A Comprehensive Benchmark for Evaluating LLM-Generated Ontologies

Julien Plu[1], Oscar Moreno Escobar[1], Edouard Trouillez[1], Axelle Gapin[1,†] and Raphaël Troncy[2]

[1]LettrIA, 13 Rue des Petits Hôtels, 75010 Paris, FRANCE
[2]EURECOM, Campus SophiaTech, 450 Route des Chappes, 06410 Biot, FRANCE

Abstract

This paper presents a methodology for evaluating ontologies that are automatically generated by Large Language Models (LLMs). Our approach combines quantitative metrics that compare generated ontologies with respect to a human-made reference and qualitative user assessments across diverse domains. We apply this methodology to evaluate the ontologies produced by various LLMs, including Claude 3.5 Sonnet, GPT-4o, and GPT-4o-mini. The results demonstrate the benchmark's effectiveness in identifying strengths and weaknesses of LLM-generated ontologies, providing valuable insights for improving automated ontology generation techniques.

Keywords

LLM, Knowledge Engineering, Ontology Development, Benchmark, Evaluation

1. Introduction

The advent of LLMs has opened new avenues for automated ontology generation, such as in [1, 2, 3, 4, 5]. However, evaluating the quality and utility of these generated ontologies presents a significant challenge. This paper introduces a comprehensive benchmark methodology designed to assess LLM-generated ontologies through both quantitative and qualitative measures through 30 criteria. Our benchmark aims to provide a standardized approach for comparing different LLM-generated ontologies, evaluating their accuracy, completeness, and practical utility across various domains. All results and documents are available on GitHub https://github.com/jplu/ontology-benchmark.

2. Ontology-Toolkit

The Ontology-Toolkit is our LLM-based tool for ontology generation, designed to evolve with the benchmark, facilitate experimentation, support domain-specific ontologies, and promote wider adoption. It features a modular process with a user-friendly interface, allowing the refinement of the results at each step. Our primary goal was to minimize user interactions, as our target users preferred a non-conversational application for quick ontology generation, making approaches such as OntoChat[6] unsuitable. The Ontology-Toolkit functions as follows:

1. Generate Classes: This initial step produces essential ontology components based on input documents, specified domain, and use case. For example, given medical results analysis documents, an appropriate domain could be *medicine and pharmacology*. Users can manually refine the generated classes. The number of classes is crucial; too many can lead to overly specific classes (e.g., *City* vs *Place*) instead of desired hierarchies like *Place -> PopulatedPlace -> City*. Overly specific classes can narrow the ontology's applicability.

2. Generate Questions: This step creates competency questions to guide ontology development and deduce class relations. The questions are based on the input documents and classes from Step 1.

ISWC 2024 Special Session on Harmonising Generative AI and Semantic Web Technologies, November 13, 2024, Baltimore, Maryland
[†]Author contributed during her internship.
julien@lettria.com (J. Plu); oscar@lettria.com (O. M. Escobar); edouard@lettria.com (E. Trouillez); axelle@lettria.com (A. Gapin); raphael.troncy@eurecom.fr (R. Troncy)

Users can manually add, remove, or update these questions. Our experiments showed that questions should be as generic as possible, cover the entire scope of the document, and refer to classes rather than specific instances. Given that example: "Barack Obama was born in Hawaii," appropriate questions are "Where was [Person] born?" or "Is [Place] the birthplace of [Person]?"

3. Generate Ontology: The final step instructs an LLM to produce the ontology in RDF using Turtle syntax. This process uses only the classes, questions, domain definition, and use case; original documents are not needed at this stage.

The Ontology-Toolkit allows for refinement at each step, balancing automation with user control. The objectives are to streamline the ontology generation process while maintaining flexibility for diverse use cases. A demo is available at https://tinyurl.com/iswc-demo.

3. Benchmark

Our benchmark is applicable to any automated ontology generation approach. We developed our own benchmark instead of using existing tools like OWLUnit[1] for three main reasons: 1) Our target users, including non-technical individuals, lack the required expertise in Semantic Web technologies; 2) The unpredictable structure of generated ontologies makes pre-defined query test suites challenging and potentially biased; 3) Existing tools cannot effectively test ontology coverage, such as class hierarchy, properties, or missing classes. Our benchmark is divided into two evaluations: 1) Quantitative: Assesses the quality of generation parameters; 2) Qualitative: Enables non-technical users to evaluate ontology coverage.

This approach addresses the limitations of existing tools and provides a more suitable evaluation method for our use case and target audience.

3.1. Quantitative Evaluation

We present one example of our quantitative evaluation on a single ontology, focusing on a single parameter (the utility of providing a use case). This evaluation has been applied to 19 other ontologies.

Our results show whether providing a use case to the model improves the generated ontology, with the answer correlating to the selected LLM. Prior evaluations determined that 50 classes and 50 competency questions yield the best results for this use case, which is why these numbers are set for this evaluation.

Ontology. We evaluate using a *real-world* use case by comparing the ontology generated by the Ontology-Toolkit with a reference ontology manually developed by professional knowledge engineers. This reference ontology belongs to the financial domain and enables to provide information about companies as well as specific financial events. It comprises 37 classes, 54 object properties, and 48 data properties. It also follows an event-based design pattern, primarily created to capture information on events affecting companies such as mergers, acquisitions, settlements, and legal disputes. The *Event* superclass plays a central role, with no fewer than 15 subclasses, including *MergerEvent*, *SettlementEvent*, and *LegalDisputeEvent*. The ontology also includes 20 object properties indicating the participation of companies and organizations in these events, such as *biddingParticipant*, *regulatorParticipant*, and *victimParticipant*.

Experiments. The generated ontology with Ontology-Toolkit used a corpus of 30 documents on financial events from BBC and Reuters, averaging 900 tokens each. The domain is defined as *market-moving events*, and the use case is *Assess and analyze the impact of market-moving events, the parties involved, and their subsequent effects*.

Two configurations named *with_use_case_50* and *no_use_case_50* are evaluated. Six ontologies were generated for this experiment, specifying a use case or not and using three LLMs: Claude 3.5 Sonnet,

[1] https://github.com/luigi-asprino/owl-unit

GPT-4o, and GPT-4o-mini. This setup allowed for a comparative analysis of the models' performance based on the presence or absence of an explicit use case.

Evaluation Results. For each generated ontology, a manual analysis was conducted, comparing the presence of classes and properties between the generated and reference ontologies. We also assessed the presence of hallucinations. We had a degree of tolerance in the comparison: missing classes or properties may correspond to elements with broader or narrower meanings, which were noted separately. For example, the object property *hasLocation* is broader than *country*, while the data property *usesCryptocurrency* is narrower than *currency*. Two sets of evaluations were conducted using different metrics: 1) Accuracy, and (2) Precision, Recall, and F1 Score. *Accuracy*: The overall proportion of matching classes and properties between the generated and reference ontologies. *Precision*: The ratio of concepts or relations in the generated ontology that are present in the reference ontology. *Recall*: The ratio of concepts or relations in the reference ontology that are also present in the generated ontology. *F1 Score*: The harmonic mean of precision and recall.

Model	Ontology	Type	Yes	Narrower	Broader	No	Total Yes, Narrower, Broader
Claude 3.5 Sonnet	With use case 50	classes	13 (35.14%)	2 (5.41%)	11 (29.73%)	11 (29.73%)	70.27%
		properties	25 (24.51%)	0 (0.00%)	43 (42.16%)	34 (33.33%)	**66.67**%
	Without use case 50	classes	13 (35.14%)	5 (13.51%)	10 (27.03%)	9 (24.32%)	**75.68**%
		properties	15 (14.71%)	0 (0.00%)	44 (43.14%)	43 (42.16%)	57.84%
GPT-4o	With use case 50	classes	13 (35.14%)	1 (2.70%)	11 (29.73%)	12 (32.43%)	67.57%
		properties	5 (4.90%)	7 (6.86%)	0 (0.00%)	90 (88.24%)	11.76%
	Without use case 50	classes	5 (13.51%)	1 (2.70%)	16 (43.24%)	15 (40.54%)	59.46%
		properties	2 (1.96%)	7 (6.86%)	38 (37.25%)	55 (53.92%)	46.08%
GPT-4o-mini	With use case 50	classes	12 (32.43%)	1 (2.70%)	13 (35.14%)	11 (29.73%)	70.27%
		properties	1 (0.98%)	0 (0.00%)	37 (36.27%)	64 (62.75%)	37.25%
	Without use case 50	classes	15 (40.54%)	0 (0.00%)	12 (32.43%)	10 (27.03%)	72.97%
		properties	5 (4.95%)	0 (0.00%)	17 (16.83%)	79 (78.22%)	21.78%

Table 1
Accuracy comparison of 6 ontologies generated using 3 LLMs with respect to a reference ontology

Tables 1 and 2 present the results for each configuration and model. Claude 3.5 Sonnet demonstrates stable performance with and without a use case being specified. GPT-4o-mini ranks second, particularly in class generation, and shows an improvement in property precision when a use case is given. In contrast, GPT-4o exhibits weaker results, especially in generating properties when a use case is provided. These findings indicate that incorporating a use case improves the consistency of ontologies generated by Claude 3.5 Sonnet and GPT-4o-mini, but has less impact on GPT-4o.

Modèle	Ontology	Precision	Recall	F1 Score	Class Precision	Class Recall	Class F1 Score	Property Precision	Property Recall	Property F1 Score
Claude 3.5 Sonnet	With use case 50	0,30	0,30	0,30	0,46	0,68	0,55	0,20	0,17	0,18
	Without use case 50	0,46	0,46	0,46	0,58	1,14	0,76	0,33	0,22	0,26
GPT-4o	With use case 50	0,57	0,36	**0,44**	0,60	0,84	0,70	0,53	0,18	0,27
	Without use case 50	0,88	0,10	0,18	1	0,11	0,20	0,83	0,10	0,18
GPT-4o-mini	With use case 50	0,39	0,25	0,31	0,52	0,73	0,61	0,22	0,08	0,12
	Without use case 50	0,4	0,28	0,33	0,53	0,76	0,62	0,24	0,10	0,14

Table 2
Precision, Recall, and F1 score comparison of 6 ontologies using 3 LLMs with respect to a reference ontology

3.2. Qualitative Evaluation

We performed a user-based evaluation in which participants were asked to use Ontology-Toolkit with Claude 3.5 Sonnet to develop an ontology on the domain of their choice, selecting relevant documents, and then assessing its quality using an evaluation grid. A detailed *How-To* guide was provided to ensure users could independently navigate the process of both generating and evaluating the ontology.

Experiments. We asked each participant (n=8) to select a document (minimum 3 pages, plain text, without tables or images) on a subject of their choice. Options included Wikipedia pages, articles, and documentation. We recommended using documents from domains they were familiar with, as this would enable them to evaluate the generated ontology more effectively. 4 users selected a document in English and 4 users selected a document in French.

We successfully generated eight ontologies, covering topics ranging from API documentation to the Capetian dynasty. The generated ontologies have labels in English or French depending on the source document language without modifying the prompt. To ensure optimal visualization and exploration of the ontologies, we used Protégé[2] as our primary tool. The original text documents, along with information on their domains and use cases, as well as the resulting generated ontologies are available in the GitHub repository.

Evaluation Grid. Building on the ontology evaluation criteria provided by Ouyang et al. [7], we developed an evaluation grid tailored to assess ontology quality composed of 30 criteria. The grid is divided into three sections: the first evaluates classes, the second focuses on properties, and the third provides an overall assessment. Each section uses criteria related to accuracy (e.g. classes accurately represent entities from the input text), completeness (e.g. key relations between classes are captured comprehensively by properties), clarity (e.g. properties are described using clear and consistent terminology), coverage (e.g. no significant information from the source material, relevant to the ontology, is omitted) and relevance (e.g. the ontology focuses on classes relevant to the intended domain) ensuring a thorough and structured evaluation of the ontology.

The grid uses an evaluation scale where each criterion is rated from *1* to *5*, where *1* represents poor performance, *2* indicates needs improvement, *3* reflects satisfactory performance, *4* denotes good performance, and *5* represents excellent performance. At the end of each section, users are encouraged to provide honest feedback, highlighting areas for improvement and noting particularly well-executed aspects.

Evaluation Grid Results. Results are reported in Table 3. The average score by user varies from *3.0* to *5.0*. The average score per section also shows some variation, with better results observed for the Properties section compared to the others. The complete grid results are available in the GitHub repository. Generally, performances are good, with most evaluations showing balanced scores across sections, as seen in the cases of users *1*, *3*, *7*, and *8*. However, some scores fell below the average, notably in the Overall section, highlighting concerns with the ontology's accuracy and consistency. User 2, for instance, gave lower scores, reflecting these concerns. The Properties section received the highest average score (*4.19*), while Classes also performed well with an average of *3.99*, suggesting that the ontology's class structure is satisfactory. However, the Overall section had the lowest average score (*3.74*), indicating that the general usability of the ontology still needs optimization according to users.

Users	S1: Classes	S2: Properties	S3: Overall	Average
1	4	4	3.7	3.9
2	3.1	3.7	2.1	3
3	3.8	4.4	4.1	4.1
4	4.7	4.6	3.7	4.3
5	4.1	3.9	3.8	3.9
6	5	5	5	5
7	3.8	3.7	4	3.8
8	3.4	4.2	3.6	3.7
Average All Users	3.99	4.19	3.74	3.97

Table 3
Grid scores results across 8 users when assessing classes and properties of the generated ontologies

To better understand these results, we identified the lowest-scoring criteria in each section and analyzed the user feedback for each area. The feedback for the class section highlights both strengths and weaknesses. The lowest scores are observed on criteria *1.1*, *1.9*, and *1.10*: users suggest that a more refined hierarchical structure and a reduction in class overload would be beneficial. While the generated ontology performs well overall, it lacks classes for specific use cases and occasionally contains irrelevant classes. The class hierarchy also needs improvement, as some related classes could be grouped under broader categories.

Users generally found the generated properties sufficient, with no major areas for improvement identified. Only criterion *2.10* received a low score. Key properties are present, providing logical links

[2] https://protege.stanford.edu/

between classes, even those not directly relevant to the use case. However, property specificity was a concern, with users noting that overly detailed distinctions between concepts (such as *zone* and *zone policy*) could hinder usability. Visualization challenges were also noted, though these concerns relate more to the tool used for visualization than the generated ontology itself. Some users struggled to navigate the ontology and understand the relationships between property domains and ranges.

For the overall section, users found that criteria *3.5*, *3.7*, and *3.8* did not meet their expectations, giving them the lowest scores. Feedback indicates that while the automatically generated ontology offers a strong and well-structured foundation, its shallow, flat design is limiting. There are too many top-level classes and insufficient depth in the hierarchy. To enhance its usefulness, we should group related classes under parent categories and create more specific subclasses. Additionally, the ontology currently lacks focus on specific use cases and is missing some crucial classes and properties. By addressing these issues, we can create a more cohesive, practical, and comprehensive ontology.

4. Conclusion

Our proposed benchmark offers a robust framework for evaluating LLM-generated ontologies. The combination of quantitative comparisons against professional references and qualitative user assessments provides a holistic view of ontology quality. Results highlight the potential of LLMs in ontology generation, with Claude 3.5 Sonnet showing particular promise. The benchmark also reveals areas for improvement, notably in overall coherence and accuracy. This work contributes to the field by providing a standardized evaluation method, paving the way for advancements in automated ontology generation techniques.

For future work, to enhance the validity of the generated ontology and verify its accuracy, we propose developing a text-to-graph approach. This method would enable us to assess the quality of data extracted by the generated ontology. Additionally, we are currently investigating the implementation of a GraphRAG approach, which would allow users to ask natural language questions as a final testing endpoint.

References

[1] J. H. Caufield, H. Hegde, V. Emonet, N. L. Harris, M. P. Joachimiak, N. Matentzoglu, H. Kim, S. Moxon, J. T. Reese, M. A. Haendel, P. N. Robinson, C. J. Mungall, Structured Prompt Interrogation and Recursive Extraction of Semantics (SPIRES): a method for populating knowledge bases using zero-shot learning, Bioinformatics 40 (2024). URL: https://doi.org/10.1093/bioinformatics/btae104.

[2] P. Mateiu, A. Groza, Ontology engineering with Large Language Models, 2023. URL: https://arxiv.org/abs/2307.16699. arXiv:2307.16699.

[3] V. K. Kommineni, B. König-Ries, S. Samuel, From human experts to machines: An LLM supported approach to ontology and knowledge graph construction, 2024. URL: https://arxiv.org/abs/2403.08345. arXiv:2403.08345.

[4] S. Bischof, E. Filtz, J. X. Parreira, S. Steyskal, LLM-based Guided Generation of Ontology Term Definitions, in: European Semantic Web Conference (ESWC), Industry Track, 2024.

[5] Y. Rebboud, L. Tailhardat, P. Lisena, R. Troncy, Can LLMs Generate Competency Questions?, in: European Semantic Web Conference (ESWC), 2024.

[6] B. Zhang, V. A. Carriero, K. Schreiberhuber, S. Tsaneva, L. S. González, J. Kim, J. de Berardinis, Ontochat: a framework for conversational ontology engineering using language models, arXiv preprint arXiv:2403.05921 (2024).

[7] L. Ouyang, B. Zou, M. Qu, C. Zhang, A method of ontology evaluation based on coverage, cohesion and coupling, in: 8^{th} International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), volume 4, 2011, pp. 2451–2455. doi:10.1109/FSKD.2011.6020046.