A Wikidata-based Benchmark for Evaluating LLM-based Question Answering

Medina Andresel[1], Jakob D. Hurst[2] and Sven Schlarb[1]

AIT Austrian Institute of Technology GmbH, Center for Digital Safety & Security, Giefinggasse 4, 1210 Vienna, Austria.
[1]*Competence Unit Data Science & Artificial Intelligence*
[2]*Competence Unit Cooperative Digital Technologies*

Abstract

While Large Language Models (LLMs) demonstrate impressive retrieval of isolated facts, their ability to maintain logical consistency across complex relational paths and temporal shifts remains under-evaluated. This paper introduces a novel benchmarking framework that utilizes Wikidata to perform automated fact-checking of LLM outputs. We focus on four core reasoning dimensions: spatial traversal, temporal sovereignty, relational succession, and quantitative aggregation. By grounding our "Ground Truth" in the Wikidata knowledge graph, we expose systemic factual incorrectness and reasoning failures. We evaluate multiple local LLMs across varying scales (4B, 7B, 9B, 27B) demonstrating that while models are factually aware, they frequently fail at tasks that require combining facts across time, space, or multiple reasoning steps. While larger LLMs seem to be more robust, the model size alone does not guarantee consistent improvements. We also find that evaluation method matters: fuzzy matching reveals many partially correct answers missed by exact string matching, whereas off-the-shelf LLM-as-judge frameworks appear less reliable for computing precision, recall, and F1 in settings that require structured factual comparison and numerical correctness.

Keywords

Wikidata, LLMs evaluation, multi-hop reasoning, benchmarking

1. Introduction

Large Language Models (LLMs) are increasingly integrated into software architectures to assist human decision-making, and they are used as reasoning engines within autonomous agents and expert systems. Evaluating their factual correctness and reasoning consistency is of therefore of utmost importance.

Although many LLMs are capable of generating syntactically correct and semantically sound text, they often lack a grounded understanding of causal relationships related to physical and temporal constraints of the real world [1]. This fundamental limitation manifests as *hallucinations* which denote fluent and plausible outputs that are factually incorrect, unsupported, or inconsistent with external evidence [2] and which are difficult to detect reliably using automatic metrics alone [3]. Beyond the generation of basic false claims, models struggle with the challenge of underlying context, often failing to maintain logical consistency when considering complex spatial and temporal boundaries [1].

To address these vulnerabilities, researchers are increasingly advocating for a shift toward "Trustworthy AI" by design. This paradigm requires moving beyond static performance benchmarks to prioritize verifiable assessments of model reliability and robustness [4]. By demanding transparent and high-quality assessments of their reasoning skills, the boundaries of LLM capabilities and the limitations of using them as trust-critical components become visible.

In this paper we describe a benchmarking methodology to assess the ability of a model to maintain logical consistency and historical continuity across various spatio-temporal scenarios. We are using the Wikidata knowledge graph as ground truth to verify model outputs in four reasoning dimensions: spatial traversal, temporal consistency, relational succession, and quantitative aggregation.

Third Workshop on Knowledge Graphs and Neurosymbolic AI (KG-NeSy 2026), in ISWC 2026 Workshops Joint Proceedings, October 25–26, 2026, Bari, Italy

medina.andresel@ait.ac.at (M. Andresel); jakob.hurst@ait.ac.at (J. D. Hurst); sven.schlarb@ait.ac.at (S. Schlarb)
0009-0002-4424-7817 (M. Andresel); 0000-0003-2270-1353 (J. D. Hurst); 0000-0003-3717-0014 (S. Schlarb)

© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-05

CEUR-WS.org/Vol-4313/kgnesy-paper9.pdf

A primary reason for selecting Wikidata is that many state-of-the-art LLMs are trained on information derived from Wikipedia and therefore tend to perform well on factual retrieval tasks. As a result, Wikidata provides an appropriate foundation for evaluating their reasoning capabilities, particularly because logical deduction requires combining multiple pieces of information to derive new facts. For example, a model may correctly identify the location of countries and list their neighboring states, yet still fail to determine all possible routes between *country A* and *country B*. Therefore, our framework aims to expose systemic failure modes, such as the inability of models to navigate geographic regions or resolve national boundaries that have changed over centuries[5].

A secondary reason for selecting Wikidata is that it enables the construction of precise, reproducible, and programmatically verifiable ground-truth answers through SPARQL queries. This ensures that model outputs can be evaluated against authoritative reference results rather than subjective judgments, making it possible to systematically measure reasoning performance across complex spatio-temporal tasks. Evaluating spatio-temporal reasoning requires testing distinct reasoning patterns. We created six tasks to represent different combinations of spatial and temporal operations: point-in-time lookup, temporal slicing, pathfinding, sequence retrieval, relational ordering, and interval aggregation. Retrieval results are not used to evaluate the factual correctness of LLMs, but are used as indicators to determine their reasoning abilities. The small, curated dataset allows verification of ground-truth answers by hand thus mitigating Wikidata's potential internal inconsistencies.

In recent years, LLMs have emerged also as evaluators to measure context relevance, faithfulness and answer accuracy. RAGAS [6] was introduced as an automated, largely reference-free evaluation framework that is being actively used to assess performance on question answering tasks. While broadly adopted, it is not yet fully established how reliable such LLM-as-judge frameworks are when evaluating structured factual answers, particularly in settings that require entity matching, numerical correctness, or the computation of precision, recall, and F1.

In this paper, we present a flexible benchmarking framework that supports:

1. the creation of novel reasoning tasks and test cases grounded in Wikidata, enabling the automatic computation of ground-truth answers;
2. the specification of suitable evaluation scoring functions depending on the identified answer type;
3. and the systematic comparison of different LLM-based question answering methods and evaluation strategies across multiple reasoning dimensions.

This design allows the benchmark to be extended with new tasks while preserving a clear link between test-case generation, verifiable reference answers, and task-specific evaluation criteria. Relying on this framework, we generate test data from a manually curated set of questions and assess local open-weight LLMs' multi-hop reasoning and LLM-as-judge performance.

The paper is structured as follows: in Section 2 we give an overview of related works including similar benchmarks, in Section 3 we present the configurable and reusable benchmarking process and describe the evaluation metrics while Section 4.1 we present some preliminary results using open-weighted on premise LLMs. The paper offers concluding remarks and future work in Section 5.

2. Related Work

Research in the area of evaluating the performance of LLMs has been documented in recent meta-reviews, such as the comprehensive survey of AI benchmarks and datasets by Ivanov and Penchev [7]. In this context, frameworks like TrustLLM [8] provide a taxonomy for assessing trustworthiness on specific dimensions such as truthfulness, safety, and robustness. Additionally, broader initiatives such as the European Union's TrustAI project [9] address the general challenges of AI reliability.

However, while comprehensive surveys [7] and frameworks such as TrustLLM are essential to establish performance baselines, their reliance on static evaluation datasets comes with a major drawback: these datasets are highly susceptible to data contamination. Because open benchmarks like MMLU [10]

are publicly accessible, they are inevitably ingested into the pre-training corpora of newer models. This systemic test leakage results in the absence of a true "hold-out set" for LLM evaluation, often leading to artificially inflated performance metrics [11].

To address the limitations of static benchmarks and the lack of reliable hold-out sets, recent research has increasingly turned to Knowledge Graphs (KGs) as structured evaluation mechanisms for LLMs. For example, FLEEK [12] extracts claims from the text generated by the LLM and cross-validates them against trustworthy KG triples. Similarly, Luo et al. [13] proposed a framework to assess the factual accuracy using automatically generated question formats directly from the KG. These approaches are valid for fact-checking because they evaluate basic subject-predicate-object triples.

Beyond basic fact retrieval, other frameworks evaluate how well LLMs comprehend graph structures and perform multi-hop reasoning. Guo et al. [14] created a dataset to evaluate to what extent LLMs interpret graph data correctly, while Tang et al. [15] developed a benchmarking tool specifically designed to evaluate LLMs on abstract graph computational problems. Moving closer to semantic reasoning, the Mintaka dataset [16] explicitly maps complex evaluation queries to Wikidata entities, and KG-LLM-Bench [17] tests the ability of models to deduce logical paths between entities. Other approaches such as PAT-Questions [18] propose a benchmark for present-anchored temporal question answering that leverage Wikidata entities as starting points in the benchmarking process. Similarly, LC-QuAD [19, 20] use a bottom-up approach by traversing the graph to construct queries and then translating them into natural language.

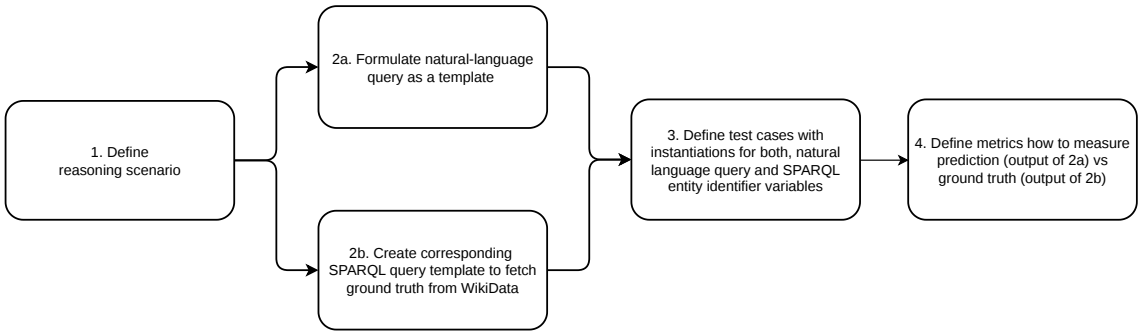
Our approach follows a top-down strategy and instead of generating test cases from simple KG triples, we manually curate a set of questions that require multi-hop reasoning constrained by explicit temporal and spatial qualifiers. For example, rather than merely evaluating whether an LLM can retrieve a static birth location, our queries require the model to deduce the specific historical sovereign state that controlled a birthplace at the exact time of birth by cross-referencing temporal boundaries within the graph. Furthermore, to ensure scalability, each manually curated question-query pair serves as a seed to expand the dataset, generating a series of test cases by iterating over variable parameters such as specific years and entity names.

3. Creating a Wikidata Benchmark for Question Answering

In the following we describe the reproducible benchmark creation methodology which allows creating an extensible dataset for assessing the deductive reasoning and evaluation capabilities of LLMs. Figure 1 illustrates the benchmark creation process we applied. The process begins with human-designed scenarios (1) which include reasoning challenge for LLMs. Each scenario has a natural-language query (2a) and a SPARQL query template (2b). Each scenario has a set of test cases with instantiations for both, natural language query and SPARQL entity identifier variables (3), which allow creating a set of concrete test cases. Finally, the metrics are defined how to measure prediction (output of 2a) vs ground truth (output of 2b).

A scenario consists of a natural-language question with replaceable key-value slots, which correspond with a SPARQL query template. The questions are curated by the authors, following the criteria that each scenario (i) targets a specific reasoning type, (ii) is answerable over Wikidata, and (iii) supports scalable test-case generation, such that expanding the set of admissible key-value instantiations yields a correspondingly larger evaluation dataset. For each scenario, the SPARQL template is manually written to return the ground-truth answer to the natural language question, and this correspondence is validated by instantiating a single test case and confirming that the SPARQL output matched the expected answer by the authors. Once validated, the evaluation dataset is generated iteratively by substituting the key values, as shown in the following scenario generation methodology example. Note that an extensive validation of the full ground truth dataset is subject to future work. Nevertheless, this methodology allows the generation of additional scenarios with spatial and temporal reasoning complexity.

Figure 1: Benchmark creation process.



3.1. Define Reasoning Scenarios

In the first step, a reasoning scenario is defined using an identifier, e.g., $historical_countries_of_city$, with a corresponding description and admissible key-value instantiations:

> Provide a list of countries or political entities to which the city $\{city\}$ belonged over time, starting from the year $\{year\}$. This includes any historical shifts in sovereignty or territorial changes that affected the city.

Fulfilling the request requires more than looking up a static fact. Reasoning is in this example involved by the following aspects:

- A timeline must be established starting precisely at year and track continuous changes up to the present day, ignoring anything that happened before that date.
- A specific geographic point (the city) must be assigned within political borders while considering that borders move, empires dissolve, and cities change names (e.g., Danzig to Gdańsk) over centuries.
- Cities may have unclear sovereignty assignments, such as short occupation periods (e.g., during wars).

The scenario is a multi-step reasoning problem because a chronological chain of historical facts must be constructed while verifying the territorial boundaries for each specific era.

Part of the definition of a scenario is the instructions for the LLM which primarily is to make sure the output format is adequate for applying precise evaluation metrics.

> You must respond with CSV only. Do not include explanations, comments, or Markdown formatting.
> Output format: country1,country2,country3, etc.

Because we demand a list of countries, this allows applying classical metrics based on precision (how many of the proposed countries are in the list of the expected ones) and recall (how many of the expected countries in the list of the proposed ones).

3.2. Natural-language and SPARQL Query Templates

Each scenario provides a natural-language (NL) and SPARQL query template pair. The NL template is used to construct LLM prompts, while the SPARQL template generates the target queries for evaluation.

The NL template can include one or more instance placeholders. For example, in the following request, $city$ and $year$ are placeholders replaced by concrete values during instantiation:

> In Europe, give me the list of countries to which the city $\{city\}$ belonged to over time since $\{year\}$.

Similarly, the SPARQL template (see Listing 1) uses placeholders like $city_id$ and $year$ to generate an executable query. Unlike the NL template, the SPARQL template uses the exact Wikidata entity identifier defined by $city_id$ instead of the city name $city$ to ensure precise retrieval and filtering of results.

```
# Countries a city (e.g., Trieste (Q546)) has belonged to over time
PREFIX wd:   <http://www.wikidata.org/entity/>
PREFIX p:    <http://www.wikidata.org/prop/>
PREFIX ps:   <http://www.wikidata.org/prop/statement/>
PREFIX pq:   <http://www.wikidata.org/prop/qualifier/>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT DISTINCT ?country ?countryLabel ?start ?end WHERE {
  wd:{{city_id}} p:P17 ?statement .
  ?statement ps:P17 ?country .

  OPTIONAL { ?statement pq:P580 ?start . }   # start time
  OPTIONAL { ?statement pq:P582 ?end . }     # end time

  # Filter: keep statements that are open-ended OR extend past 1900
  FILTER(!BOUND(?end) || ?end >= "{{year}}T00:00:00Z"^^xsd:dateTime)

  SERVICE wikibase:label { bd:serviceParam wikibase:language "en". }
}
ORDER BY ?start
```

Listing 1: SPARQL Query Template

Each evaluation scenario defines a schema of test case configuration parameters. Every placeholder object consists of a natural-language identifier ($name$), a corresponding knowledge graph variable identifier ($sparql_key$), and a human-readable definition. An example of such schema is summarized in Table 1.

Table 1
Evaluation Scenario Placeholders

NL Prompt Name	SPARQL Key	Description
`city`	`city_id`	Geographic city entity
`year`	`year`	Temporal constraint (Year)

During instantiation, the $name$ placeholder (e.g., $city$) is populated with a natural-language literal to construct the LLM prompt. Simultaneously, the $sparql_key$ parameter (e.g., $city_id$) injects a unique WikiData entity identifier—such as a Wikidata $Q - number$— into the SPARQL template. This way the ambiguous NL name for the LLM prompt is separated from the unambiguous identifier for the graph database query.

Table 2
Examples of Concrete Instantiated Test Cases

Case ID	City Name (NL)	Wikidata QName (`city_id`)	Target Date (`year`)
`Trieste_1800`	Trieste	`wd:Q546`	1800-01-01
`Trieste_1900`	Trieste	`wd:Q546`	1900-01-01
`Strasbourg_1870`	Strasbourg	`wd:Q6602`	1870-01-01
`Strasbourg_1910`	Strasbourg	`wd:Q6602`	1910-01-01
`Gdansk_1945`	Gdańsk	`wd:Q1792`	1945-01-01

To evaluate a model's temporal and spatial reasoning performance, templates are systematically populated using targeted test cases (examples are shown in Table 2). Each test case is uniquely identified by a $case_ID$ and contains a set of parameters configured to challenge specific historical transitions or contested geopolitical periods. This means that for each each scenario using the assigned NL and SPARQL templates, a series of test cases is generated.

These cases challenge LLMs to account for shifting geopolitical borders and sovereignty transitions. For instance, the cases targeting Trieste (wd:$Q546$) - where wd is the is the URI prefix http://www.wikidata.org/entity/ for Wikidata entities - in 1800 and 1900 are related to a century of shifting sovereignty within European empires. Similarly, for Strasbourg (wd:$Q6602$) in 1870 and 1910 there are territorial ownership changes following the Franco-Prussian War, while in the case of Gdańsk (wd:$Q1792$) 1945 was a transition year at the end of World War II.

3.3. Scoring Metrics

Following the generation of these datasets, the evaluation process splits into parallel paths to facilitate comparison against an authoritative ground truth. In the prediction path, NL query instances and their associated output schema definitions are submitted to the LLM for response generation. Simultaneously, in the ground-truth path, the framework executes corresponding SPARQL queries against the Wikidata knowledge graph to extract verifiable data.

The final scoring phase compares the LLM's prediction against the Wikidata-derived ground truth using a combination of exact string matching, fuzzy matching, and RAGAS-based metrics to account for the inherent variability in generative outputs. The culminating output of this workflow provides a comprehensive assessment of the model's performance, specifically quantifying its logical reasoning aptitude and its proficiency as an LLM-as-a-judge for evaluation tasks.

To evaluate the quality of the generated outputs, we compute precision, recall, and F1 score under three complementary matching strategies: exact string matching, fuzzy string matching, and LLM-as-judge matching. Precision measures the proportion of predicted items that are correct, while recall measures the proportion of gold-standard items that are successfully retrieved. The F1 score provides the harmonic mean of precision and recall, thereby balancing both criteria.

More precisely, let TP, FP, and FN denote the number of true positives, false positives, and false negatives, respectively. Precision, recall, and F1 score are defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{1}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{2}$$

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{3}$$

The values for TP, FP, and FN depend on the matching strategy used to compare the predicted answer list with the ground-truth answer list. Below, we briefly describe how matches are identified under each strategy.

Exact String Matching Under this approach, a predicted item is considered correct only if it matches exactly a corresponding reference item at the string level. Formally, for a predicted string $\overline{y}$ and a reference string y, a match is counted if $\overline{y} = y$. This criterion is intentionally strict and does not allow for spelling variations, differences in word order, abbreviations, or semantically equivalent paraphrases. Exact matching therefore provides a conservative estimate of performance and is most suitable when the expected output has a canonical textual form.

Fuzzy String Matching The second evaluation strategy is fuzzy matching. Here, similarity between a predicted answer and a ground-truth answer is computed using a weighted fuzzy string similarity measure. The weighted ratio score combines multiple string comparison strategies, including full-string, partial-string, and token-based matching, making it robust to spelling variations, reordered words, and partial overlaps. Although we use the weighted ratio score in our experiments, other fuzzy similarity measures may also be used depending on the desired matching behavior. Formally, for a predicted string $\overline{y}$ and a reference string y, a match is counted if $sim(\overline{y}, y) \geq t$, where sim denotes a similarity scoring function with values in $[0, 1]$, and t is a predefined threshold. Order of the predicted answers does not matter as comparison is done pairwise.

LLM-as-Judge In this setting, an LLM-based evaluator is used to determine whether the predicted answers match the ground-truth answers according to semantic criteria rather than surface-level string similarity. This allows the evaluation to account for paraphrases, synonymous expressions, abbreviations, and other meaning-preserving reformulations that may not be captured by exact or fuzzy matching. In our experiments, we implement this strategy using the off-the-shelf precision, recall, and F1-style metrics provided by the RAGAS framework. In this framework, the predicted response and the reference answer are decomposed into factual claims, and an LLM-based natural language inference step is used to determine whether claims from one text are supported by the other. Precision is computed based on the proportion of claims in the predicted response that are supported by the reference, recall is computed based on the proportion of reference claims that are supported by the predicted response, and F1 combines these two quantities within the framework. Although we use RAGAS for this implementation, the evaluation strategy is not specific to this framework. Other LLM-based evaluation frameworks or prompting schemes could be used to make similar semantic matching decisions.

Comparison of different matching strategies Since the expected answers vary across test cases, the choice of matching strategy can affect the resulting precision, recall, and F1 scores. We therefore compare exact matching, fuzzy matching, and LLM-as-judge matching to assess different notions of correctness, ranging from strict textual matching to lexical similarity and semantic equivalence. This comparison is particularly important for LLM-as-judge evaluation, which is a relatively new approach and requires careful assessment, since LLM-based evaluators may themselves produce inconsistent or incorrect judgments.

4. Experimental Evaluation

In this section we describe the preliminary experiments we performed to assess reasoning capabilities of various small open-weights LLMs. Data and code for benchmark generation and evaluation can be accessed using repository: https://zenodo.org/records/22644304.

4.1. Evaluation dataset

The benchmark generation process is being used to define several reasoning scenarios and create test cases for each reasoning task. In Table 3 are described the reasoning tasks used in our experiments. For each task, we also define the corresponding SPARQL query template and a number of test cases by identifying concrete entities to be used for the instantiation of the test case. In Table 4 we provide the number of test cases per task, query type complexity described by metrics such as number of joins, maximum path, use of optional or union etc., and the expected answer type. In total we obtain 100 questions and SPARQL queries which are being used to perform the evaluation.

Table 3
Task descriptions, abbreviations, and reasoning dimensions

Abbrev.	Task	Dimension	Description
BHS	birthplace historical sovereignty	Temporal Sovereignty (person-anchored)	Identify which sovereign state controlled a persons birthplace at the time of their birth.
COMA	countries organizations members at year	Temporal Sovereignty (point in time)	Retrieve the set of countries that were members of a given international organisation in a specified year.
CTC	countries to cross	Spatial Traversal	Determine which countries must be crossed when traveling overland between two locations.
HCC	historical countries of city	Temporal Sovereignty (sequential)	List the sovereign states that have historically governed a given city over time.
PS	presidential successor	Relational Succession	Identify the president who directly succeeded a given officeholder in a specified country.
SYPR	sum of years person in role	Quantitative Aggregation	Compute the total number of years a person held a particular role, aggregated across potentially non-contiguous terms.

Table 4
Dataset statistics by task. Query complexity is summarized using compact structural indicators: joins (J), union/optional blocks (U/O), filters (F), values clauses (V), maximum property-path length (MP), and GROUP BY with SUM aggregation (G). In the MP column, $*$ denotes an unbounded variable-length property path.

Task	#Test Cases	Query Complexity						Answer Type
		J	U/O	F	V	MP	G	
CTC	9	2	1	1	–	3	–	string set
HCC	26	3	2	1	–	3	–	string set
COMA	26	5	2	2	1	4	–	string set
PS	15	4	1	2	–	*	–	string
BHS	11	3	2	2	–	4	–	string
SYPR	13	2	1	1	2	–	1	numeric
Total	**100**							

4.2. Experimental setup

In our preliminary experiments, we evaluate open-weight LLMs that can be executed on premise. This choice reflects the intended deployment setting, where privacy, reproducibility, and independence from external API providers are important considerations. The selected models range from approximately 4 billion to 27 billion parameters.

We evaluate models from different families, including EU-based models: Mistral, EuroLLM, US-based: Gemma and China-based: Qwen and DeepSeek models, more specifically: `qwen3.5:4b`, `mistral:7b`, `qwen3.5:27b`, `gemma4:e4b`, `gemma:7b`, `euroLLM-9b-instruct`, and `deepseek-r1:7b`. This selection is sufficient for a preliminary study because it covers multiple model sizes, architectures, and instruction-tuning strategies, while avoiding dependence on a single model family. It also allows us to analyze the trade-off between computational cost and output quality in a local deployment scenario. The temperature parameter was set to 0.0 to minimize non-determinism, whereas all remaining parameters were left at their default values. For fuzzy string matching we used PolyFuzz library version 0.4.3[1] using edit-distance RapidFuzz model with a threshold value of 0.65.

The experiments are conducted on an on-premise HPC cluster with 28 nodes, comprising 11 GPU nodes and 17 CPU nodes, approximately 2,500 CPU threads, 17.2 TB of CPU RAM, and 33 GPUs with a total of 1.5 TB of GPU memory. The GPU partition includes one dual NVIDIA GH200 node and two

[1] https://maartengr.github.io/PolyFuzz/

Table 5
Summary: macro-averaged Precision (exact match) per model and task with the Avg. column reporting the average precision across tasks for each model.

llm_model	BHS	COMA	CTC	HCC	PS	SYPR	**Avg.**
qwen3.5:4b	0.23	0.65	0.70	0.54	0.47	0.08	0.44
gemma4:e4b	0.09	0.65	0.22	0.64	0.40	0.08	0.35
deepseek-r1:7b	0.27	0.65	0.46	0.46	0.07	0.46	0.40
gemma:7b	0.09	0.69	0.63	0.56	0.53	0.15	0.44
mistral:7b	0.23	0.72	0.42	0.27	0.67	0.46	0.46
EuroLLM-9B-Instruct	0.09	0.67	0.26	0.50	0.13	0.23	0.31
qwen3.5:27b	0.27	0.75	0.69	0.60	0.60	0.38	0.54

Table 6
Summary: macro-averaged Recall (exact match) per model and task with Avg. column reporting the average recall across tasks for each model.

llm_model	BHS	COMA	CTC	HCC	PS	SYPR	**Avg.**
qwen3.5:4b	0.27	0.74	0.58	0.30	0.47	0.08	0.41
gemma4:e4b	0.09	0.65	0.32	0.38	0.40	0.08	0.32
deepseek-r1:7b	0.27	0.71	0.59	0.30	0.07	0.46	0.40
gemma:7b	0.09	0.61	0.53	0.42	0.53	0.15	0.39
mistral:7b	0.27	0.76	0.50	0.11	0.67	0.46	0.46
EuroLLM-9B-Instruct	0.09	0.72	0.40	0.37	0.13	0.23	0.32
qwen3.5:27b	0.27	0.73	0.49	0.35	0.60	0.38	0.47

Table 7
Summary: macro-averaged F1 (exact match) per model and task with Avg. column reporting the average F1 across tasks for each model.

llm_model	BHS	COMA	CTC	HCC	PS	SYPR	**Avg.**
qwen3.5:4b	0.24	0.67	0.61	0.37	0.47	0.08	0.41
gemma4:e4b	0.09	0.63	0.25	0.44	0.40	0.08	0.32
deepseek-r1:7b	0.27	0.65	0.46	0.33	0.07	0.46	0.37
gemma:7b	0.09	0.62	0.54	0.45	0.53	0.15	0.39
mistral:7b	0.24	0.72	0.43	0.15	0.67	0.46	0.45
EuroLLM-9B-Instruct	0.09	0.65	0.29	0.40	0.13	0.23	0.30
qwen3.5:27b	0.27	0.74	0.54	0.40	0.60	0.38	0.48

dual NVIDIA RTX6000-Pro Blackwell nodes for LLM inference, with access governed by HPC resource policies and regulated through SLURM Fair Share scheduling.

Regarding implementation, each LLM was hosted on premise using the Ollama[2] model-serving framework and accessed through the Python LangChain[3] library. No additional prompting strategy was applied; instead, each model received the question directly as derived from the corresponding test case. To enable structured answer evaluation, we defined a generic Pydantic schema and used the structured-output option of the LangChain Ollama client.

4.3. Results and Analysis

In Table 5, Table 6 and Table 7 we present the averaged precision, recall and F1 results for each model and each reasoning task using the exact string matching strategy. Blue color intensity reflects performance, with darker shades indicating higher precision, recall and F1 score[4]. One main observation is that precision, recall and F1 values per model and task are very comparable which can be explained by

[2]https://github.com/ollama/ollama
[3]https://github.com/langchain-ai/langchain
[4]White text is used solely for improving readability.

the fact that each task description usually has a small cardinality for expected and produced answers. Moreover macro-averaging over multiple test cases smoothens out individual differences making the precision, recall and F1 scores appear very similar for each model-task combination. Therefore we will focus on analyzing further the F1 results.

Robustness analysis. The results indicate that model performance is strongly task-dependent, with no single model dominating across all tasks. Overall, `qwen3.5:27b` achieves the highest average macro-F1, suggesting that increased model capacity generally improves robustness across tasks. However, this trend is not uniform: `qwen3.5:4b` outperforms its larger counterpart on CTC, while `mistral:7b` achieves the best performance on PS and ties for the best result on SYPR. Notably, `mistral:7b` remains highly competitive despite its smaller size, while `gemma4:e4b` does not consistently improve over the older `gemma:7b`.

Reasoning analysis. Based on the task descriptions and the reported results, the main observation is that the evaluated LLMs perform best on tasks that are closer to retrieval of known facts or bounded set membership, and worse on tasks requiring multi-step temporal, spatial, or quantitative reasoning. The strongest and most stable task is COMA: countries that were members of an organization in a specified year. This task belongs to the Temporal Sovereignty/point-in-time dimension, but it is largely a constrained set-retrieval problem. Across models, COMA consistently receives relatively high F1 values (≥ 0.62). Performance drops for HCC, which asks for the historical sovereign states that governed a city over time. Unlike COMA, this requires sequential temporal reasoning, because the model must identify multiple historical transitions and produce an ordered and complete set of sovereign state entities. Moreover, most of the models underperform on BHS task which reveal that person-anchored temporal sovereignty is more difficult. CTC shows moderate performance which suggests that some LLMs can approximate geographical traversal. PS, presidential successor, shows large differences across models. For example, mistral:7b performs very strongly, while deepseek-r1:7b and EuroLLM-9B-Instruct perform poorly. This suggests that relational succession is not uniformly difficult, but depends strongly on whether a model has internalized structured political chronology. SYPR, sum of years person in role, is among the most challenging tasks for most models. This task requires identifying all relevant terms in office, handling non-contiguous periods, computing durations, and summing them. The relatively low and variable results indicate that models struggle with multi-step quantitative aggregation, especially when the calculation depends on historical timelines rather than a single memorized value.

String Matching Strategies versus RAGAS. In Table 8 we show the difference when using Fuzzy string matching strategy. When evaluating using the fuzzy matching strategy, also answers that vary based on small syntax variations are considered to be a match, therefore it is somehow obvious that fuzzy string matching would be less restrictive and lead to moderately higher F1 scores. However it is worth mentioning that in some cases this strategy can lead to a few incorrect matches since in this setting we do not take into account the semantics of answers (E.g., "Finland" and "Iceland" match due to common substring "land").

In our RAGAS evaluation, we employed the `qwen2.5:8b` model to act as an evaluator and produce the numbers. The matching and computation of precision, recall and F1 are done internally in the RAGAS framework[5]. Compared to the fuzzy strategy, the RAGAS evaluation leads to worsen values, as shown in Table 9, compared to exact matching strategy. Moreover, the results show a substantial performance decline on the most challenging reasoning tasks, suggesting that the evaluator LLM struggle to resolve entities in complex spatio-temporal contexts. In addition, the large negative deltas for SYPR suggest that this RAGAS implementation is less suitable for precise quantitative aggregation tasks, where correctness depends on exact numerical agreement rather than semantic similarity.

[5]By using provided factual correctness metrics of RAGAS version 0.3.9: https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/factual_correctness/

Table 8
Delta F1 score gain from fuzzy over exact matching (macro-averaged per model and task)

llm_model	BHS	COMA	CTC	HCC	PS	SYPR
qwen3.5:4b	+0.24	+0.01	+0.06	+0.18	+0.13	+0.15
gemma4:e4b	+0.00	+0.04	+0.03	+0.15	+0.07	+0.08
deepseek-r1	+0.09	+0.02	+0.02	+0.16	+0.00	+0.00
gemma:7b	+0.27	+0.01	+0.00	+0.13	+0.13	+0.15
mistral	+0.33	+0.02	+0.00	+0.11	+0.13	+0.00
EuroLLM-9B-Instruct	+0.22	+0.03	+0.04	+0.21	+0.00	+0.00
qwen3.5:27b	+0.09	+0.01	+0.00	+0.23	+0.13	+0.00

Table 9
Delta F1 score from RAGAS over exact (macro-averaged per model and task)

llm_model	BHS	COMA	CTC	HCC	PS	SYPR
qwen3.5:4b	-0.24	-0.02	-0.06	+0.03	-0.13	-0.08
gemma4:e4b	-0.09	+0.11	+0.00	-0.06	-0.07	-0.08
deepseek-r1	-0.27	+0.02	+0.00	+0.05	+0.00	-0.46
gemma:7b	-0.09	+0.14	+0.00	+0.03	-0.07	-0.15
mistral	-0.24	+0.01	+0.00	+0.01	-0.07	-0.46
EuroLLM-9B-Instruct	-0.09	+0.05	+0.00	+0.06	+0.00	-0.23
qwen3.5:27b	-0.27	-0.02	-0.21	-0.02	-0.07	-0.38

5. Conclusion and Future Work

This paper introduced a Wikidata-grounded benchmarking framework for evaluating LLM performance on different reasoning tasks. By covering spatial traversal, temporal sovereignty, relational succession, and quantitative aggregation, the benchmark exposes limitations that are not visible in simple factual recall tasks. Our results show that local open-weight LLMs often contain relevant factual knowledge, but their performance remains highly task-dependent and declines when answers require multi-step reasoning over entities, time, space, or numerical aggregation. We also show that evaluation strategy can affect the measured results: fuzzy matching can recover partially correct answers missed by exact string matching, while off-the-shelf LLM-as-judge approaches such as RAGAS factual correctness appear less reliable when computing precision, recall, and F1 over structured answers types.

Future work will extend the benchmark with additional reasoning dimensions, larger and more diverse test sets, and more fine-grained answer-type-specific scoring functions. We also plan to investigate disambiguation strategies, especially for historical entities by integrating also alternative names and aliases. Restricting model-size comparisons to controlled within-family settings would further shed light on the effect of model scaling on reasoning performance, which we leave for future work. Another direction is to evaluate whether retrieval-augmented generation can improve performance on the most challenging tasks. Finally, future work should further examine the reliability of LLM-based evaluators across different reasoning settings, whilst at the same time increasing the experimental test cases for each reasoning task or scenario to gain stronger support on the overall performance claims.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT and Microsoft Copilot, in order to: Grammar and spelling check, Paraphrase and reword. After using these tools/services, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] D. Hutchinson, Mapping the Latent Past: Assessing Large Language Models as Digital Tools through Source Criticism, Journal of Digital History 3 (2024).

[2] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, A. Madotto, P. Fung, Survey of hallucination in natural language generation, ACM Computing Surveys 55 (2023) 1–38. doi:10.1145/3571730.

[3] J. Maynez, S. Narayan, B. Bohnet, R. McDonald, On faithfulness and factuality in abstractive summarization, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 1906–1919. doi:10.18653/v1/2020.acl-main.173.

[4] M. M. Ferdaus, M. Abdelguerfi, E. Loup, K. N. Niles, K. Pathak, S. Sloan, Towards trustworthy ai: A review of ethical and robust large language models, ACM Comput. Surv. 58 (2026). URL: https://doi.org/10.1145/3777382. doi:10.1145/3777382.

[5] F. Ortu, J. Yook, P. S. Pandey, K. Samway, B. Schölkopf, A. Cazzaniga, R. Mihalcea, Z. Jin, Preserving Historical Truth: Detecting Historical Revisionism in Large Language Models, 2026. URL: http://arxiv.org/abs/2602.17433. doi:10.48550/arXiv.2602.17433, arXiv:2602.17433 [cs].

[6] S. Es, J. James, L. Espinosa Anke, S. Schockaert, RAGAs: Automated evaluation of retrieval augmented generation, in: Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, Association for Computational Linguistics, St. Julians, Malta, 2024, pp. 150–158. doi:10.18653/v1/2024.eacl-demo.16.

[7] T. Ivanov, V. Penchev, Ai benchmarks and datasets for llm evaluation, 2024. URL: https://arxiv.org/abs/2412.01020. arXiv:2412.01020.

[8] Y. Huang, Q. Zhang, et al., Position: TrustLLM: Trustworthiness in large language models, in: Proceedings of the 41st International Conference on Machine Learning, volume 235 of *Proceedings of Machine Learning Research*, PMLR, 2024. URL: https://proceedings.mlr.press/v235/huang24x.html.

[9] TrustAI Consortium, TrustAI: Transparent, reliable and unbiased smart tool for AI, https://www.trustai.eu, 2020. Funded by the European Union's Horizon 2020 research and innovation programme.

[10] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. M. employment, D. Song, J. Steinhardt, Measuring massive multitask language understanding, in: International Conference on Learning Representations, 2021. URL: https://openreview.net/forum?id=d7KBjmI3GmQ.

[11] J. Haimes, C. Wenner, K. Thaman, V. Tashev, C. Neo, E. Kran, J. Schreiber, Benchmark inflation: Revealing llm performance gaps using retro-holdouts, 2024. URL: https://arxiv.org/abs/2410.09247. arXiv:2410.09247.

[12] F. F. Bayat, K. Qian, B. Han, Y. Sang, A. Belyi, S. Khorshidi, F. Wu, I. F. Ilyas, Y. Li, Fleek: Factual error detection and correction with evidence retrieved from external knowledge, 2023. URL: https://arxiv.org/abs/2310.17119. arXiv:2310.17119.

[13] L. Luo, T.-T. Vu, D. Phung, G. Haffari, Systematic assessment of factual knowledge in large language models, 2023. URL: https://arxiv.org/abs/2310.11638. arXiv:2310.11638.

[14] J. Guo, L. Du, H. Liu, M. Zhou, X. He, S. Han, Gpt4graph: Can large language models understand graph structured data ? an empirical evaluation and benchmarking, 2023. URL: https://arxiv.org/abs/2305.15066. arXiv:2305.15066.

[15] J. Tang, Q. Zhang, Y. Li, N. Chen, J. Li, Grapharena: Evaluating and exploring large language models on graph computation, 2025. URL: https://arxiv.org/abs/2407.00379. arXiv:2407.00379.

[16] P. Sen, A. F. Aji, A. Saffari, Mintaka: A complex, natural, and multilingual dataset for end-to-end question answering, in: N. Calzolari, C.-R. Huang, H. Kim, J. Pustejovsky, L. Wanner, K.-S. Choi, P.-M. Ryu, H.-H. Chen, L. Donatelli, H. Ji, S. Kurohashi, P. Paggio, N. Xue, S. Kim, Y. Hahm, Z. He, T. K. Lee, E. Santus, F. Bond, S.-H. Na (Eds.), Proceedings of the 29th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Gyeongju, Republic of Korea, 2022, pp. 1604–1619. URL: https://aclanthology.org/2022.coling-1.138/.

[17] E. Markowitz, K. Galiya, G. V. Steeg, A. Galstyan, Kg-llm-bench: A scalable benchmark for evaluating llm reasoning on textualized knowledge graphs, 2025. URL: https://arxiv.org/abs/2504.

07087. arXiv:2504.07087.

[18] J. A. Meem, M. S. Rashid, Y. Dong, V. Hristidis, Pat-questions: A self-updating benchmark for present-anchored temporal question-answering, in: ACL (Findings), volume ACL 2024 of *Findings of ACL*, Association for Computational Linguistics, 2024, pp. 13129–13148.

[19] P. Trivedi, G. Maheshwari, M. Dubey, J. Lehmann, Lc-quad: A corpus for complex question answering over knowledge graphs, in: International Semantic Web Conference, Springer, 2017, pp. 210–218.

[20] M. Dubey, D. Banerjee, A. Abdelkawi, J. Lehmann, Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia, in: ISWC (2), volume 11779 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 69–78.