LLMs into a KB (see Table 1). General-world KBs like Wikidata (Vrandecic and Krötzsch, 2014), Yago (Suchanek et al., 2007) and DBpedia (Auer et al., 2007) are important and long-standing backbones for AI applications, yet have seen little innovation in the last years, and have only been constructed manually, or by scraping semistructured web content (Weikum et al., 2021). In particular, we propose to use recursive graph exploration to obtain comprehensive (named-entity-

---

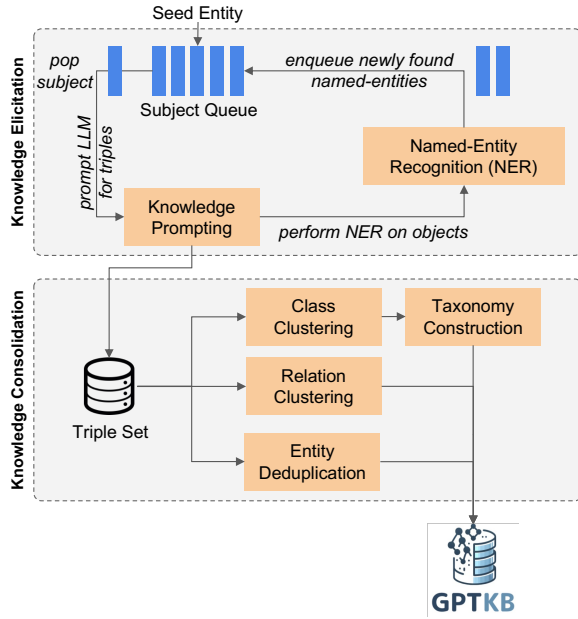[1]The terminology here is contentious, see Section 6.1.

Figure 1: Overview of our approach for factual knowledge materialization from LLMs.

centric) LLM knowledge, and to consolidate it via LLM-based entity disambiguation, class and relation canonicalization, and taxonomy induction (see Fig. 1). This proposal faces several challenges:

1. *Runtime and cost:* State-of-the-art KBs contain millions of entries, and LLM inference is relatively slow. It is therefore unclear how to perform comprehensive knowledge elicitation under practical monetary and time constraints.

2. *Variance, hallucinations, and scoping:* Latent knowledge in LLMs covers a wide set of topics, varies by entity, and includes hallucinations. We need a method that elicits a high quantity, without encouraging hallucinations, and without falling into bottomless corners, e.g., open-ended phrases or translations.

3. *Global inconsistency*: Distinctive prompts risks surfacing outputs that are not globally consistent, e.g., introducing duplicate relations, entities, or disconnected class hierarchies.

Our approach builds on the following ideas: To overcome scaling issues, and obtain relevant knowledge, we utilize a commercial API that allows to massively batch requests, and utilize named entity recognition (NER) and carefully crafted prompts to restrict the explored space, along with prompts that encourage varied answer sizes. To obtain a

coherent KB, we perform a set of canonicalization and disambiguation steps, entirely relying on the LLM itself. In summary, our salient **contributions** are:

1. To the best of our knowledge, we are the first to propose extensive materialization for analyzing factual LLM knowledge.

2. We present a scalable methodology to elicit massive amounts of LLM knowledge, and to consolidate it.

3. Using GPT-4o-mini, we construct GPTKB, the first large-scale KB entirely built from an LLM's parametric knowledge, containing 101M assertions for over 2.9M entities.

4. We use GPTKB to exemplarily analyze GPT-4o-mini's factual knowledge in terms of scale, accuracy, bias, and cutoff, at the same time.

Our work is a significant advancement for two areas: For LLM research, we introduce a proof-of-concept methodology that, for the first time, enables *constructive* insights into the knowledge (or beliefs) of LLMs. For the Semantic Web, we provide fresh momentum for the long stale problem of open-domain KB construction. We provide code and a concrete resource, GPTKB, both as a 3.8 GB download, and via an online browsing interface and SPARQL query interface at `anonymized`.

2 Related work

Extent of LLM knowledge Since the emergence of LLMs, the question of how much these models know has frequently been raised (Petroni et al., 2019; Roberts et al., 2020; Jiang et al., 2020; Veseli et al., 2023; Sun et al., 2024; Wu et al., 2024). So far, the widely adopted approach is to sample a domain of interest, e.g., question-answer pairs from a common benchmark, or triples from Wikidata, transiently probe the LLM, and compute the fraction that the LLM can correctly answer/complete. This is prone to "availability bias" (Tversky and Kahneman, 1973) since we do not get a comprehensive view of the LLM knowledge outside the focus of existing benchmarks. For instance, we found that GPT-4o-mini holds substantial knowledge about art periods or hobbies, which are not covered in existing KBs. Kassner et al. (2021) propose a small-scale persistent memory component for ensuring LLM answers remain consistent

KB	#entities	#assertions
Wikimedia-related		
Wikidata	113M	1.62B
Wikidata5m	5M	20M
Yago 4.5	50M	140M
DBpedia	3.8M	75M
Text-extracted		
NELL	?	12M
ReVerb	?	15M
Generative		
GPTKB	2.9M	101M

Table 2: Size comparison of major KBs. See Appendix F for sources.

Figure 2: Graph exploration from the seed entity *Vannevar Bush*.

over multiple prompts. Nguyen et al. (2024) iteratively prompt GPT-3.5 for obtaining 167K sentences containing cultural commonsense for 11K subjects. Cohen et al. (2023) propose to iteratively prompt GPT-3 for relations and relational assertions for triples. In difference to (Nguyen et al., 2024), we are after structured content (triples), at a much larger scale. In difference to (Cohen et al., 2023), we unveil the practical challenges that such a graph exploration faces, tackle them, and perform LLM-based KB construction at scale.

Iterative information extraction Iterative information extraction is a long-standing idea, already prototyped by Google cofounder Sergey Brin via the DIPRE system (Brin, 1998). The Snowball system by Agichtein and Gravano (2000) presented a large-scale implementation of this idea, harvesting tuples from over 300,000 newspaper documents. The KnowItAll system (Etzioni et al., 2004) followed a similar approach, and led to the ReVerb KB (Fader et al., 2011). Despite a proposal by (Cohen et al., 2023), to date, we are not aware of any attempts to extract knowledge from LLMs at scale.

Large-scale KB construction Dominating public large-scale KBs are Wikidata (Vrandecic and Krötzsch, 2014), Yago (Suchanek et al., 2007) and DBpedia (Auer et al., 2007), all started more than 10 years ago. While Wikidata is constructed by volunteers, Yago and DBpedia represent the paradigm of (semi-)structured information harvesting and integration, extracting in particular from Wikipedia infoboxes and Wikidata (Weikum et al., 2021). They all remain incomplete (Razniewski et al., 2024), warranting the search for novel paradigms. Commercial projects like the Google KG (Singhal, 2012) or Amazon's KG (Dong et al., 2020) have
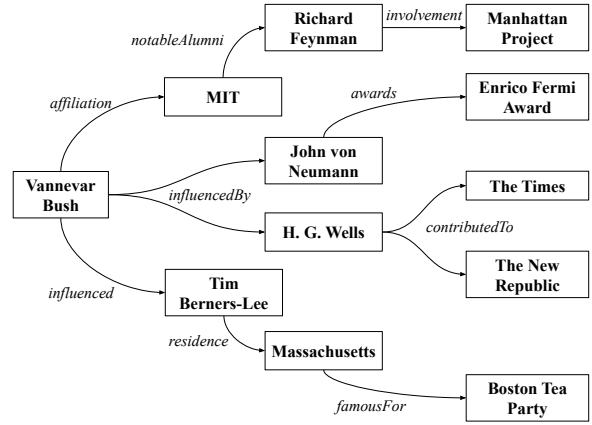
usually followed these approaches. By comparison, text-based KB construction, e.g., in NELL (Mitchell et al., 2018) or ReVerb (Fader et al., 2011), has achieved less adoption. Our approach is more related to the latter approaches, as LLMs are distillations of large text corpora. Table 2 gives an overview of major KB projects.

3 Methodology

An overview of our approach is shown in Figure 1. In the first phase, **knowledge elicitation**, we iteratively elicit LLM triples for a given subject, and enqueue newly encountered named entities for further triple elicitation. In the second phase, **knowledge consolidation**, we consolidate the resulting triple set by canonicalizing entities, relations and classes, and by constructing a coherent taxonomy. Our paradigm refrains from imposing any standardized vocabularies, or using existing KBs as disambiguation references to effectively materialize the parameterized LLM knowledge.

3.1 Knowledge elicitation

Iterative graph expansion The process starts from one or a set of seed subjects, e.g., *Vannevar Bush*, the visionary behind the concept of hyperlink-based knowledge organization (Bush, 1945). From the objects in the triples obtained for him, we can then identify further entities (e.g., *MIT* (affiliation) or *Everett, MA* (birth place)), for which we can then elicit further knowledge, and so on (Brin, 1998). Figure 2 illustrates how in 3 hops, we arrive at entities of diverse types, such as *historical event* (Boston Tea Party, Manhattan Project), *newspaper* (The Times), and *magazine* (The New Republic).

3

Knowledge elicitation A major challenge is to elicit as much knowledge as possible, but at the same time, not encourage hallucinations. We found that without guidelines on the expected number of triples, LLMs showed too little variance in the number of triples per subject, while we would expect them to return many more triples for *Einstein* than for other entities. We solve this via indications that are defined in relation to the entity's popularity. In difference to the proposal (Cohen et al., 2023), we also drop the separate relation elicitation, and relation-specific knowledge extraction, as these do not scale. To structure the knowledge, we also request that at least one *instanceOf* triple should be returned. Output parsing is eased by using the structured outputs feature of OpenAI's API. In turn, this also reduces hallucinations, e.g., additional qualifiers, textual descriptions, or similar. The full prompt is in Figure 8 in the Appendix.

Named-entity recognition (NER) Our early attempts at graph exploration were plagued by topical runaway into linguistic knowledge, translations, etc., because LLM-generated objects cover a wide range of imprecisely delineated data types. We experimented with various ways to filter non-named entities from the expansion, but found that existing NER frameworks struggled with the context-free entity labels available from our crawl process (most NER models are trained on sentences). In the end, we used the LLM itself to identify named entities, processing multiple candidates at once. The full prompt is presented in Figure 9 in the Appendix.

3.2 Knowledge consolidation

The knowledge elicitation phase returns a huge degree of redundancy and variance. Feeding a large amount of existing entities or taxonomy into the knowledge elicitation prompt could mitigate this problem. However, that would also increase costs significantly, and even be infeasible once we have generated millions of entities and triples. Instead, we introduce several steps to consolidate LLMs output post-hoc.

Relation clustering The elicitation phase generates 788K distinct relation names, with many obvious duplicates, e.g., *instanceOf*, *isA*, or *InstanceOf*. We apply a greedy clustering algorithm to this set (see Algorithm 1). This process merges relation r into a more frequent relation s, selected as the one with the highest textual embedding similarity to r among all relations more frequent than r, if

the similarity is greater than an adaptive threshold (defined in line 6 in Algorithm 1). This threshold varies with the frequency of the relation, leading to a more aggressive removal of relations with low frequency.

Algorithm 1 Relation clustering

Input: A set of relations $\mathcal{R}$, hyperparameter α
Output: A map from relation to cluster-id $\mathcal{C}$
1: $\mathcal{R} \leftarrow \text{sort}(\mathcal{R}, \text{key} = \text{frequency}, \text{reverse} = \text{True})$
2: $\mathcal{C} \leftarrow \{\}$
3: $next\_cluster\_id \leftarrow 0$
4: **for** $r \in \mathcal{R}$ **do**
5:     $s' = \text{argmax}([\text{similarity}(r, s) \textbf{ for } s \in \mathcal{R} \textbf{ upto } r])$
6:     $threshold \leftarrow \alpha \times \frac{\log(\text{frequency}(r))}{\log(\text{frequency}(\text{first}(\mathcal{R})))}$
7:     **if** $\text{similarity}(r, s') > threshold$ **then**
8:         $\mathcal{C}[r] \leftarrow \mathcal{C}[s']$
9:     **else**
10:        $\mathcal{C}[r] \leftarrow next\_cluster\_id$
11:        $next\_cluster\_id \leftarrow next\_cluster\_id + 1$
12:    **end if**
13: **end for**
14: **return** $\mathcal{C}$

Class clustering Similarly to relations, there are 103K distinct class names (i.e., objects for *instanceOf* relations) generated by the knowledge elicitation phase, with both obvious duplicates, and many overly specific cases. Algorithm 1 is also applied to clean this set.

In our experiment, both relation and class clustering uses SentenceTransformers (Reimers and Gurevych, 2019) for embedding cosine similarity computation, with $\alpha = 1.2$ chosen via manual inspection of small held-out sets.

Taxonomy construction The classes in the KB so far do not form a coherent, or even connected taxonomy, as the knowledge elicitation process only expands named entities. We propose Algorithm 2, which is based on the LLM itself, to build a complete taxonomy from these individual classes.

The construction starts by generating a high-level taxonomy via the LLM. Then, for each of the existing classes in the KB (sorted by generality scores given by the LLM), we find the lowest node of which the given class is identified as a subclass, via depth-first search (see function IN-SERT_CLASS_RECURSIVE in Algorithm 2). Then, we ask the LLM to update the sub-taxonomy starting from the found node with the given class. In this step, the LLM may generate new intermediate classes on the path from the found node to the given class.

The prompts used for seed taxonomy generation, generality scoring, superclass identification, and

Algorithm 2 Taxonomy construction

Input: Knowledge base (KB)
Output: Taxonomy
 1: **function** CONSTRUCT_TAXONOMY(KB)
 2:     *taxonomy* ← LLM_get_high_level_taxonomy()
 3:     **for** *class* ∈ KB.classes **do**
 4:         *class*.generality_score ← LLM_get_score(*class*)
 5:     **end for**
 6:     *classes* ← sort(KB.classes, key=generality_score)
 7:     **for** *class* ∈ classes **do**
 8:         insert_class_recursive(*taxonomy*.root, *class*)
 9:     **end for**
10:     **return** *taxonomy*
11: **end function**

12: **function** INSERT_CLASS_RECURSIVE(*node*, *class*)
13:     **if** has_children(*node*) **then**
14:         *next* ← LLM_superclass(*class*, *node*.children)
15:         **if** *next* **is not** NULL **then**
16:             insert_class_recursive(*next*, *class*)
17:         **else**
18:             LLM_update_taxonomy(*node*, *class*)
19:         **end if**
20:     **else**
21:         LLM_update_taxonomy(*node*, *class*)
22:     **end if**
23: **end function**

Entities	2.9M
Triples	101M
Relations	1,804 (788K before canonicalization)
Classes	473 (103K before canonicalization)
Triple objects	37M entities, 64M literals
Avg. triples/entity	36
Avg. label length	24.5 characters
Avg. outlinks	4
Subject-precision*	74% Verifiable, 9% Plausible, 17% Unverifiable
Subjects in Wikidata*	37% in WD, 63% not in WD
Triple-precision*	31% True, 61% Plausible, 1% Implausible, 7% False

* as a weighted average across layers.

Table 3: Statistics of GPTKB.

off between performance and cost. The batch feature is particularly important, enabling us to send, after startup, up to 100 batches of 10,000 entities in parallel. The model's size is not publicly released, but has been estimated to be around 8B (Howarth, 2024; Zeff, 2024).

Seed entity, result size, runtime and cost We start the process from a single entity, *Vannevar Bush*, in honor of his vision of interlinked knowledge (Bush, 1945). Note that this choice is arbitrary and overall inconsequential. As general-world knowledge graphs are densely connected (Hogan et al., 2022), any seed entity connected to popular entities serves as well. For instance, we reach *MIT* in one hop, *Alan Turing* in 2 hops, *Kyoto* in 3 hops.

We require a total of 2,200 batches to prompt the LLM up to BFS-depth 10, which corresponded to 5.8M prompted entities, of which for 2.9M a non-empty answer was returned. The whole process took 27 hours, and including trial runs, constructing GPTKB cost us a total of $3,500 for API calls.

GPTKB statistics Our KB contains a total of 101M triples for 2.9M entities, organized into 1,804 relations and 473 classes. This gives an average of 36 triples per subject, with two distinct clusters, in particular, 651K entities with 10 triples and 86K with 50 triples, with most others near these values. Of all triples, more than 37M have an entity as object, and more than 67M have a literal as an object. The average length of entity labels is 24.5 characters. Example output for *Vannevar Bush* is shown in Figure 2. Statistics are shown in Table 3.

Dataset provision and license We provide our KB both as a download (3.8 GB in TTL format), via a web browsing interface, and via a SPARQL query endpoint at `anonymized`. We provide our KB under the noncommercial CC BY-NC 4.0 license, not out

taxonomy update are provided in Figure 10 in Appendix C. Further LLM-based refinement in the style of (Peng et al., 2024) could be considered.

Entity deduplication Naive graph exploration frequently yields duplicates, such as *John F. Kennedy* and *John Fitzgerald Kennedy*. To remove these duplicates without requiring extraordinary runtime, we follow the standard blocking-based deduplication approach by Köpcke and Rahm (2010). This approach is based on choosing a blocking key, which is an attribute by whose values we obtain meaningful partitions on entities.

We focus on entities that are instances of the most interesting class, *humans*. We choose to block entities by birth dates. Within each block, we consider an entity pair as duplicates if (i) their labels have close meanings (i.e., cosine similarity between SentenceTransformers embeddings are greater than 0.85), and (ii) 30% of their triples are exactly the same. More advanced methods could utilize LLMs themselves for deciding on entity equivalence (Ding et al., 2024), but it would incur more cost and longer runtime.

4 Implementation

LLM choice and parallelization We chose *GPT-4o-mini* (OpenAI, 2024) for our experiments, because of (i) its ability to process requests in batches, (ii) its structured output feature, (iii) its good trade-

of commercial self-interest, but because we want to avoid potential misuses (see Section 8). This license is compatible with the permissive terms of use of OpenAI (OpenAI, 2022).

5 Analysis

5.1 Precision

Evaluating the precision of large-scale KBs is not straightforward, because they contain a significant amount of long-tail knowledge, for which finding evidence or counter-evidence is difficult. In line with (Suchanek et al., 2007), we evaluate precision within the context of web-retrievable information. Figure 3 summarizes our findings.

For verifying **entities**, we retrieved 5 search results from a web search API for the entity label, which are then used to decide whether the entity's existence could be verified, appears plausible, or could not be verified. Deciding on these labels based on textual context represents a task of textual entailment, a.k.a. natural language inference (NLI), a task generally considered solved for LLMs.[2] We therefore use another LLM, Llama-3.1-70B-Instruct (Dubey et al., 2024), for this task. The LLM hereby only judges textual entailment, it does not independently judge truth from its parameters. Based on 1,000 samples, average entity verifiability is 74%, and as shown in Figure 3, we observe a continuous drop of verifiability over the layers, stabilizing at layer 10 at about 70%. The plot also includes Wikidata-existence, which paints a similar trend at lower levels. We generally observe a wide variance across classes, finding, for instance, that persons consistently have a higher verifiability than fictional character, where many codes were made up (e.g., *Officer K.I.T.T. XV* and *Officer K.I.T.T. XVI*) or varying incorrect type numbers are added to real names continuously by the LLM (e.g., *Agent 71* and *Agent 72*).

For a given **triple**, we retrieve the top-5 webpage snippets for the search term *<subject> <object>* from a search engine API, then again perform LLM-based textual entailment to decide whether the given triple is entailed (entailed/plausible/implausible/false), given the snippets. We specifically adapt the prompt used by

---

[2]On LLM performance on textual entailment, the popular SNLI (Bowman et al., 2015) is comparable, which was considered solved in 2020 by BERT architectures (Zhang et al., 2020) at 92% accuracy, which is the ceiling, given dataset noise. This is also why the literature mentions no numbers for newer models like Llama on this task anymore.
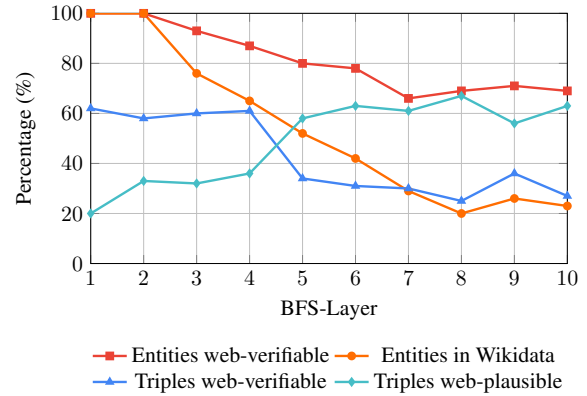

Figure 3: Accuracy of GPTKB across layers.

Adam and Kliegr (2024), extending it with the "false" level. Based on 1,000 samples, we find that on average, 31% of triples are entailed, 61% are plausible, 1% is implausible, and 7% are false.

In terms of the accuracy of the **taxonomy**, we follow the per-edge evaluation scheme proposed by Bordea et al. (2016), finding that 64% of all subclass-superclass edges are considered correct, using the Llama model as judge. We also evaluate whether a superclass is most appropriate, by offering all siblings as alternatives. Here, for 70% of all subclass-superclass edges, the superclass is considered the best alternative. Note that the structure of our taxonomy still leaves room for improvement in terms of long-range dependencies and distributions, that are not easy to quantify or address locally.

5.2 GPTKB content and comparison

Classes The most frequent classes generated for entities are shown in Figure 4, where we find that *Person* dominates the dataset. Several more classes are actually merged into *Person*, e.g., *human, hominid*. Including these, the total number of *Person* is more than 287K. Further details by layer are in Appendix B.

Properties The most frequent properties in the whole KB are *instanceOf* (3.1M) and *features* (2M) with similarly generic properties following at the entity level. At the class level, we find more specific properties, for instance, the most frequent properties in the class *Person* are *instanceOf* (309K), *occupation* (198K), *knownFor* (179K), *exhibition* (178K).

Content bias We also analyze the geographic bias of our KB, using the listed nationalities as proxy. We observe a clear bias towards English-language nationalities: the top-2 are American
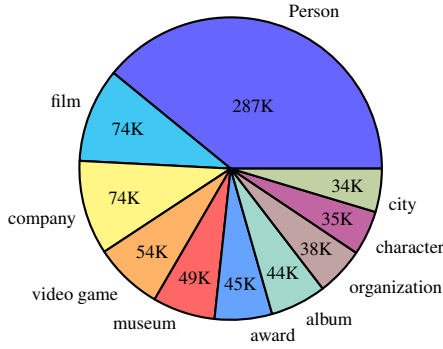
Figure 4: The 10 most frequent classes in GPTKB, which make up 25% of all entities.

(119K) and British (35K), while the next 3 are French (18K), German (14K), and Japanese (11K). This bias in GPTKB is stronger than, e.g., in Wikidata (Shaik et al., 2021), and a curious lack of others (e.g., Chinese only 3K, compared to 8K for Russian, or 7K for Indian/Spanish), likely reflecting the (undisclosed) English-centric training corpus of GPT-4o-mini.

An interesting point is also gender bias: The *gender* property of GPTKB contains 15K female versus 8K male values. For first names, male ones are still more than female (47% vs. 37% based on the *gender-guesser 0.4.0* Python library), but this is still a much lower bias than in other KBs, likely reflecting the effects of gender debiasing.

Wikidata overlap and novelty   We compare with Wikidata on several aspects: **First**, we compute the fraction of subjects that exist in Wikidata. From a random sample of 2,000 subjects from GPTKB, 24% have an entity with exactly matching label in Wikidata. A further 6.5% have a non-empty search result, i.e., an entity of paraphrased or similar label. The remaining 69.5% appear novel. **Second**, we exemplarily look at the 41 triples for *Vannevar Bush*, of which we find that more than 10 are not contained in Wikidata, e.g., his affiliation with the US government, his children count (incorrect by 1), or him inventing the concept of hypertext.[3] **Third**, we identify several properties not modelled at all in Wikidata, for instance, *historicalSignificance* (396K triples), *artStyle* (174K triples), or *hobbies* (44K triples). This indicates that GPTKB potentially contains a significant amount of novel knowledge. A more comprehensive KB comparison in the style of (Färber et al., 2018) is planned.

---

[3]An exact comparison is not straightforward, because relation names do not perfectly align, and objects in GPTKB are substantially more wordy.

6   Discussion

6.1   Lessons for LLM epistemology

Terminological observations   The notion and definition of LLM knowledge itself is controversial (Fierro et al., 2024). We adopt here the term *knowledge base* because it is common in the field, but would consider the term *belief base* equally appropriate. In terms of the definitions of knowledge that are introduced by Fierro et al. (2024), we find that our output falls into the minimalistic sui-generis (g-knowledge) category, as our KB contains false as well as inconsistent triples. Beyond the reported errors, e.g., in many cases, spouse relations are not symmetrically represented.

Storage capacity   Scaling laws and storage capacity are intensively investigated (Allen-Zhu and Li, 2024), but usually on synthetic data. Our experiments provide a lower bound on real data: As discussed in Section 4, GPT-4o-mini likely has in the order of 8B parameters. Given the 101M triples that we obtained, that makes *79 parameters per triple*.[4] We emphasize that this number captures only encyclopedic knowledge, yet that LLMs also possess other knowledge, e.g., linguistic knowledge, procedural knowledge.

Complementarity   We observe a high complementarity to existing resources, with 63% of generated entities not being present in Wikidata. Particularly, from Figure 3, GPTKB's entities remain web-verifiable even in deeper layers, about 70% of the entities are web-verifiable in layer 10, even though the percentage of entities found in Wikidata drops to a little over 20%. Apart from named-entities, some particularly noteworthy complementary slices of LLM knowledge concern digital media artifacts, art periods, and people's hobbies.

Accuracy   While on the entity level, most entities appear to truly exist (74% verifiable, 9% plausible), on the triple level, results are lower (31% verifiable, 61% plausible). This indicates that the LLM generally has a grasp of entities, but has more difficulties in correctly modeling their relationships. BFS-depth and quality are negatively correlated, but there is no complete quality collapse within the first 10 levels.

---

[4]From (Allen-Zhu and Li, 2024), a theoretical lower bound of 4 parameters/triple can be deduced, based on Remark 4.4, and their synthetic dataset containing 6 triples per entity. However, the generalizability of results on that synthetic dataset is unclear.

Enabled analyses Our materialized resource enables a range of analyses about factual LLM knowledge, which for space reasons we only skim here: **1) Accuracy**: See Sec. 5.1 and paragraph above. **2) LLM bias**: See Sec. 5.2. **3) Timeliness**: Exemplarily, we collect the most frequently mentioned years, and observe that there is sharp drop after 2023, which matches the knowledge cutoff of the LLM (see plot in Appendix D). **4) Subject-wise consistency:** Our KB still contains many duplicates, e.g., *The Elbe River, River Elbe, Elbe River, river Elbe, Elbe*. Studying their triples gives insights into consistency, e.g., we observe a significant semantic overlap, but also frequent different wordings (e.g., *wildlife - various fish species / fish species*), and minor factual deviations (e.g., length 3x 1094km, 2x 1091km). **5) Structural consistency:** An interesting observation concerns the difficulty that LLMs have with inverse relations (Berglund et al., 2024). We can observe this too, for instance, out of 318K *spouse* triples, only 8K are symmetric, and out of 61K *parentCompany* triples, only 6K are mirrored in *subsidiary* triples.

6.2 Lessons for KB construction

The GPTKB prototype reveals several important lessons for LLM-based KB construction. In particular, we find it notable that building such a large KB was possible so quickly, with a relatively small model.

Precision-recall trade-off The biggest challenge in our view is precision, both in terms of hallucinated entities, and triples. We do not expect that larger models alone will solve this problem, because the long tail, where hallucinations occur, would likely just be pushed farther, but remain difficult to delineate. Tuning the precision-recall trade-off, for example, via more conservative prompts, or via thresholding based on elicited confidence values (Xiong et al., 2024), might be a way forward.

Emerging entities Some applications are especially interested in newly emerging entities, and these are a long-standing challenge for traditional KBC (Hoffart et al., 2016). Web-scraped KBs like Yago and DBpedia could in principle re-run their scrapers periodically, while text-extracted KBs like Nell and ReVerb would require re-runs on new web crawls. Our approach could proceed analogous, re-running the materialization on a newer version of the utilized LLM. If one were to know which entities are affected by updates, one could also perform retrieval-augmented generation, however, knowing where updates occured essentially requires an oracle, and none of the existing KBs successfully employed selective updating.

Consolidation challenges Our exploration surfaced a potpourri of other challenges, some of them known to KB construction research since years (Weikum et al., 2021), others requiring novel adaptations in the light of LLMs. These concern (1) NER for short labels without context, (2) entity deduplication, (3) entity canonicalization, (4) literal typing and canonicalization, (5) relation clustering, (6) relation organization in terms of subrelations, (7) class clustering, (8) taxonomy construction, and (9) triple verification. While we explored simple techniques for 1, 2, 5, 7, 8, each has room upward, and other tasks were not treated so far.

Cost-effectiveness The GPTKB approach deviates from traditional Wikimedia- and data-integration focused KB construction, and although its precision still needs improvement, it stands out with its potential for cost efficiency. In a back-of-the-envelope calculation, Paulheim (2018) estimated the cost per triple for existing manual KB construction projects at $2-6, and for existing automated KB construction projects at $0.01-0.15. In contrast, in our prototypical execution, the cost is just $0.0001 per correct triple ($3.5K/33M), i.e., >100x less than with previous methods.

6.3 Other LLMs

We performed a parallel exploration using Llama-3.1-70B-Instruct on local HPC hardware, and while accuracy is higher (69% of triples verified true in a test run), this did not scale. We also envisioned a run using the strongest publicly available LLM, GPT-4o (80% triples verified true), however, by its 15x higher API cost, and its estimated 25x larger parameter set, assuming knowledge is roughly proportional to parameters, at about $825K, this is beyond our budget. See Appendix E.

7 Conclusion

Our work is a landmark for two fields: For NLP, for the first time, it provides comprehensive insights into what LLMs know (or believe). For the Semantic Web, it shows novel ways forward for the long stale topic of open-domain KB construction. GPTKB is accessible at `anonymized`.

8

8 Limitations

As already observed by Petroni et al. (2019), prompt formulations heavily influence LLM responses, and hence may bias the resulting KB. GPTKB presents one way of materializing LLM knowledge into structured format, but different prompts could give different output. It is also important to observe that LLMs operate inherently probabilistically, while our output (and most web-scale KBs) are binary. Any process translating from probabilistic to binary implicitly encodes a choice between precision and recall, and we do so by the triple counts given as guidance in the elicitation prompt. In our view, such a biasing is hard to avoid, and also not solved by instruction-free sample-only (few-shot) prompts as proposed by Wu et al. (2024), as also examples bias the responses, both in terms of quantity, topics, and vocabulary.

Our exemplary run presents a proof of concept for materializing LLM knowledge. While this knowledge has slices complementary to existing KB knowledge, so far our experiments on verifiability does not satisfy the precision requirements of most downstream use cases, nor is its precision even accurately established, because it contains many long-tail triples without web sources that enable confirming or refuting their truth. The knowledge in our KB can intrinsically not be sourced back to any specific document and contributor. Also, as we do not prescribe any standardized vocabulary, and only apply lightweight consolidation, our KB does not yet possess the consistency of existing projects. If you consider using a KB in production, for the time being, consider an alternative like Wikidata.

Our materialized KB is based on a closed-source LLM that also does not come with guarantees regarding persistent servicing. In the past, similar models have been discontinued, hence, long-term reproducibility is not guaranteed. An execution based on an open model is planned.

References

Daniel Adam and Tomáš Kliegr. 2024. Traceable llm-based validation of statements in knowledge graphs. *arXiv:2409.07507*.

Eugene Agichtein and Luis Gravano. 2000. *Snowball*: extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM Conference on Digital Libraries*.

Zeyuan Allen-Zhu and Yuanzhi Li. 2024. Physics of language models: Part 3.3, knowledge capacity scaling laws. *arXiv preprint arXiv:2404.05405*.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. Dbpedia: A nucleus for a web of open data. In *ISWC*.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2024. The reversal curse: Llms trained on "a is b" fail to learn "b is a". In *ICLR*.

Georgeta Bordea, Els Lefever, and Paul Buitelaar. 2016. Semeval-2016 task 13: Taxonomy extraction evaluation (texeval-2). In *SemEval@NAACL-HLT*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Sergey Brin. 1998. Extracting patterns and relations from the world wide web. In *WebDB*.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Vannevar Bush. 1945. As we may think. *Atlantic Monthly, July*.

Roi Cohen, Mor Geva, Jonathan Berant, and Amir Globerson. 2023. Crawling the internal knowledge-base of language models. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1856–1869, Dubrovnik, Croatia. Association for Computational Linguistics.

Yifan Ding, Amrit Poudel, Qingkai Zeng, Tim Weninger, Balaji Veeramani, and Sanmitra Bhattacharya. 2024. Entgpt: Linking generative large language models with knowledge bases. *arXiv e-prints*, pages arXiv–2402.

Xin Luna Dong, Xiang He, Andrey Kan, Xian Li, Yan Liang, Jun Ma, Yifan Ethan Xu, Chenwei Zhang, Tong Zhao, Gabriel Blanco Saldana, et al. 2020. Autoknow: Self-driving knowledge collection for products of thousands of types. In *KDD*, pages 2724–2734.

Abhimanyu Dubey, Abhinav Jauhri, and Abhinav Pandey et al. 2024. The Llama 3 herd of models. *CoRR*, abs/2407.21783.

9

Oren Etzioni, Michael J. Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2004. Web-scale information extraction in knowitall: (preliminary results). In *WWW*.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. 2018. Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web*, 9(1):77–129.

Constanza Fierro, Ruchira Dhar, Filippos Stamatiou, Nicolas Garneau, and Anders Søgaard. 2024. Defining knowledge: Bridging epistemology and large language models. *arXiv preprint arXiv:2410.02499*.

Johannes Hoffart, Dragan Milchevski, Gerhard Weikum, Avishek Anand, and Jaspreet Singh. 2016. The knowledge awakens: Keeping knowledge bases fresh with emerging entities. In *WWW*.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. 2022. Knowledge graphs. *ACM Comput. Surv.*, 54(4):71:1–71:37.

Josh Howarth. 2024. Number of parameters in gpt-4 (latest data). Available online: https://explodingtopics.com/blog/gpt-parameters/. Accessed: December 11, 2024.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Nora Kassner, Oyvind Tafjord, Hinrich Schütze, and Peter Clark. 2021. BeliefBank: Adding memory to a pre-trained language model for a systematic notion of belief. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8849–8861, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hanna Köpcke and Erhard Rahm. 2010. Frameworks for entity matching: A comparison. *Data & Knowledge Engineering*, 69(2):197–210.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Thomas Lin, Mausam, and Oren Etzioni. 2012. Entity linking at web scale. In *AKBC-WEKEX*, pages 84–88.

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. 2018. Never-ending learning. *Communications of the ACM*, 61(5):103–115.

Tuan-Phong Nguyen, Simon Razniewski, and Gerhard Weikum. 2024. Cultural commonsense knowledge for intercultural dialogues. In *CIKM*.

OpenAI. 2022. Sharing & publication policy. Available online: https://openai.com/policies/sharing-publication-policy. Accessed: December 11, 2024.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Heiko Paulheim. 2018. How much is a triple? estimating the cost of knowledge graph creation. In *ISWC P&D*.

Yiwen Peng, Thomas Bonald, and Mehwish Alam. 2024. Refining wikidata taxonomy using large language models. In *CIKM*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Simon Razniewski, Hiba Arnaout, Shrestha Ghosh, and Fabian Suchanek. 2024. Completeness, recall, and negation in open-world knowledge bases: A survey. *ACM Computing Surveys*, 56(6):1–42.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Zaina Shaik, Filip Ilievski, and Fred Morstatter. 2021. Analyzing race and citizenship bias in wikidata. In *International conference on mobile Ad Hoc and smart systems (MASS)*, pages 665–666. IEEE.

Amit Singhal. 2012. Introducing the knowledge graph: Things, not strings. Google Official Blog. Retrieved September 25, 2024.

Fabian M. Suchanek, Mehwish Alam, Thomas Bonald, Lihu Chen, Pierre-Henri Paris, and Jules Soria. 2024. YAGO 4.5: A large and clean knowledge base with a rich taxonomy. In *SIGIR*.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *WWW*.

Kai Sun, Yifan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2024. Head-to-tail: How knowledgeable are large language models (LLMs)? A.K.A. will LLMs replace knowledge graphs? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 311–325, Mexico City, Mexico. Association for Computational Linguistics.

Amos Tversky and Daniel Kahneman. 1973. Availability: A heuristic for judging frequency and probability. *Cognitive Psychology*, 5(2):207–232.

Blerta Veseli, Sneha Singhania, Simon Razniewski, and Gerhard Weikum. 2023. Evaluating language models for knowledge base completion. In *ESWC*, pages 227–243.

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

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Gerhard Weikum, Xin Luna Dong, Simon Razniewski, Fabian Suchanek, et al. 2021. Machine knowledge: Creation and curation of comprehensive knowledge bases. *Foundations and Trends in Databases*, 10(2-4):108–490.

Qinyuan Wu, Mohammad Aflah Khan, Soumi Das, Vedant Nanda, Bishwamittra Ghosh, Camila Kolling, Till Speicher, Laurent Bindschaedler, Krishna P Gummadi, and Evimaria Terzi. 2024. Towards reliable latent knowledge estimation in llms: In-context learning vs. prompting based factual knowledge extraction. *arXiv*.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. In *ICLR*.

Maxwell Zeff. 2024. Openai unveils gpt-4o mini, a smaller and cheaper ai model. Available online: https://techcrunch.com/2024/07/18/openai-unveils-gpt-4o-mini-a-small-ai-model-powering-chatgpt/. Accessed: December 11, 2024.

Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. 2020. Semantics-aware BERT for language understanding. In *AAAI*.

A Further KB Content

Figures 5 and 6 provide further examples of GPTKB content. In the first case, all triples appear correct, but the predicate *famousFor* dominates the entity. In second case, most triples are correct, but the spouse is made up. In the online OpenReview appendix to this submission, we provide a larger sample of 10K triples.

Vienna

URI: https://<anonymized>/entity/Vienna

GPTKB entity

Statements (58)

Predicate	Object
gptkbp:instanceOf	City
gptkbp:area	414.87 km²
gptkbp:capital	gptkb:Austria
gptkbp:climate	Oceanic climate
gptkbp:currency	Euro
gptkbp:famousFor	gptkb:Vienna_Festival
	gptkb:Vienna_State_Ballet
	gptkb:Ludwig_van_Beethoven
	gptkb:Vienna_Philharmonic_Orchestra
	gptkb:Vienna_International_Book_Fair
	gptkb:Vienna_Fashion_Week
	gptkb:The_Vienna_Boys'_Choir
	gptkb:Vienna_International_Film_Festival
	gptkb:Vienna_Secession
	gptkb:Vienna_Jazz_Festival
	gptkb:Sigmund_Freud
	gptkb:Vienna_University_of_Economics_and_Business
	gptkb:Vienna_Coffee_Festival

Figure 5: Excerpt of GPTKB triples for subject *Vienna*.

Jorge Cham

URI: https://<anonymized>/entity/Jorge_Cham

GPTKB entity

Statements (13)

Predicate	Object
gptkbp:instanceOf	Person
gptkbp:almaMater	gptkb:Georgia_Institute_of_Technology
gptkbp:birthPlace	gptkb:Panama
gptkbp:children	2
gptkbp:genre	Humor
https://www.w3.org/2000/01/rdf-schema#label	Jorge Cham
gptkbp:knownFor	gptkb:PHD_Comics
gptkbp:nationality	American
gptkbp:occupation	Cartoonist
	Filmmaker
	Writer
gptkbp:spouse	Tanya K. K. Cham
gptkbp:website	http://www.phdcomics.com

Figure 6: Excerpt of GPTKB triples for subject *Jorge Cham* of PhDComics.

B Class distribution by BFS-level

Figure 7 shows the most frequent classes by BFS-level. As one can see, diversity increases towards lower BFS-levels, and some classes, like *awards*, only emerge later.
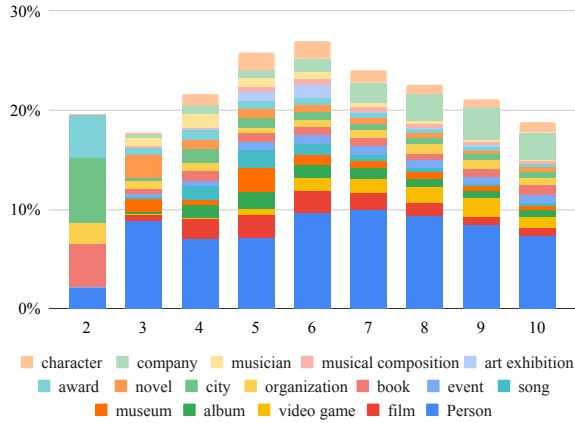

Figure 7: Most frequent classes by BFS-level. The root level (#1) is excluded from the chart.

C Prompts

All prompts below are aided by OpenAI's structured output feature, which allows defining a specific JSON schema, that the output has to adhere to. Therefore, the description of the output format in the prompts is less expansive than what is common in many other works.

Figure 8 contains the prompt used for triple elicitation. Figure 9 shows the prompt for NER. Figure 10 lists the prompt for taxonomy construction.

D Analysis of temporal cutoff

In Figure 11, we plot the frequency of values for the *year* property. One can observe two things: 1) A steady increase of frequency with recency, likely mirroring the growth of web corpora over the years, which made more recent content overrepresented. 2) A sudden dropoff between 2023 (549 triples) and 2024 (75 triples), matching the self-declared knowledge cutoff of the GPT-4o-mini LLM.
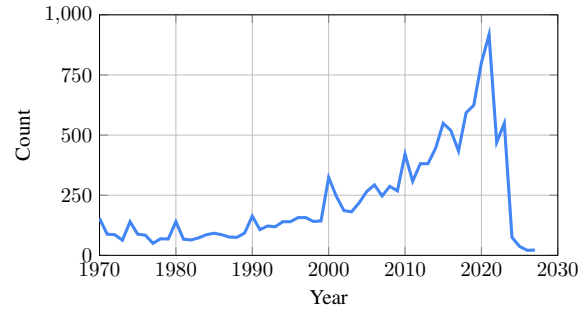

Figure 11: Number of triples for the *year* property, per year. The sudden drop from 2023 to 2024 matches the self-declared 2023 knowledge cutoff of GPT-4o-mini.

E Comparison of LLMs

In Table 4, we compare GPT-4o-mini with GPT-4o and Llama 3.1 70B in terms of accuracy. All LLMs are evaluated on a similar-sized set of entities (20K). One can observe significant accuracy differences, consistent with the order of their (estimated) parameter size, and general benchmark results.

	GPT 4o-mini*	Llama 3.1 70B	GPT 4o
Web-verified triples	0.38	0.69	0.78
Entities on Wikidata	0.78	0.83	0.88
Web-verified entities	0.80	0.95	0.98

Table 4: Entity and triple verifiability comparison between different LLMs on a similar-sized set of entities. GPT-4o-mini* corresponds to the first 5 layers of what is reported in Figure 3.

F Sources for KB comparison

Sources of KB sizes in the comparison presented in Table 2: **Wikidata**: See https://www.wikidata.org/wiki/Wikidata:Statistics and https://grafana.wikimedia.org/d/000000175/wikidata-datamodel-statements. **Wikidata5m**: (Wang et al., 2021). **Yago4.5**: (Suchanek et al., 2024). **DBpedia**: English version as per Table 2 in (Lehmann et al., 2015). **NELL**: As per Figure 5 (left) in (Mitchell et al., 2018). **Reverb**: As per (Lin et al., 2012) and https://web.archive.org/web/20220307185343/https://openie.allenai.org/.

You are a knowledge base construction expert. Given a subject entity, return all facts that you know for the subject as a list of subject, predicate, object triples. The number of facts may be very high, between 50 to 100 or more, for very popular subjects. For less popular subjects, the number of facts can be very low, like 5 or 10.

Important:
- If you don't know the subject, return an empty list.
- If the subject is not a named entity, return an empty list.
- If the subject is a named entity, include at least one triple where predicate is "instanceOf".
- Do not get too wordy.
- Separate several objects into multiple triples with one object.

Figure 8: Prompt for knowledge elicitation.

You are an expert on named entity recognition (NER). Your task is to classify if given phrases are named entities (e.g., persons, organizations, works of art), or not (e.g., literals, dates, URLs, verbose phrases). Each phrase is given to you in a line.

Figure 9: Prompt for named-entity recognition (NER).

You are a knowledge base construction expert. Your task is to initialize a seed taxonomy with general categories, which you will update later with given classes. Please return only the seed taxonomy in json form with indentation.

Class: <new class>

You are a knowledge base construction expert. Your task is to create a taxonomy for a knowledge base. Beforehand, you need to give each given class a score describing how general it is. The score is an integer ranging only from 1, for the most general concept, to 10, for the most specific concept. Please return only the score of the given class.

Candidate branches: <branches of current node>
Class: <class to add>

You are a knowledge base construction expert. Your task is to integrate a given class into the taxonomy. If the given class is a subclass of one of the candidate branches, return only the exact name of that branch. Otherwise, return only NULL.

Taxonomy: <taxonomy from current node>
Class: <class to add>

You are a knowledge base construction expert. Your task is to update the given taxonomy with the given class. You can consider the categorization of the taxonomy, but you can not modify the names of the classes in the taxonomy. Please return only the updated taxonomy in JSON form.

Figure 10: Prompts for seed taxonomy construction (top), class generality scoring (upper middle), recursive class insertion check (lower middle), and (sub)taxonomy update (bottom).