The Role of Generative AI in Competency Question Retrofitting

Reham Alharbi[0000−0002−8332−3803], Valentina Tamma[0000−0002−1320−610X], Floriana Grasso[0000−0001−8419−6554], and Terry R. Payne[0000−0002−0106−8731]

University of Liverpool, Liverpool, UK
{R.Alharbi, V.Tamma, Floriana, T.R.Payne} @Liverpool.ac.uk

Abstract. Competency Questions (CQs) are essential in ontology engineering; they express an ontology's functional requirements as natural language questions, offer crucial insights into an ontology's scope and are pivotal for various tasks, e.g. ontology reuse, testing, requirement specification, and pattern definition. Despite their importance, the practice of publishing CQs alongside ontological artefacts is not commonly adopted. We propose an approach based on Generative AI, specifically Large Language Models (LLMs) for retrofitting CQs from existing ontologies and we study how the control parameters in two LLMs (i.e. `gpt-3.5-turbo` and `gpt-4`) affect their performance and investigate the interplay between prompts and configuration for retrofitting viable CQs.

Keywords: Competency Questions · Large Language Models · Ontology Engineering Methodologies

1 Introduction

Competency Questions (CQs) [9] are natural language questions characterising the scope of knowledge represented by an ontology. They model the functional requirements that an ontology or ontology-based information system should satisfy to achieve its intended purpose.

Within the early stages of ontology development, they can be used to suggest possible concepts and relationships the ontology should model [15,20,21,23,25], and can also be used in subsequent phases to verify and validate the knowledge encapsulated in the ontology [5,10]. However, as it is not always possible to obtain the original CQs when working with many existing ontologies, the `RETROFIT-CQs` approach was proposed [1] to automatically generate candidate CQs from ontology triples by leveraging Large Language Models (LLMs). This work represented a significant shift towards a hybrid model of knowledge representation, merging explicit and parametric knowledge [4,18]. Although an initial analysis (conducted across various ontologies from the CORAL repository [8]) confirmed that CQs could not only be generated, but that they also closely matched the intended design CQs; a number of research questions remained regarding the nuances of LLMs, particularly regarding the research question: *To what extent do the control parameters, such as creativity settings and the specificity of prompts, affect*

the performance of `RETROFIT-CQs`*?* This question is addressed by evaluating two hypotheses: that additional context specified in the LLM prompt will enhance model comprehension and response accuracy; and that more robust and reliable CQs can be obtained by reducing the creativity parameter (i.e. *temperature*) of the LLMs, as defined in GPT API documentation.[1]

We investigate these hypotheses by examining the resulting efficacy of our `RETROFIT-CQs` approach. In particular, the parameters' effect on the stochasticity of LLM-generated text was analysed, by evaluating its default and deterministic settings, together with the influence that different prompts (i.e. the natural language texts used for communicating with LLMs) has on the resulting CQs. We run a comparative analysis of the generated CQs against those found in existing benchmarks, e.g. CORAL [8] and the CQs dataset in [26]. The study confirms the first hypothesis; that the addition of context within the prompts can result in a more precise and coherent LLM response. However, contrary to our expectations, the second hypothesis was not supported. Our findings suggest that the overall performance of the LLMs, when used by our `RETROFIT-CQs` approach, is robust and replicable, exhibiting only marginal stochasticity when changing the control parameters (including changes to the creativity parameter, prompt, and even choice of LLM).

The paper is structured as follows: the use of LLMs is briefly discussed in Section 2, before detailing the methodology used (including a brief outline of the `RETROFIT-CQs` approach) in Section 3. The results are presented in Section 4, and we summarise our conclusions and outline future research directions in Section 5.

2 Background

Large Language Models (LLMs) have shown promise for a plethora of tasks, including the automatic generation of natural language questions [14]. Autoregressive LLMs such as those in the GPT family [16] are deep learning models trained on vast data corpora, and are used to predict the next word in a sequence based on the previous context. Through their use, a new text generation paradigm has emerged whereby a 'prompt' guides the generation of various outputs [14]. These prompts, consisting of strings prepended to the input context, incorporate control elements (such as keywords) to guide the text generation [13]. Initial research has already investigated the significant impact of different prompt designs on the performance and outputs of LLMs [24], effectively laying the groundwork for the field of prompt engineering [13]. Despite the impressive capability that LLMs have to produce syntactically correct and complex natural language, ensuring that this output is meaningful and accurate remains a challenge [3]. A more nuanced view suggests that LLMs, when combined with traditional symbolic approaches, can play a vital role in knowledge engineering workflows, leading to a new era in knowledge representation that merges explicit and parametric knowledge [2,3]. The effectiveness of these meth-

[1] `https://platform.openai.com/docs/api-reference/chat/create`

ods must be validated by addressing LLM-related challenges such as expressivity vs decidability [18], thoroughly evaluating approaches that incorporate LLM components [4], and tackling issues stemming from insufficient information about LLMs, including their reliability and replicability [12].

3 Experimental setup

The `RETROFIT-CQs` approach [1] we propose generates candidate CQs by utilising a pipeline that consists of three phases: (i) extract triples from the ontology to represent its statements; (ii) generate an LLM prompt by integrating the triples into a template that also includes contextual cues; and (iii) filter the resulting questions generated by the LLM to remove duplicates and irrelevant questions. In this study, we investigate the impact on the quality of the candidate CQs by investigating the role of various zero-shot prompts and the influence that different creativity parameter settings have on CQ generation, by comparing deterministic and default values. This directly relates to the two hypotheses identified in Section 1:

- *Hypothesis 1: Prompting an LLM with more contextual information results in the generation of more concise and coherent responses.*
  This hypothesis stems from the premise that additional relevant information could enhance the model's understanding and response accuracy.
- *Hypothesis 2: Employing the default value of the creativity parameter 'temperature' in an LLM tends to produce responses that are more varied and less focused, in contrast to using a deterministic value which is expected to yield factual responses more closely aligned with the original text.*

In this study we focus on two OpenAI GPT models: `gpt-3.5-turbo-0613` and `gpt-4-0613`[2], that are extensively used as Language-Models-as-a-Service (LMaaS) [12]. We focus on these models because they expose little information to the user, and because we wanted to investigate their stochastic behaviour. A further study with a wider range of LLMs is currently in preparation.

We compare the CQs generated by our approach against a benchmark comprising two existing CQ repositories: CORAL [8] and the dataset in [26]. Four ontologies were selected from the CQ benchmark based on three criteria: (i) the ontologies were produced by different developers; (ii) they represent various domains; and (iii) each had a significant number of published CQs. The selected ontologies are: (1) 'Video Game'[19]; (2) 'African Wildlife'[11] (3) 'Dem@care' [8]; and (4) 'VICINITY Core' [8]. The characteristics of each of these ontologies (i.e. number of both design CQs and triples) are stated in Table 1.

For each of these ontologies, we generate CQs using the different prompts and the two GPT models. These prompts allow us to examine the impact of transitioning from general to granular when generating candidate CQs; and to understand how LLMs can achieve the highest accuracy in the targeted task.

[2] `https://platform.openai.com/docs/models/overview`

	Prompt	Unmatched CQs (#) %							
		gpt-3.5-turbo				gpt-4			
		CP=0.0		CP=0.7		CP=0.0		CP=0.7	
Video Game	P1	(5)	7.57%	(3)	4.54%	(1)	1.51%	(1)	1.51%
Design CQs: 66	P2	(5)	7.57%	(8)	12.12%	(3)	4.54%	(1)	1.51%
# of Triples: 57	P3	(2)	3.03%	(2)	3.03%	(1)	1.51%	(1)	1.51%
African Wildlife	P1	(2)	14.28%	(1)	7.14%	(2)	14.28%	(4)	28.57%
Design CQs: 14	P2	(2)	14.28%	(2)	14.28%	(2)	14.28%	(2)	14.28%
# of Triples: 26	P3	(1)	7.14%	(1)	7.14%	(2)	14.28%	(2)	14.28%
Dem@care	P1	(11)	10.28%	(7)	6.54%	(7)	6.54%	(7)	6.54%
Design CQs: 107	P2	(10)	9.34%	(9)	8.41%	(4)	3.73%	(9)	8.41%
# of Triples: 146	P3	(3)	2.80%	(3)	2.80%	(2)	1.86%	(5)	4.67%
VICINITY Core	P1	(5)	8.77%	(3)	5.26%	(4)	7.01%	(2)	3.50%
Design CQs: 57	P2	(4)	7.01%	(1)	1.75%	(2)	3.50%	(2)	3.50%
# of Triples: 226	P3	(1)	1.75%	(1)	1.75%	(1)	1.75%	(1)	1.75%

Table 1: Number (percent) of unmatched Design CQs for each prompt and LLM, comparing deterministic (CP=0.0) and default (CP=0.7) values.

Furthermore, we investigate the effect on the accuracy of the generated CQs of injecting more context to the prompt. In a previous study [1] we discussed how LLMs can generate *'narrative questions'*, i.e. questions that can elicit expansive, descriptive responses [7], often representing subjective views. For example, the CQs that `gpt-4` generates for the triple *'Achievement, isAchievementInGame, Game'* with Prompt 1 include *"Can you recall an achievement in a game that you found extremely satisfying to unlock"*, that is not a suitable CQ. The injection of *context* limits the generation of such questions and ensures that the candidate CQs remain focused on defining the ontology's scope and providing context in terms of *how, where, when, why, who* [23]. We define three prompt templates, each providing increasingly richer context:

P1 *General Competency Questions*: this instructs an LLM to generate competency questions for a given statement: *["Based on <statement>, generate a list of competency questions" avoid using narrative questions + statement].*

P2 *Definitions of Competency Questions:* this prompt explicitly includes the definition of a CQ: *["Based on the <statement>, generate a list of competency question. Definition of competency questions: the questions that outline the scope of an ontology and provide an idea about the knowledge that needs to be entailed in the ontology." avoid using narrative questions + statement].*

P3 *Use of a Role with Definitions of Competency Questions:* this contextualises the prompt by specifying the role of "Ontology Engineer", implying a more methodological approach to question formulation that focuses on the structural aspects of the ontology development, with the aim of explicitly generating CQs by including the definition of CQs: *["As an ontology engineer, generate a list of competency questions based on the <statement>. Definition of competency questions: the questions that outline the scope of ontology and provide an idea about the knowledge that needs to be entailed in the ontology" avoid using narrative questions + statement].*

	Prompt	LLMs	No. Q.	Number of CQs		Performance		
				Candidate	Validated	Prec.	Rec.	F1
Video Game	P1	gpt-3.5-turbo	555	555	251	0.452	0.980	0.619
		gpt-4	776	591	482	0.816	0.998	0.898
	P2	gpt-3.5-turbo	570	569	399	0.701	0.988	0.820
		gpt-4	1033	810	639	0.789	0.995	0.880
	P3	gpt-3.5-turbo	570	565	434	0.844	0.995	0.914
		gpt-4	1197	911	759	0.833	0.999	0.908
African Wildlife	P1	gpt-3.5-turbo	215	213	136	0.638	0.986	0.775
		gpt-4	496	373	156	0.418	0.987	0.588
	P2	gpt-3.5-turbo	260	258	151	0.585	0.987	0.735
		gpt-4	423	357	186	0.521	0.989	0.683
	P3	gpt-3.5-turbo	270	256	185	0.723	0.995	0.837
		gpt-4	255	174	94	0.540	0.979	0.696
Dem@care	P1	gpt-3.5-turbo	1360	1339	474	0.354	0.977	0.520
		gpt-4	2039	1660	512	0.308	0.987	0.470
	P2	gpt-3.5-turbo	1435	1418	403	0.284	0.976	0.440
		gpt-4	2574	2042	633	0.310	0.994	0.473
	P3	gpt-3.5-turbo	1461	1386	622	0.449	0.995	0.619
		gpt-4	2850	2129	656	0.308	0.997	0.471
VICINITY Core	P1	gpt-3.5-turbo	2179	2119	501	0.236	0.990	0.382
		gpt-4	4320	3428	1122	0.327	0.996	0.493
	P2	gpt-3.5-turbo	2219	2150	547	0.254	0.993	0.405
		gpt-4	4549	3505	1333	0.380	0.999	0.550
	P3	gpt-3.5-turbo	2249	2115	947	0.448	0.999	0.618
		gpt-4	4958	3863	1485	0.384	0.999	0.555

Table 2: Summary for each prompt with deterministic creativity value (CP=0.0).

One of the documented limitations of the GPT models is their stochastic nature [12]. We investigate the diversity of text generated by the LLMs by adjusting the creativity (CP) or *temperature* parameter, whose value is in the range $[0, 2]$. We explore two CP settings: (i) a *deterministic* value of 0.0, which eliminates stochasticity and focuses on the consistent generation of text; and (ii) the *default value*, that allows the generation of more diverse and creative responses.[3]

As identical prompts can produce varied responses depending on the setting of the creativity parameter, in this study, we explore how the creativity parameter's default and deterministic settings impact prompt performance.

4 Results

The evaluation contrasted the performance of our `RETROFIT-CQs` approach using two LLMs (`gpt-3.5-turbo` and `gpt-4`) for the three prompt templates described in Section 3 across statements extracted from the four ontologies. The results[4] for two control parameters – determininstic (CP=0.0) and default (CP=0.7) – are discussed below. Table 1 presents the number of *design CQs* (i.e. the original CQs provided for each ontology in the benchmark datasets) for which no corresponding CQs were generated by the evaluated approach.

[3] The default setting, as of December 2023, was 0.7 but has since been adjusted to 1.0

[4] `https://github.com/SemTech23/RETROFIT-CQs_GPT`

	Prompt	LLMs	No. Q.	Number of CQs		Performance		
				Candidate	Validated	Prec.	Rec.	F1
Video Game	P1	gpt-3.5-turbo	543	543	348	0.641	0.991	0.779
		gpt-4	1249	1205	963	0.799	0.999	0.888
	P2	gpt-3.5-turbo	567	567	365	0.644	0.979	0.777
		gpt-4	1084	1061	852	0.803	0.999	0.890
	P3	gpt-3.5-turbo	570	565	429	0.759	0.995	0.861
		gpt-4	797	765	628	0.821	0.998	0.901
African Wildlife	P1	gpt-3.5-turbo	206	205	128	0.624	0.992	0.766
		gpt-4	274	266	128	0.481	0.970	0.643
	P2	gpt-3.5-turbo	260	259	141	0.544	0.986	0.701
		gpt-4	441	437	173	0.396	0.989	0.565
	P3	gpt-3.5-turbo	265	262	198	0.756	0.995	0.859
		gpt-4	517	459	229	0.499	0.991	0.664
Dem@care	P1	gpt-3.5-turbo	1329	1319	452	0.343	0.985	0.508
		gpt-4	2134	2101	552	0.263	0.987	0.415
	P2	gpt-3.5-turbo	1428	1406	423	0.301	0.979	0.460
		gpt-4	2681	2628	780	0.297	0.989	0.457
	P3	gpt-3.5-turbo	1475	1459	616	0.422	0.995	0.593
		gpt-4	2929	2811	863	0.307	0.994	0.469
VICINITY Core	P1	gpt-3.5-turbo	2177	2160	573	0.265	0.995	0.419
		gpt-4	4444	4276	1430	0.334	0.999	0.501
	P2	gpt-3.5-turbo	2202	2199	596	0.271	0.998	0.426
		gpt-4	4824	4695	1723	0.367	0.999	0.537
	P3	gpt-3.5-turbo	2265	2230	947	0.425	0.999	0.596
		gpt-4	4975	4787	1887	0.623	0.999	0.767

Table 3: Summary for each prompt with default creativity value (CP=0.7).

Tables 2 and 3 report the number of CQs generated for the deterministic and default control parameter settings and for each prompt template respectively, and present: (i) number of generated questions (No. Q.); (ii) filtered questions in the final output (No. Candidate CQs); (iii) number of validated candidate CQs against existing CQs (No. of Validated CQs); i.e. those CQs that appear in the benchmark dataset; and (iv) Performance Metrics (i.e. Precision, Recall & F1 score). In validating the candidate CQs, we use `SBERT` [22] to assess the similarity between CQs while mitigating the effect of paraphrasing, or the use of different morphological structures (e.g. plurals) on the similarity assessment. If *No. of Validated CQs* denotes the number of CQs that are assessed as having similar meaning by `SBERT` (true positives), and if `No. Unmatched CQs` corresponds to the number of questions in the benchmark that do not match any of the generated CQs (true negatives), then $Precision = \frac{No.\ of\ Validated\ CQs}{No.\ Q}$ and $Recall = \frac{No.\ of\ Validated\ CQs}{No.\ of\ Validated\ CQs + No.\ Unmatched\ CQs}$.

The results show that neither changing the creativity parameter nor changing the specificity of the prompts has a significantly adverse effect on the number of valid CQs generated, based on the high recall scores observed in both Tables 2 and 3. Our approach consistently achieves a recall of 0.96 or higher for all prompts, regardless of the creativity parameter settings. Thus, the majority of the design CQs catalogued in the benchmark are matched (Table 1), supporting the claim that viable CQs can be generated. However, the precision varies

	Candidate CQs
gpt-3 P3	What are the different types of achievements that can be earned in a game?
	How is an achievement defined within the context of a game?
	What criteria must be met in order for a player to earn an achievement in a game?
	Can achievements be earned in any type of game, or are they specific to certain genres or platforms?
	How are achievements tracked and recorded within a game?
	Are there any limitations or restrictions on the number of achievements that can be earned in a game?
	Can achievements be earned by individual players or are they only awarded to teams or groups?
	Are there any rewards or benefits associated with earning achievements in a game?
	Can achievements be earned retroactively, or must they be earned during a specific timeframe?
	How are achievements displayed or showcased within a game?
gpt-3 P1	Can you provide examples of specific achievements you have earned within the game?
	How do you track and measure your achievements in the game?
	Have you ever faced any challenges or obstacles while trying to achieve a specific goal within the game? How did you overcome them?
	Can you describe a time when you felt particularly proud of an achievement you earned in the game?
	Have you ever collaborated with other players to achieve a specific in-game accomplishment? How did you contribute to the team effort?
	Can you explain the process you follow to unlock new achievements within the game?
	Can you provide an example of a time when you had to demonstrate a specific skill or strategy in order to achieve an in-game accomplishment?
Design CQ: `What are the types of achievements in this game?`	
Design CQ: `What are the types of achievements a game can have?`	

Table 4: Candidate CQs Generated for the Video Game Ontology Triple *'Achievement isAchievementInGame Game'* with CP=0.0. Green CQs match both Design CQs (at the bottom of the Table), the ones in blue match only the first (blue) Design CQ; likewise the ones in red match the second design CQs.

and is influenced by both the prompt's specificity and the creativity parameter. Notably, for all four ontologies, the highest precision is achieved using Prompt 3, where we defined the role of the ontology developer and the definition of CQs. The lowest overall precision was recorded for prompt P1 in VICINITY Core (with `gpt-3.5-turbo`), where CQs were requested without additional clarification. This prompt lacked contextual information and used the deterministic value (CP=0.7), although a low precision was also observed with CP=0.0 using P1 for both the VICINITY Core and Dem@care data sets. As a result, several design CQs were not matched (see the "Unmatched CQs %" column in Table 1), and other irrelevant CQs were generated.

When considering the specificity of the prompt, P3 overall achieved higher precision scores than either P1 or P2 for both creative parameters considered. This can be illustrated using the Video Game ontology. Table 4 presents the candidate CQs for the triple *'Achievement isAchievementInGame Game'* corresponding to each prompt, with the Design CQs relating to this triple. Even when CP=0.0, P1 (which only provided context) elicited narrative questions [7]. In contrast, the inclusion of both the role of ontology engineer and the definition of CQs in P3 resulted in CQs that align more closely with the non-narrative di-

Ontology	Category	all P in GPT3		all P in GPT4		all P in all LLMs	
		0.0	0.7	0.0	0.7	0.0	0.7
Video	# Candidate CQs	1638	1675	2312	3031	3950	4706
Game	# Overlapping CQs (%)	1402 (85.59%)	1382 (82.51%)	2087 (90.27%)	2827 **(93.27%)**	3538 (89.57%)	4274 **(90.82%)**
African	# Candidate CQs	727	726	904	1162	1631	1888
Wildlife	# Overlapping CQs (%)	693 (95.32%)	681 (93.80%)	884 **(97.79%)**	1100 (94.66%)	1578 (96.75%)	1861 **(98.57%)**
Dem@care	# Candidate CQs	4143	4184	5832	7540	9975	11724
	# Overlapping CQs (%)	3725 (88.91%)	3655 (87.36%)	5173 (88.70%)	7085 **(93.97%)**	8906 (89.28%)	10826 **(92.34%)**
VICINITY	# Candidate CQs	6384	6589	10796	13758	17180	20347
Core	# Overlapping CQs (%)	5590 (87.56%)	5574 (84.60%)	9507 (88.06%)	12414 **(90.23%)**	15254 (88.79%)	18262 **(89.75%)**

Table 5: Summary of the overlap in candidate CQs for each ontology, including the total number of Candidate CQs (# Candidate CQs), the total number of overlapping CQs, and their percentage (# Overlapping CQs (%)).

rective [23]. These questions are designed to collect objective information on the classification and properties of achievements within games (thus aligning with the requirements of ontology engineering). This explains the high precision score obtained using P3, as the majority of generated CQs match those in the dataset, compared to P1's three matching CQs.

An oft-stated concern regarding the use of LLMs is that their performance is not replicable, as the results of repeated prompts can vary. Therefore, we also examined the overlap in the resulting CQs; i.e. verifying that the same CQs were generated despite changes to the prompt, 'temperature value' and LLM used. Our results demonstrate that: (i) there is only a marginal difference in the recall value between Tables 2 and 3 when varying the creativity parameter, suggesting this value has a marginal effect on the viability of the resulting CQs; and (ii) there is a consistency in the CQs generated by different prompts when one looks at the overlap (Table 5) of the generated CQs, regardless of the LLM used. For example, the overlap of candidate CQs for the Dem@care ontology, across all prompts is 88.91% and 88.70% respectively for `gpt-3.5-turbo` and `gpt-4` when CP=0.0 (conversely, 87.36% and 93.97% for CP=0.7). It was noted, however, that even when CP=0.0, there was a slight variance in the resulting CQs over repeated prompts, thus supporting the claim that both `gpt-3.5-turbo` and `gpt-4` are non-deterministic at the lowest 'temperature' setting [6,17].

5 Conclusions

This paper offers significant insights into the application of LLMs in ontology engineering, in particular for the retrofitting of competency questions from published ontologies. We conducted a study to assess how well LLMs can capture the scope of an ontology. Our analysis confirms that the use of explicit knowledge (ontology triples) paired with specific prompts is effective in generating valid CQs and that these results are independent of the creativity parameter settings: in particular, contextual information is effective in enhancing the precision and coherence of LLM responses, while just using the default creativity setting of the models used produces focused responses, therefore mitigating against the inherent non determinism of these models. In this study we focus on GPT models, and an extended study is currently in preparation to incorporate more LLMs.

References

1. Alharbi, R., et al.: An experiment in retrofitting competency questions for existing ontologies. In: Proc. of the 39th ACM/SIGAPP Symposium On Applied Computing (to appear) (2024)
2. AlKhamissi, B., et al.: A review on language models as knowledge bases. CoRR (2022). https://doi.org/10.48550/ARXIV.2204.06031
3. Allen, B.P., et al.: Identifying and consolidating knowledge engineering requirements. CoRR (2023). https://doi.org/10.48550/ARXIV.2306.15124
4. Allen, B.P., et al.: Knowledge Engineering Using Large Language Models. Transactions on Graph Data and Knowledge **1**(1), 3:1–3:19 (2023)
5. Bezerra, C., Freitas, F.: Verifying description logic ontologies based on competency questions and unit testing. In: Proc. of the IX Seminar on Ontology Research and I Doctoral and Masters Consortium on Ontologies. vol. 1908, pp. 159–164 (2017)
6. Chann, S.: Non-determinism in gpt-4 is caused by sparse moe (2023), `https://152334h.github.io/blog/non-determinism-in-gpt-4/`, accessed on January 20, 2024
7. Clandinin, D.: Handbook of Narrative Inquiry: Mapping a Methodology. SAGE Publications, Inc, Thousand Oaks, California (2007). https://doi.org/10.4135/9781452226552
8. Fernández-Izquierdo, A., et al.: Coral: A corpus of ontological requirements annotated with lexico-syntactic patterns. In: Proc. of the 16th International Conf. on The Semantic Web, ESWC 2019. pp. 443–458 (2019)
9. Grüninger, M., Fox, M.S.: The Role of Competency Questions in Enterprise Engineering, pp. 22–31. Springer US (1995)
10. Keet, C.M., Ławrynowicz, A.: Test-driven development of ontologies. In: Proc. of the 13th International Conf. on The Semantic Web, ESWC 2016. pp. 642–657 (2016)
11. Keet, C.M.: The african wildlife ontology tutorial ontologies. Journal of Biomedical Semantics **11** (2019), `https://api.semanticscholar.org/CorpusID:219981977`
12. La Malfa, E., et al.: Language models as a service: Overview of a new paradigm and its challenges. CoRR **abs/2309.16573** (2023)
13. Liu, P., et al.: Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. ACM Computing Surveys **55**(9) (2023)
14. Mulla, N., Gharpure, P.: Automatic question generation: a review of methodologies, datasets, evaluation metrics, and applications. Progress in Artificial Intelligence **12**(1), 1–32 (2023)
15. Noy, N., et al.: Ontology development 101: A guide to creating your first ontology. Tech. rep., Stanford knowledge systems laboratory technical report KSL-01-05 (2001)
16. Ouyang, L., et al.: Training language models to follow instructions with human feedback. In: Proc. of the Advances in Neural Information Processing Systems, NeurIPS 2022. vol. 35, pp. 27730–27744 (2022)
17. Ouyang, S., et al.: Llm is like a box of chocolates: the non-determinism of chatgpt in code generation. arXiv (2023)
18. Pan, J.Z., et al.: Large Language Models and Knowledge Graphs: Opportunities and Challenges. Transactions on Graph Data and Knowledge **1**(1), 2:1–2:38 (2023)
19. Parkkila, J., et al.: An ontology for videogame interoperability. Multimedia tools and applications **76**(4), 4981–5000 (2017)

20. Poveda-Villalón, M., et al.: Lot: An industrial oriented ontology engineering framework. Engineering Applications of Artificial Intelligence **111**, 104755 (2022)
21. Presutti, V., et al.: Extreme design with content ontology design patterns. In: Proc. of the 2009 International Conf. on Ontology Patterns. vol. 516, p. 83–97 (2009)
22. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In: Proc. of the 2019 Conf. on Empirical Methods in Natural Language Proc. and the 9th International Joint Conf. on Natural Language Proc. (EMNLP-IJCNLP). pp. 3982–3992 (2019)
23. Sequeda, J.F., et al.: A pay-as-you-go methodology to design and build enterprise knowledge graphs from relational databases. In: Proc. of the 18th International Semantic Web Conf., ISWC 2019. pp. 526–545 (2019)
24. Shin, T., et al.: AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In: Proc. of the 2020 Conf. on Empirical Methods in Natural Language Processing (EMNLP). pp. 4222–4235. Association for Computational Linguistics (2020)
25. Suárez-Figueroa, M.C., et al.: The neon methodology framework: A scenario-based methodology for ontology development. Applied ontology **10**(2), 107–145 (2015)
26. Wiśniewski, D., et al.: Analysis of ontology competency questions and their formalizations in sparql-owl. Journal of Web Semantics **59**, 100534 (2019)